

Oracle® Communications
Evolved Communications Application Server
Operator's Guide
Release 7.1
E66295-01

May 2016

Copyright © 2015, 2016, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface	ix
Audience	ix
Documentation Accessibility	ix
Accessing Oracle Communications Documentation	ix
Related Documents	ix
1 Using Oracle Communications Evolved Communications Application Server	
About the Session Design Center	1-1
Accessing Session Design Center	1-1
Using the Session Design Center Console	1-2
Using Control Flows, Resources and Change Sets	1-2
Provisioning and Accessing Subscriber Data	1-3
2 Working with Control Flows	
About the VoLTE and VoWiFi Application	2-1
About the Message Control Flows	2-1
About the Application Control Flows	2-1
About Control Flow Activities	2-2
About Using Control Flows	2-3
About the Control Flow Editor	2-3
About the Editing Toolbar	2-3
About the Information Panel	2-4
About the Changes Panel	2-4
Creating a Control Flow	2-5
Creating a New Control Flow	2-5
Constructing the Control Flow	2-6
Connecting Activities	2-6
Configuring an Activity	2-7
About Compiling Control Flows	2-7
Checking for Errors	2-7
Reusing Control Flow Logic	2-7
Copying an Activity	2-8
Copying a Group of Activities	2-8
Creating an SNMP Event	2-8
About the Object Identifier	2-9

Creating an SNMP Event in a Control Flow	2-9
Importing and Exporting a Control Flow	2-10
Modifying a Control Flow	2-10
Locating a Control Flow	2-10
Modifying a Control Flow	2-10
Reverting Changes in a Control Flow	2-10
Deleting a Control Flow	2-11
Working with External Concepts	2-12
Adding an External Concept	2-14
About Charging Concepts	2-14
Working with Charging Templates	2-14

3 Working with Resources

About Working with Resources	3-1
Working with Locales	3-2
About the Default Locale Data	3-2
Adding a Locale	3-2
Working with Media Resources and Servers	3-2
About Using Media Resources	3-3
About the Sample Media Resources	3-3
Adding a Media Resource to a Change Set	3-3
About the Media Resource Data	3-4
Media Files for Media Resources	3-5
Specifying the Default Locale	3-5
Specifying Other Locales	3-5
Updating the Setup for a Media Resource	3-6
Working with Media Servers	3-6
About the Sample Media Server	3-6
Adding a Media Server to a Change Set	3-8
Updating the Setup for a Media Server	3-9
Working with Prefix Trees	3-9
About the Default Prefix Tree	3-9
About the Prefix Tree Data	3-9
Creating a Custom Prefix Tree	3-10
Adding to a Prefix Tree	3-10
Removing a Branch from the Prefix Tree	3-11
Working with Templates	3-12
About the Default Notification Template	3-12
Working with Notification Templates	3-13
About the Notification Template Data	3-13
Adding a Notification Template	3-13
Updating a Notification Template	3-14
Working with Web Services Templates	3-14
Creating a Web Service Template Manually	3-14
Importing a WSDL File	3-15

4 Working with Change Sets

Managing Change Sets	4-1
Creating a Change Set	4-1
Locating and Viewing Details for a Change Set	4-2
Modifying a Change Set	4-2
Modifying the Baseline for a Change Set	4-2
Viewing the Changes to a Change Set	4-3
Bundling Change Sets	4-3
Adding Change Sets to a Bundle	4-3
Working with a Change Set Bundle	4-4
Errors When Deploying a Change Set Bundle	4-4
Copying Change Sets	4-4
To Copy a Change Set in Projects	4-4
To Copy a Change Set in Environments	4-5
Viewing Copy Progress	4-5
Errors When Copying	4-5
Deploying Change Sets	4-6
About the Deploying to the Production Pipeline	4-6
Deploying a Change Set	4-6
Deploying a Change Set to the Testing Environment	4-7
Deploying a Change Set to the Staging or Production Environment	4-7
Correcting Problems in Deployed Change Sets	4-8
Correcting a Change Set Without Stopping Service	4-8
Correcting a Change Set by Removing the Service Immediately	4-8
Retracting a Change Set	4-8
Redeploying a Change Set	4-9

5 Working with Subscriber Data

About Subscriber Data	5-1
Types of Subscriber Data	5-1
Accessing Subscriber Profile and Services Data	5-1
Accessing Supplementary Services Data	5-2
Accessing SRVCC Data	5-2
Federating the View	5-3
Translating HSS Data	5-5
Adding a Plug-in Parser	5-5
Configuring OCECAS Data Stores	5-6
Adding an HSS Subscriber Data Store	5-7
Configuring OCECAS to Accept Data from a Diameter Sh HSS	5-7
Advanced Configuration of Your ESS NoSQL Subscriber Data Store	5-8
Adding Additional Data Sources and Formats to the View	5-9
Replacing the ESS NoSQL Database	5-9
Adding a New UDR Using a New Data Type	5-9
Using Different Types of Data from the Same Data Provider	5-9
Changing the Default ESS NoSQL Schema	5-9
Changing the Default ESS Schema	5-9

Updating Federation Script Behavior During Runtime	5-10
About the OCECAS Diameter Sh Interface	5-10
Configuring Subscriber Data Quickly for a Test or Demonstration System	5-10
Provisioning the ESS with Subscriber Data	5-10
Connecting to the ESS Server	5-11
Creating a New Subscriber	5-11
Getting Subscriber Data	5-12
Amending Subscriber Data.....	5-13
Deleting a Subscriber	5-13
Getting Subscriber Aliases	5-13
Updating a Subscriber Alias.....	5-14
Getting jsonData for a Subscriber	5-15
Adding jsonData for a Subscriber.....	5-15

6 Accessing Service Data with the RESTful API

About Accessing Service Data	6-1
Logging In to the Management Domain.....	6-2
Obtaining a CSRF Token	6-2
Submitting the CSRF Token	6-3
Creating a Change Set	6-3
Getting Change Set Details	6-4
Getting a Change Set	6-4
Getting Change Set Differences	6-4
Getting a List of Environments	6-4
Getting a List of Pipelines	6-4
Deploying a Change Set	6-5
Getting Telemetry Records	6-6
Getting a Named Telemetry Record.....	6-6
Getting Statistics Definitions	6-6
Getting Statistics.....	6-6
Exporting a Control Flow	6-7
Updating a Control Flow	6-7
Importing a Control Flow	6-7

A The Default ESS Subscriber Store Schema

Understanding the Default ESS Objects	A-1
Default Subscriber Record	A-4

B Activities Reference

Add EDR Field Value	B-1
Adjust Media.....	B-2
Alarm	B-3
Array Index.....	B-4
Compare	B-4
Compare Date Time	B-5
Compare Day of Week	B-6

Compare List And Value	B-7
Copy Value	B-7
Date Time Offset	B-8
Delete	B-9
End.....	B-9
End Charging Session	B-9
Event Charge	B-10
Extract And Store String.....	B-10
Find and Replace And Store Value.....	B-11
Generate Document and Store.....	B-12
Increment Statistic.....	B-12
Load Service Definition	B-13
Maths	B-13
Prefix Tree Lookup.....	B-14
Release	B-15
Remote Copy	B-16
Retrieve Session List.....	B-16
Route	B-17
Route Changed.....	B-18
Run Control Flow	B-18
Run Service Definition	B-18
Run Web Service	B-19
Send Inter-Session Event.....	B-20
Send Message.....	B-21
Send Subscription Notify	B-21
Start	B-22
Start Back to Back Session.....	B-23
Start Charging Session	B-24
Start Collecting Digits	B-24
Start Conference Session	B-25
Start Forked Session	B-26
Start Subscription Notifier Session.....	B-26
Start Playing Media	B-27
Start Recording Message.....	B-28
Start Timer	B-29
Statistic Branching	B-29
Stop Media.....	B-30
Store	B-31
Store Session Key	B-31
Subscribe Event Package	B-32
Sync Statistic	B-33
Telemetry	B-33
Translate and Store Value	B-34
Update Charging Session	B-35
Update Profile	B-35
Wait For Event.....	B-36
XCAP Authorize	B-36

XCAP Fetch And Store Document	B-37
XCAP Process Document	B-37
XCAP Update Document	B-38

C External References for Supported Interactions

Preface

This document describes the tasks that an operator performs to create and use services with Oracle Communications Evolved Communications Application Server (OCECAS).

Audience

This document is intended for an OCECAS operator who is tasked with adding subscribers to OCECAS and implementing services for them.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Accessing Oracle Communications Documentation

OCECAS documentation is available from the Oracle Documentation Web site: <http://docs.oracle.com>.

Related Documents

For more information, see the following OCECAS documentation:

- *Oracle Communications Evolved Communications Application Server Release Notes*
- *Oracle Communications Evolved Communications Application Server Installation Guide*
- *Oracle Communications Evolved Communications Application Server Concepts*
- *Oracle Communications Evolved Communications Application Server System Administrator's Guide*
- *Oracle Communications Evolved Communications Application Server Security Guide*
- *Oracle Communications Evolved Communications Application Server Compliance Guide*

- *Oracle Communications Evolved Communications Application Server RESTful API Reference*
- *Oracle Fusion Middleware 12c Documentation Library*
- *Oracle Database Installation Guide 12c Release 1 (12.1) for Linux*
- *Oracle Database Administrator's Guide 12c Release 1 (12.1)*

Using Oracle Communications Evolved Communications Application Server

This chapter describes how to access and use Oracle Communications Evolved Communications Application Server (OCECAS). Using OCECAS consists of using the Session Design Center to create and access subscriber services and using the subscriber data store to provision and access subscribers.

For an introduction to OCECAS and an overview of Session Design Center and the subscriber data store, see *Oracle Communications Evolved Communications Application Server Concepts*. For information on how to configure and administer OCECAS, see *Oracle Communications Evolved Communications Application Server System Administrator's Guide*.

For a description of scenarios and corresponding call-session flows that OCECAS supports, see "[External References for Supported Interactions](#)".

About the Session Design Center

Session Design Center runs in the OCECAS management domain. It is a graphical environment that enables you to create and manage application service logic without writing code.

Accessing Session Design Center

To log in to OCECAS and access Session Design Center:

1. In your web browser, enter the URL to go to the Session Design Center.

For example:

```
https://hostname:port/sdc/login.html?target=
```

The OCECAS login page displays.

2. In the **User Name** field, enter the user name for your account.
3. In the **Password** field, enter the password.
4. Click **Sign In**.

Once you successfully log in, the Home page, which is also known as the landing page, displays.

If your interaction with Session Design Center is idle for 30 minutes, the application logs you out.

Using the Session Design Center Console

The Session Design Center Home page includes the following elements:

- Header bar

The bar at the top of the interface that includes the name of the application, a **Help** link, and a **user** drop-down menu, which allows you to log out.

- Navigation panes

Below the header bar, the following panes provide access to principal areas of functionality:

- **Application** contains a **View Application Configuration >** link that provides access to application control flows and resources.

The Application section contains the VoLTE and VoWiFi application, all of the control flows that have been created for the application, and all of the resources that are available to the application. The VoLTE and VoWiFi application is the core of OCECAS's capabilities. You tailor OCECAS to your customer's requirements by creating control flows to add subscriber services.


- **Projects**, or **Open Projects**, provides access to change sets.

If open projects exist, the title of the section becomes **Open Projects** and links to any open projects are listed below. A **View All Projects >** link takes you to a list of all change sets for the application.

The Projects section contains all of the change sets, which are also known as projects, that have been created to make changes to the application. A change set is a group of related configuration and control flow changes.

- **Environments** provides access to the deployment pipeline, which consists of Testing, Staging, and Production environments. Use the **View Environment Configuration >** link to view and access projects in their stages of deployment.

You deploy change sets sequentially into environments to safely migrate them into production. Each environment also provides access to its service configuration.

To return to the **Home** page at any time, click the Home icon () displayed below the Header bar.

WARNING: Do not open multiple instances of Session Design Center in different browser tabs or windows.

Using Control Flows, Resources and Change Sets

The remainder of the Session Design Center interface is dedicated to working with control flows, resources, and change sets. See the following chapters for more information on using the Session Design Center:

- [Working with Control Flows](#)
- [Working with Resources](#)
- [Working with Change Sets](#)

Provisioning and Accessing Subscriber Data

For information on working with the subscriber data store to provision and access subscriber data, see "[Working with Subscriber Data](#)".

For a detailed description of the subscriber data store schema, see "[The Default ESS Subscriber Store Schema](#)".

Working with Control Flows

This chapter describes how to create and work with control flows to augment the services provided by the Volte and VoWiFi application in Oracle Communications Evolved Communications Application Server (OCECAS).

For an introduction to the Session Design Center and an overview of control flows, see “About the Session Design Center” in *Oracle Communications Evolved Communications Application Server Concepts*.

About the VoLTE and VoWiFi Application

The VoLTE and VoWiFi application provides session control and services for voice calls and multimedia sessions over Long Term Evolution (LTE) and for voice calls over WiFi. To define new services and behaviors for subscribers, you make changes to the VoLTE and VoWiFi application by creating or modifying control flows. You create and modify control flows in the context of a change set. For more information about change sets and change management, see "[Working with Change Sets](#)".

The VoLTE and VoWiFi application consists of four message control flows and five application control flows.

About the Message Control Flows

Message control flows are triggered when the application server receives one of the following SIP messages: HTTP DELETE, HTTP GET, HTTP POST, HTTP PUT, SIP INVITE, SIP OPTIONS, SIP REGISTER, or SIP SUBSCRIBE. Each of these messages triggers a control flow of the same name. For descriptions of these message control flows, see “About Applications” in *Oracle Communications Evolved Communications Application Server Concepts*.

You can view the Message Control Flows by clicking **View Application Configuration** from the SDC home page and then clicking the name of the Message Control Flow that you want to view under the Application tab.

About the Application Control Flows

The VoLTE and VoWiFi Application control flows provide supplementary services and manage the phases of a session life cycle. The Session Origination and Session Termination control flows invoke additional control flows to deliver supplementary services. Supplementary services are services that supplement the telephony and data services.

The Session Deregistration control flow manages the life cycle of deregistration.

The Session Origination control flow manages the life cycle of the originating session. It executes additional control flows to invoke the following supplementary services:

- Ad-hoc Conferencing - allows a subscriber to transform a one-to-one call session into a multi-party call session
- Communication Hold Charging
- Communication Hold (origination)
- Originating Identity Restriction
- Outgoing Communication Barring
- Terminating Identity Presentation

The Session Registration control flow manages the life cycle of registration.

The Session Termination control flow manages the life cycle of the terminating session. It executes additional control flows to invoke these supplementary services:

- Anonymous Communication Restriction
- Communication Diversion
- Communication Hold (termination)
- Early Communication Diversion
- Incoming Communication Barring
- Originating Identity Presentation
- Terminating Identity Restriction

The Terminating Access Domain Selection control flow selects the circuit-switched access or packet-switched access network(s) to use to deliver a terminating voice session to the user endpoint.

You can add Activities to the **VoLTE, VoWiFi and eSRVCC** session registration and de-registration control flows to perform specific tasks required by a customer in response to registration or de-registration events.

About Control Flow Activities

A session control flow consists of logically linked Activities, such as **Start, Run Web Service, Start Charging Session, Start Conference Session**, and so on. You must configure an Activity when you add it to a control flow.

Configuring an Activity consists of specifying values for the following elements:

- **Name**

This is the Activity name, which can be the default name or a custom name that you assign. When you assign a custom name, the control flow editor retains the default name for internal use and displays it in smaller grey text.
- **Comments**

You can add a short, optional comment to describe the Activity.
- **Parameters**

An Activity can require a set of parameters for which you need to provide appropriate values.

All required parameters must be provided with values for the control flow to compile successfully.

- **Exits**

Except for End, each Activity has one or more exits, which you must connect to the next logical action. All exits must have a connection for the control flow to compile successfully.

You can define dynamic exits for the Wait for Events Activity. A dynamic exit is one that you add to the Activity at the time you include the Activity in the control flow.

For a complete list with descriptions of the control flow Activities, including descriptions of their parameters and exits, see "[Activities Reference](#)".

About Using Control Flows

To define new services and behaviors for subscribers, you make changes to the VoLTE and VoWiFi application by creating or modifying control flows. A control flow consists of logically-linked actions (Activities) and decisions that specify how to process a voice call or multimedia session.

To understand the basic concepts of control flows, see "About the Session Design Center" in *Oracle Communications Evolved Communications Application Server Concepts*.

About the Control Flow Editor

When you open an existing control flow or create a new one, you open the control flow editor and the editing canvas. The editing canvas is the work area on which you create or edit the control flow logic. The canvas has vertical and horizontal scroll bars

The editing canvas also includes the following parts:

- The Editing Toolbar
- The Information panel
- The Changes panel

About the Editing Toolbar

[Table 2–1](#) lists the tool icons that can be found on the editing toolbar, which appears on the left side of the canvas:

Table 2–1 *Editing Toolbar Icons*

Icon	Function
Spyglass (search)	Search the control flow. Clicking it opens an adjoining panel with a Search field followed by a list of control flows contained in the changed set.
Pencil (edits)	Shows control flow differences. Clicking opens a panel that details the differences between the current change set and the baseline change set.
Circled Red Exclamation Point (warning)	Shows issues found in this control flow. Clicking it opens a list of control flows having errors, along with descriptions of each of the errors found.
Save	Manually saves the control flow.

Table 2-1 (Cont.) Editing Toolbar Icons

Icon	Function
Undo	Undoes (reverts) the last change. Greyed out if no changes have been made.
Redo	Restores the last change that was undone. Greyed out if nothing has changed.
Scissors	Cuts (removes) an item from the canvas and saves it to the clipboard.
Copy	Copies an item from the canvas and saves it to the clipboard. See "Reusing Control Flow Logic" .
Paste	Pastes an item from the clipboard to the canvas. See "Reusing Control Flow Logic" .

Note: For a read-only control flow, only the Spyglass and Copy icons are available. The Copy icon is inactive until you select something to copy.

About the Information Panel

The Information panel is adjacent to the right side of the editing toolbar. The Information panel opens when you click the Spyglass icon, the Pencil icon, or the Warning icon in the editing toolbar.

When you click the Spyglass (search) icon in the editing toolbar, the Information panel displays a search field that allows you to search for an Activity in the control flow. Enter a string of characters to be matched against an Activity name. The control flow editor tries to match the string against both the original name of the Activity and its assigned name, if the name has been changed. If an Activity occurs more than once, the panel displays all occurrences of that Activity. When you click an entry in the list, the Activity is highlighted on the canvas.

When you click the Pencil (edits) icon, the Information panel shows the differences between the current control flow and the control flow in the baseline change set. For a new control flow that you created in this change set, the panel shows no differences. If you are modifying a control flow that was created in an earlier change set, the panel shows the differences between the current and original control flows.

When you click the Warning icon in the editing toolbar, this panel displays the Activities in the control flow that have issues and require corrections.

About the Changes Panel

The Changes panel is the right-most panel on the editing canvas. It shows the changes in the change set. If a control flow is created in a change set, no differences are shown. If a control flow is inherited from the base change set and is modified in the current change set, the Changes panel shows the differences. If a control flow is inherited from the base change set and is deleted in the current change set, the Changes panel displays the control with a retract symbol, allowing you to restore it to the change set. The control flow appears in the **All Control Flows** tab with a line through it and a retract symbol next to it. See ["Working with Change Sets"](#) for more information.

Click the arrow next to a category to expand the list of changes to it.

The Changes panel shows any changes, including creations, modifications, or deletions to the entities in the following list.

- Control Flows
- Concepts
- Charging Templates
- EDRs
- Import WSDL
- Locales
- Media Resources
- Media Servers
- Notifications
- Prefix Trees
- Schemas
- Statistics
- Unclassified Service Data
- Web Services

For **CONTROL FLOWS**, the Changes panel displays the following alert for unconnected exits:

Unconnected exit (*N*)

where *N* is the total number of unconnected exits in the control flow.

Creating a Control Flow

You must create or modify a control flow within a change set. A change set is a logical grouping of configuration and control flow changes that makes it easier to organize and deploy changes.

Note: You cannot create a control flow for a change set whose deployment status is **Active**.

For an understanding of change set concepts, see "About Change Management" in *Oracle Communications Evolved Communications Application Server Concepts*.

Creating a New Control Flow

To create a new control flow for a change set:

1. Open the change set. See "[Locating and Viewing Details for a Change Set](#)".
2. Click **All Control Flows**.
3. Click **New**.
4. In the New Control Flow dialog box, enter the following.
 - a. In the **Name** field, a unique name for the control flow.
 - b. In the **Description** field, a short description.
5. Click **Create**.

The control flow editor places the **Start** Activity icon on the canvas.

Constructing the Control Flow

Constructing a control flow consists primarily of two operations: connecting and configuring Activities.

Connecting Activities

You construct control flow logic by connecting Activities to create a logical processing sequence for a call or multimedia session. You can create multiple branches in a control flow based on Activity outcomes such as Success, Error, Busy, No Answer, Waiting for Event, and so on.

To connect Activities:

1. Place your cursor on an unconnected exit for an Activity.

Tip: All unconnected exits are displayed in red at the bottom of the Activity icon.

2. Drag your cursor to the location where you want to place the next Activity and release it.

The Activities selection window displays.

3. Select the next Activity in the logic flow in one of the following ways:
 - Use the vertical scroll bar of the Activities selection window to scroll to the Activity you want.
 - Enter a search string in the Search field to filter the available Activities to those that have matching text in either their name or description.
4. Click the desired Activity to select it.

The selected Activity is placed on the control flow canvas with a line connecting it to the selected exit of the preceding Activity. The line represents the path that will be taken when that exit condition occurs.

Note: You can connect multiple exits to the selected Activity.

Note: You must terminate each branch in the control flow with an End Activity.

Tip: When you add an Activity:

- The Search display panel display is updated.
- The Changes panel updates its counter of undefined values.

5. Configure the Activity, if necessary. See "[Configuring an Activity](#)" for more information.
6. Repeat steps 1 through 5 to add additional Activities until the control flow is complete.
7. Click **Save**.

Configuring an Activity

To configure an Activity, you can assign it a different name, add comments to it, and specify values for the Activity's parameters

To configure an Activity:

1. Double-click the Activity to display the Activity detail panel.
2. To change the name of the Activity:
 - a. Click the pencil icon next to the name.
 - b. Enter a unique name.
 - c. Click the check mark.

Note: OCECAS also retains the original name of the Activity for internal use.

3. To add comments or notes to the Activity:
 - a. Click the pencil icon next to **Add comments**.
 - b. Enter your comments.
 - c. Click the check mark.
4. In the **Parameters** section, provide values for each entry. See "[Configuring an Activity](#)" for more information.
5. In the **Exits** section, ensure that all exits for this Activity are connected.

About Compiling Control Flows

An editable control flow is automatically saved, validated and compiled whenever you make a change to the flow in the control flow editor. For a control flow to compile successfully, all the Activities in the control flow must contain valid configuration data and all Activity exits must be connected.

Checking for Errors

The most common causes of errors in a control flow are Activity parameters that have not been set and Activity exits that have not been connected. For more information about errors, see "[About the Changes Panel](#)" and "[About the Information Panel](#)".

The control flow editor also writes information to event detail records when a control flow is compiled. For more information, see "Understanding EDRs" in *Oracle Communications Evolved Communications Application Server*.

Reusing Control Flow Logic

You can reuse control flow logic in more than one location by using the copy and paste tools in the editing tool bar.

Important:

- The target location can be in the same control flow, in another control flow of the same change set, or in another change set.
To paste control flow logic into another change set, the target change set must be editable.
 - Configure each Activity after you paste in the target location. Ensure that all parameters and exits are defined.
-
-

Copying an Activity

To create a copy of an Activity:

1. In the source control flow, locate the Activity group.
2. Click the Activity to select it.
3. Right-click and select **Copy**.
4. Go to the target location.

As stated earlier, the target location for the Activity should be an editable change set.

5. Right click and select **Paste**.
6. Configure the Activity in the target location.

Copying a Group of Activities

To copy a group of connected Activities:

1. In the source control flow, locate the Activity group.
2. Move the cursor to the Activity that is the starting Activity for this group.
3. Left-click on the mouse and drag it to include the set of connected Activities.
4. Stop and release the cursor.

The set of connected Activities are selected.

5. Right-click and select **Copy**.
6. Go to the target location.
The target location for the Activity group must be an editable change set.
7. Right-click the mouse and select **Paste**.
8. In the target location, configure each Activity in the copied set.

Creating an SNMP Event

You can create an SNMP event in a control flow by including the SNMP Event Activity. [Table 2-2](#) describes the attributes that make up an SNMP event definition:

Table 2-2 *SNMP Event Attributes*

Attribute	Description
OID	A event object identifier, that uniquely identifies fields in an SNMP message.
eventName	A text-based event name.

Table 2–2 (Cont.) SNMP Event Attributes

Attribute	Description
severity	A suggested severity level. This value is for informational purposes. It is not passed in the SNMP trap. The responsibility Network Management System (NMS) determines whether an event should be turned into an alarm, as well as what severity to assign the alarm.

See “About SNMP Events” in *Oracle Communications Evolved Communications Concepts* for an overview of OCECAS SNMP Events. See “Managing SNMP Events” and “SNMP Events Reference” in *Oracle Communications Evolved Communications System Administrator’s Guide* for information on managing SNMP Events and a complete list of OCECAS SNMP Events.

About the Object Identifier

The SNMP event object identifier (OID) consists of a fixed prefix that is used for all OCECAS SNMP events and a postfix that identifies a specific event. The default fixed prefix for SNMP events issued by OCECAS consists of the following elements:

```
Iso(1).org(3).dod(6).internet(1).private(4).IANA
Registered(1).oracle(111).productId(10)
```

You can change the value of the prefix to your own internal value through the Administration Console. For more information, see “Specifying SNMP Event Options” in *Evolved Communications System Administrator’s Guide*.

The postfix of the OID consists of these elements:

```
app-specific(6).scf(1).action(X).sip(X).actionInvalidState(X)
```

where the value of X varies by the type of event. For example, the postfix for the error ‘Could not send CCR: {}’ is the following:

```
app-specific(6).scf(1).action(7).sip(1).chargingRequestFailed(1)
Or
6.1.7.1.1
```

For a complete list of OCECAS SNMP Event OIDs, see “SNMP Events Reference” in *Oracle Communications Evolved Communications System Administrator’s Guide*.

Creating an SNMP Event in a Control Flow

A control flow event must use the following specific OID definition:

```
Iso(1).org(3).dod(6).internet(1).private(4).IANA
Registered(1).oracle(111).productId(10).app-specific(6).control-flow(2).event(X)
```

Where X represents the event ID. For example, the OID for a control flow with an event ID of 23 would be:

```
Iso(1).org(3).dod(6).internet(1).private(4).IANA
Registered(1).oracle(111).productId(10).app-specific(6).control-flow(2).event(23)
```

You specify the event ID when you add the SNMP Event Activity to a control flow. You can enter additional dot-delimited numbers if you need to further identify the event.

Importing and Exporting a Control Flow

You can import and export a control flow using the RESTful API. For information on importing a control flow, see ["Importing a Control Flow"](#). For information on exporting a control flow, see ["Exporting a Control Flow"](#).

Modifying a Control Flow

You must modify a control flow within a change set. You can either create a new change set or locate an existing change set to modify.

Note: If a change set has been deployed to the staging or production environment, you cannot modify its control flows.

Locating a Control Flow

To locate a control flow within a change set:

1. Select the change set. See ["Locating and Viewing Details for a Change Set"](#).
2. Locate the control flow in one of the following ways:
 - Click **All Control Flows**.
A table displays the current set of control flows in the change set. To sort the list, click a column heading. Locate the flow.
 - In the Changes panel to the right, click **Control Flows** to expand the section. Locate the flow.
3. Click the control flow entry.
The control flow opens on the editing canvas.

Modifying a Control Flow

To modify a control flow:

1. Locate the control flow in the change set. See ["Locating a Control Flow"](#).
2. Update the control flow as required. See ["Constructing the Control Flow"](#) for more information.
3. Check for errors:
 - In the Differences panel to the left, review the entries under the this control flow.
 - In the Changes panel, review the changes to the control flow.
4. Click **Save**.

Reverting Changes in a Control Flow

You can revert any changes that you have made to a control flow in a change set.


If you have created a new control flow in a change set, you will actually delete it—or, undo its creation.

If you have made changes to a control flow that is derived from the base change set, you discard or undo the changes. The control flow is then as it is in the base change set.

To revert changes to a control flow:

1. Locate the change set and open it. See ["Locating and Viewing Details for a Change Set"](#) for more information.
2. Do one of the following:
 - Select the **All Control Flows** tab
 - Click the **Control Flows** item in the Changes panel
3. Locate the control flow in the list.

Note: In the **All Control Flows** tab, control flows that display the eye icon to the far right side of the line are read-only. You cannot revert them.

4. Click the revert icon  to the right side of the control flow name to undo the changes to it.

If you are reverting a new control flow, a confirmation query displays to ask if you want to discard the control flow.

- Click **Discard** to delete the control flow.

Deleting a Control Flow

If you have not made any changes to a control flow, you will see a trash-bin icon beside it under the **All Control Flows** tab. You can click this icon to delete the control flow from the current change set. In this case, you have made a change to the control flow so you will see the revert icon beside it, which gives you a chance to revert or undo your deletion.

To delete a control flow:

1. Locate the change set containing the control flow and open it. See ["Locating and Viewing Details for a Change Set"](#) for more information.
2. Select the **All Control Flows** tab.
3. Locate the control flow in the list.

4. Click the trash-bin icon () to the right side of the control flow name.

A confirmation query displays to ask if you are sure that you want to delete the control flow.

5. Click **Delete** to delete the control flow.

The control flow name displays with a line through it and the revert symbol appears next to it, giving you the option to restore the control flow to the change set.

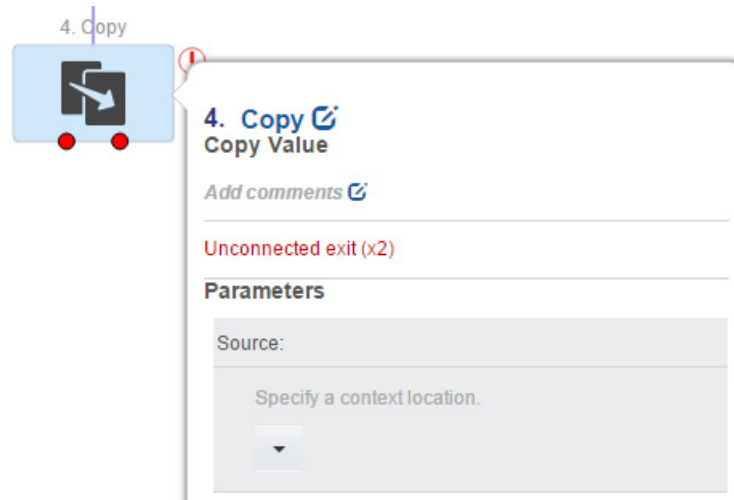
If you have created a new control flow in a change set, you can delete it by reverting it. See ["Reverting Changes in a Control Flow"](#) for more information.

Working with External Concepts

External concepts consist of various data sources that you can access through Activity parameters in a control flow. For example, you could access data from the inbound or outbound message headers of a call session.

Figure 2–1 illustrates how to access external concepts to set parameter values for an Activity. Double-click the Activity icon to open the configuration panel. See "Configuring an Activity" for more information on setting Activity parameters.

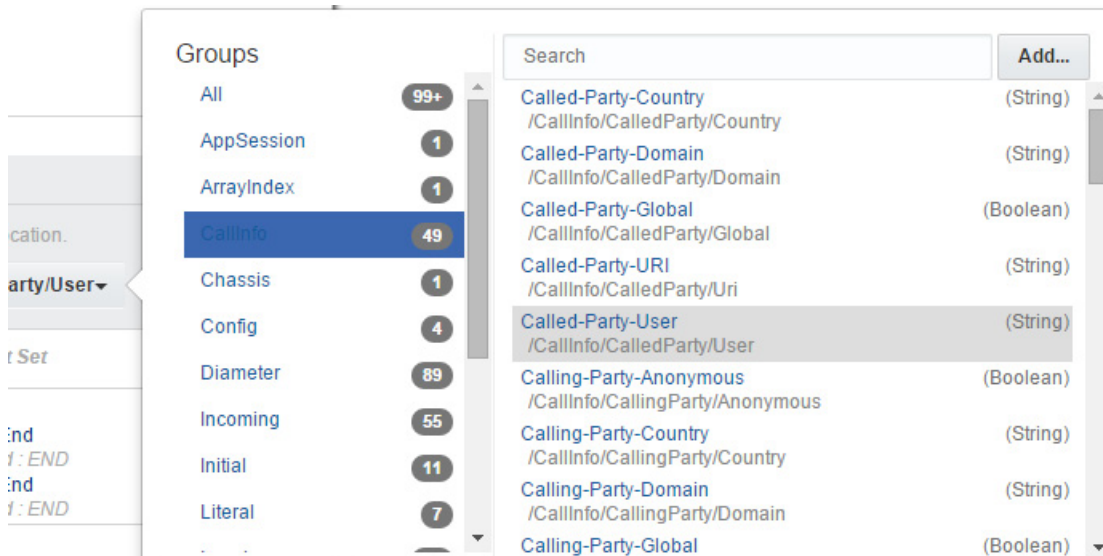
Figure 2–1 Activity Parameters



Click the parameter that you want to set and then click the down-arrow to open a list of context locations that reference the external concepts.

Figure 2–2 illustrates the groupings and the list of external concepts.

Figure 2–2 List of Groups and External Concepts



Select a particular group to filter external concepts to only those that belong to the group.

Enter a keyword in the Search field to narrow the list to those entries that contain the keyword in either the concept name or the context location.

Table 2–3 describes the groupings into which OCECAS organizes external concepts:

Table 2–3 External Concepts Groupings

External Concept Group	Description
AppSession	All data related to the SIP Application Session, including the Session ID and the last accessed time.
ArrayIndex	These values are used to define dummy array index values used in charging control flows.
CacheData	Provides context access to a Coherence cache. Currently used to store registration data, which stays in the cache until the registration expires. PrimaryKey and SecondaryKey must be set to access data in the cache.
Call Info	Information derived from the call, for example, the calling party and the called party.
Chassis	Data required for chassis.
Compiled Data	Control flow data derived from compilation.
Configuration Data	Data related to the configuration. Includes, for example, the ATU-STI url, the conference factory URI, whether charging is enabled, and so on.
Diameter Field Group	Incoming and outgoing credit control messages.
EDR Field Group	Data from an Event Detail Record.
Incoming Request	Data that is presented in an incoming request, including Call-ID, Timestamp, Request-URI, and so on.
Initial Request	Data presented in the initial request in an SIP header, for example the From, Call ID, P-Asserted-Service, and so on.
Local Context	Data that has been defined in the control flow.
Notified Field Group	Data from the fields of a notification message.
Outgoing Request	Data presented in an outgoing request, for example Contact, Expires, SDP video streams, and so on.
SDP Data	Data from the Session Description Protocol.
Service Data	Data used by services or applications, including service-specific data and general data, for example, the home network ID list for VoLTE and VoWiFi service, the T-ADS preferences for SRVCC service, country-code mapping for all services, and so on.
Service Loader	External concepts that are used by bootstrap control flows such as SIP INVITE, SIP OPTIONS, and so on, to decide which sub-application or service to run.
SIP Application Session Data	Data from the SIP Application session.
User Profile	Data from the user's profile, such as HomeCountry, TimeZone, the control flow names of registered services, Language, MSISDN, STN-SR, and so on.

OCECAS uses context locations to locate information associated with a session. For example, outgoing credit control messages that are created during a session are located with the information about Multiple Services Credit Control (MSCC) at /Diameter/Outgoing/MSCC.

You can also add a local context external concept to a control flow.

Adding an External Concept

You can add a local context external concept when you configure an Activity that accepts an external concept as a parameter value.

To add an external concept to a control flow:

1. Open the control flow.
See "[Locating a Control Flow](#)" for more information.
2. Double click the Activity whose parameter requires the external concept.
The configuration panel opens.
3. Click the parameter you want to set.
A dialog box displays the groups and a list of the external concepts.
4. Click **Add...** See [Figure 2-2](#) for an example.
The Add External Concept dialog box displays.
5. Enter the information to describe the external concept.
 - a. In the **Name** field, enter a unique name for the concept. Name must be alphanumeric characters plus -, _, and space.
 - b. In the **Path** field, specify the location of the runtime data, beginning with **/Local/**. For example:

```
/Local/new_concept_name
```


Path must be alphanumeric characters plus -, _, and /.
 - c. For the **Data Type** field, select from **String**, **Integer**, **Float**, **Boolean**, **Date**, **MapArray**, **StringArray**, **Integer Array**, and **FloatArray**.
6. Click **Confirm**.
The external concept is added and displays as a selection in the Local group.

About Charging Concepts

Charging concepts consist of data that results from some calculation such as the number of used units since the last request, or data that the Activities need to know explicitly, such as the validity time of the granted quota. OCECAS generates this information when it is required for an outgoing message or a message receipt. You normally use the request or response message template to copy these values to or from the context map for the session.

Working with Charging Templates

You need to reference the appropriate values for the **Request Template** and **Answer Template** parameter when you define the following Activities:

- **Start Charging Session**
- **Event Charge**
- **Update Charging Session**
- **End Charging Session**

For the template parameters, OCECAS provides four Charging templates called **ACAnswer**, **ACRequest**, **CCAnswer**, and **CCRequest**. Select a corresponding template for the Activity.

Working with Resources

This chapter describes how to use Session Design Center (SDC) resources in Oracle Communications Evolved Communications Application Server (OCECAS).

For an overview of OCECAS resources, see "About Resources" in *Oracle Communications Evolved Communications Application Server Concepts*.

About Working with Resources

OCECAS resources consist of data that you can use in control flows. These resources consist of data sets, announcements, notifications, and web services. [Table 3–1](#) describes the four categories into which resources are divided:

Table 3–1 OCECAS Resource Categories

Resource Category	Description
General	The General resource category consists of the Locales data set. Locales serve to identify the location and language for originating and terminating endpoints in a call. See "Working with Locales" for more information.
Media	The Media resource category consists of media resources, which are announcements, and the servers where they reside. See "Working with Media Resources and Servers" for more information.
Numbers	The Numbers category consists of prefix trees, including the default CountryCodePrefixTree, which maps global prefixes and country codes. See "Working with Prefix Trees" for more information.
Templates	<p>The Templates category consists of the following types of templates:</p> <ul style="list-style-type: none"> ■ Notifications A template that you can use in a control flow to generate a runtime notification. ■ Web Services A template to describe how to access a web service. ■ Import WSDL The Web Service Description Language import template enables you to create web services templates from content that has been imported from WSDL files. <p>See "Working with Templates" for more information.</p>

Working with Locales

Locales are used to specify the language used in media resources such as announcements. A locale consists of a country code and a language code and also identifies the originating or terminating location in a voice call or multimedia session.

About the Default Locale Data

The VoLTE and VoWiFi application supports a default set of locales. To access the default locales, select **View Application Configuration** on the SDC home screen, then click the **Resources** tab and select Locales in the General category.

Each locale consists of the following data elements:

- **Name**

The name of the language, such as Japanese. The name also allows for regional differentiation, such as Spanish (Mexico), Spanish (Guatemala), Spanish (Colombia) and so on.
- **Description**

The description entry allows for a more detailed description of the locale.
- **Language code**

The two letter language code that specifies the language, regardless of region. For example, **es** is the language code for Spanish, including Spanish (Mexico), Spanish (Guatemala), Spanish (Colombia) and so on. The language code is based on the ISO 639 standard.
- **Country code**

The two letter country code, for example, **MX** for Mexico, **GT** for Guatemala, **JP** for Japan, and so on. The country code is based on the ISO 3166 standard.

Adding a Locale

You can add a locale to a change set that is not yet deployed.

To add a locale to a change set:

1. Select the change set. See "[Locating and Viewing Details for a Change Set](#)".
2. On the page for the change set, click **All Resources**.
3. In the General category, click **Locales**.

This displays the set of existing locales.
4. Click **New**.

A dialog box opens, containing the fields Name, Description, Language Code, and Country Code.
5. Enter the information required to define the locale.
6. Click **Save**.

Working with Media Resources and Servers

OCECAS requires a media server from which it can access media resources, such as the announcements it must play to the parties in a call session. The VoLTE and VoWiFi application provides sample media resources that are associated with a sample media

server named VoLTE. To use these samples you must first configure them for your site. For example, you must update the driver of the example media server to match what you have. You must also update the .wav file locations of the sample media resources to match your actual media server configuration

To add a media server, see "[Adding a Media Server to a Change Set](#)" for more information.

To add needed announcements and store them on a media server, see "[Adding a Media Resource to a Change Set](#)" for more information.

About Using Media Resources

A media resource is used when the control flow for a call session initiates it.

To initiate the use of a media resource in a control flow, add either the **Start Playing Media** or **Start Recording Media** Activity. These Activities require you to specify a session endpoint, which is one of the following: Initiating Endpoint, Destination Endpoint, Event Source or Conference Endpoint. To play an announcement to the caller who started the session, for example, select Initiating Endpoint. You must also specify a language by entering it, or by requesting the program to select it from context. In the latter case, OCECAS uses the language parameter that is configured in the user profile under `/UserData/Subscriber/Profile` in the user database repository. Additionally, you select a media resource from the list of available media resources and specify how long the resource must play.

At run-time the request that is sent to the media server is constructed from the selected endpoint, the user's chosen language, and the configuration of the selected media resource. The list of .wav files specified for the user's language (locale) is sent to the media server in the request. The media server responds by playing the specified .wav files to the user.

As for recording a message, you might want to record a greeting message when your are unable to answer a call. You could also want to record a notification when you start a conference to tell callers what the conference is about?

About the Sample Media Resources

OCECAS provides the following sample media resources:

- **HoldMusic**
VoLTE Hold Music announcement
- **ICBMessage**
VoLTE Incoming Call Barred Message announcement
- **OCBMessage**
VoLTE Outgoing Call Barred Message announcement
- **SessionEndedMessage**
VoLTE Session Ended Message announcement

To use these sample resources, you must update the definitions to apply to your site. See "[About the Media Resource Data](#)" for more information.

Adding a Media Resource to a Change Set

To add a media resource to a change set:

1. Select the change set. See ["Locating and Viewing Details for a Change Set"](#).
2. On the page for the change set, click **All Resources**.
3. In the displayed selection, click **Media Resource**.
The page displays the set of default media resources that are available.
4. Click **New**.
A dialog box opens.
5. Enter the required information in the dialog box. See ["About the Media Resource Data"](#).
6. Click **Save**.

About the Media Resource Data

[Table 3–2](#) lists the data required to define a media resource.

Table 3–2 Media Resource File Data

Entry	Description
Name	A required field. Enter a unique name.
Description	Enter a description of the media resource.
Resource Type	Specify the resource type as one of the following: <ul style="list-style-type: none"> ▪ Play ▪ Record ▪ Collect
Media Server	The name of the media server. See "Adding a Media Server to a Change Set" for information on adding a media server. OCECAS includes a sample media server called VoLTE that you can use as an example.

Each media resource has zero or more parameters that you populate when you use the resource in a control flow. Consequently, when you create the resource, each parameter is a placeholder so you only need to specify its name and type, along with an optional description. The control flow editor allows you to specify a value for each parameter when you select the resource within a control flow.

[Table 3–3](#) lists the entries to define a media resource parameter:

Table 3–3 Media Resource Parameters

Entry	Description
Name	A unique parameter name.
Data Type	The parameter's data type. Select one of the following types: <ul style="list-style-type: none"> ▪ String ▪ Integer ▪ Float ▪ Boolean ▪ Date
Description	A description of the parameter.

You must also specify the locales that the media resource supports. For each locale you specify an ordered list of media files that will be played to the user when the media resource is used. The locale is selected at run-time based on user preference; if not known, the Default locale is selected. See "[Media Files for Media Resources](#)" for more information.

Media Files for Media Resources

For each locale, you specify a list of URIs. The media server uses the URI to locate a file or resource to play. Each URI is configured as a combination of a protocol that you select from a drop-down list and a body that you enter, which names the file.

The supported protocols associated with the retrieval of the resource are:

- **file://**, to use with a resource on the file system of the media server.
- **http://**, to use with a resource to be retrieved over HTTP by the media server.
- **https://**, to use with a resource to be retrieved over HTTPS by the media server.
- **data://**, to use data URI encoding scheme to include snippets of data directly. For information about the URI data encoding scheme, see

<https://tools.ietf.org/html/rfc2397>

- **other://**, to use when you specify the entire URI, including the protocol.

OCECAS forms the URI by combining the selected protocol with the text that you enter in the body (which names the file). For example, selecting **file://** and entering `//hello/there.wav`, results in the URI: `file:///hello/there.wav`. For **other://**, if you specify text in the body as `xyz://joe@bloggs.com/foo`, then the uri will be `xyz://joe@bloggs.com/foo`.

At run-time the locale is selected based on user preference; if not known the Default locale is selected.

Specifying the Default Locale

To specify the Default locale:

1. Click **Default**.
2. Click **New**.
3. Select a protocol from the drop-down list.
4. Enter a body to specify the location and name of the media file.
5. Repeat steps 2 through 4 to add URIs for additional media files.

Specifying Other Locales

To specify other locales:

1. Click the plus sign (+) next to **Locales**.
2. Select a locale from the drop-down list.
3. Click **Add**.

To add media files for this locale:

1. Click **New**.
2. Select a protocol from the drop-down list.
3. Enter a body to specify the location and name of the media file.

4. Repeat steps 1 through 3 to add URIs for additional media files.

Updating the Setup for a Media Resource

To view the setup for a media resource:

1. Select the change set. See ["Locating and Viewing Details for a Change Set"](#).
2. On the page for the change set, click **All Resources**.
3. In the displayed selection, click **Media Resource**.

The page displays the set of available media resources.

4. Click the entry for the resource.

A dialog box opens.

5. Update the data in the dialog box. See ["About the Media Resource Data"](#).
6. Click **Save**.

Working with Media Servers

OCECAS does not deliver a media server. However, it organizes the media servers you configure into groups.

About the Sample Media Server

In order to enable media resources to be played to users, you need to integrate the media servers you want to use with OCECAS. To do so, determine the appropriate driver string and URI from your media server vendor. OCECAS provides a sample media server named VoLTE with a sample configuration consisting of Jsr309TestDriver as the driver and a sample environment with URI configured as file:test/driver. It uses the following context map to specify the location:

```
domain_home/MediaServer/Production/VoLTE
```

where, *domain_home* specifies the runtime domain in your installation.

About the Media Server Data

[Table 3–4](#) lists the data required to define a media server.

Table 3–4 Media Server Data

Field	Entry	Description
Details	Name	A required field. Enter a unique name.
Details	Description	Enter a description.
Driver	Interface type	Select: <ul style="list-style-type: none"> ▪ name: to enter the driver name. ▪ jndi: to enter Java Naming and Directory Interface (JNDI).

Table 3–4 (Cont.) Media Server Data

Field	Entry	Description
Environments	All (Default)	<p>The list of environments specified for this server.</p> <p>To add an environment:</p> <ol style="list-style-type: none"> 1. Click New. 2. Enter the name of the environment. For example, Default, Testing, Staging, Production. 3. For the media server URI: <p>Select the type of media server from file://, http://, https://, data:, and other:.</p> <p>In the adjoining field enter the path name to the location of the media server file.</p> <p>Use the trash can icon to delete a row entry.</p>
Parameters	Collect	<p>To add a Collect parameter:</p> <ol style="list-style-type: none"> 1. Click New. 2. In the Name field, enter the name of the parameter. 3. In the Data Type field, select from String, Integer, Float, Boolean, and Date. <p>Use the trash can icon to delete a row entry.</p> <p>OCECAS provides the following Collect parameters:</p> <ul style="list-style-type: none"> ■ Max Duration: Integer ■ Initial Digit Timeout: Integer ■ Buffer Digit Collection: Boolean ■ Barge In: Boolean

Table 3–4 (Cont.) Media Server Data

Field	Entry	Description
Parameters	Play	<p>To add a Play parameter:</p> <ol style="list-style-type: none"> 1. Click New. 2. In the Name field, enter the name of the parameter. 3. In the Data Type field, select from String, Integer, Float, Boolean, and Date. <p>Use the trash can icon to delete a row entry.</p> <p>OCECAS provides the following Play parameters:</p> <ul style="list-style-type: none"> ■ Max Duration: Integer ■ Repeat Count: Integer ■ Repeat Interval: Integer ■ Queue Announcement: Boolean
Parameters	Record	<p>To add a Record parameter:</p> <ol style="list-style-type: none"> 1. Click New. 2. In the Name field, enter the name of the parameter. 3. In the Data Type field, select from String, Integer, Float, Boolean, and Date. <p>Use the trash can icon to delete a row entry.</p> <p>OCECAS provides the following Record parameters:</p> <ul style="list-style-type: none"> ■ Max Duration: Integer ■ Min Duration: Integer ■ Append Recording: Boolean ■ Start Beep: Boolean ■ Start in Paused Mode: Boolean ■ Stop on Silence: Boolean ■ Initial Timeout: Integer ■ Final Timeout: Integer
Parameters	Conference	<p>To add a Conference parameter:</p> <ol style="list-style-type: none"> 1. Click New. 2. In the Name field, enter the name of the parameter. 3. In the Data Type field, select from String, Integer, Float, Boolean, and Date. <p>Use the trash can icon to delete a row entry.</p> <p>OCECAS provides the following Conference parameters:</p> <ul style="list-style-type: none"> ■ Max Ports: Integer ■ Max Active Inputs: Integer

Adding a Media Server to a Change Set

To add a media server to a change set:

1. Select the change set. See "[Locating and Viewing Details for a Change Set](#)".
2. On the page for the change set, click **All Resources**.
3. In the displayed selection, click **Media Server**.

The page displays the set of default locales.

4. Click **New**.
A dialog box opens.
5. In the dialog box configure the media server data. See "[About the Media Server Data](#)".
6. Click **Save**.

Updating the Setup for a Media Server

To view the setup of a media server for a change set:

1. Select the change set. See "[Locating and Viewing Details for a Change Set](#)".
2. On the page for the change set, click **All Resources**.
3. In the displayed selection, click **Media Server**.
The page displays the set of default locales.
4. In the **Name** column, locate the entry for the resource. Click the entry.
A dialog box opens.
5. In the dialog box update the media server configuration. See "[About the Media Server Data](#)".
6. Click **Save**.

Working with Prefix Trees

Prefix trees in OCECAS contain mappings between global prefixes and country codes.

About the Default Prefix Tree

OCECAS provides a default prefix tree, **CountryCodePrefixTree**. This tree is located at **/PrefixTree/Global/CountryCodePrefixTree**. Each branch in this tree consists of a three-letter country code and its dialing code prefix. For example, ALB: 355 represents the country calling code for Albania.

Prefix tree data is stored in the **sd_service_data** table in the management domain (the Session Design Center domain).

A prefix can include multiple countries. For example, the prefix 1 includes the USA, Canada, Puerto Rico, and the Dominican Republic. **CountryCodePrefixTree** displays the following for Puerto Rico.

- **PRI** : No prefix
 - **Eastern** : 1787
 - **Western** : 1939

In this case, the prefix 1 is built into the full prefixes for the children **Eastern** and **Western**.

About the Prefix Tree Data

[Table 3–5](#) lists the data required to define a prefix tree.

Table 3–5 Prefix Tree Data

Entry	Description
Name	Required. Enter a unique name for the prefix tree.
Key	Enter the key. The default entry is /PrefixTree/Global . This is the service data key. An entry with the same key value is also stored in the sd_c_external_concept table and is accessible when you reference the prefix tree in an Activity.
Description	Enter a description for the prefix tree.
Prefix Tree	The prefix tree with its branches. You can add branches to the prefix tree. See " Adding to a Prefix Tree ".

Creating a Custom Prefix Tree

To create a custom prefix tree for a change set:

1. Select the change set. See "[Locating and Viewing Details for a Change Set](#)".
2. On the page for the change set, click **All Resources**.
3. In the Numbers category, click **Prefix Trees**.
The page displays the defined prefix trees.
4. Click **New**.
A dialog box opens.
5. In the dialog box enter the prefix tree information as described in "[About the Prefix Tree Data](#)".
6. Click **Save**.

Adding to a Prefix Tree

You can create a new branch for a prefix tree and you can append a new child to a branch.

To append a new branch to prefix tree:

1. Select the change set. See "[Locating and Viewing Details for a Change Set](#)".
2. On the page for the change set, click **All Resources**.
3. In the Numbers category, click **Prefix Trees**.
The page displays the defined prefix trees.
4. To access a specific prefix tree, click on its name under **Name**.
5. Click the + sign next to the Prefix Tree icon at the top of the tree.

Tip: You must move the cursor over the Prefix Tree icon for + and - signs to appear.

The Append a New Branch dialog box opens.

- a. In the **Branch Name** field, enter the name for the new branch.
- b. Do one of the following:

Leave **Prefix Digits** empty to create a branch with no prefix. Do this if you want to add children with unique new prefixes to the new branch.

Or

Enter a prefix for the branch in the **Prefix Digits** field.

c. Click **Append**.

6. Click **Save**.

To append a new child to an existing prefix branch:

1. Select the change set. See "[Locating and Viewing Details for a Change Set](#)".

2. On the page for the change set, click **All Resources**.

3. In the Numbers category, click **Prefix Trees**.

The page displays the defined prefix trees.

4. To access a specific prefix tree, click on its name under **Name**.

5. In the prefix tree, locate the branch to which you want to append a child.

6. Click the + sign next to the branch.

Tip: You must move the cursor over the branch for + and - signs to appear.

The Append a New Branch dialog box opens.

a. In the **Branch Name** field, enter the name for the new child branch.

b. Do one of the following:

Leave **Prefix Digits** empty to create a branch with no prefix. Do this if you want to add one or more children with unique prefixes to the new branch.

Or

Enter a prefix for the branch in the **Prefix Digits** field.

Note: If the parent has a prefix, the child prefix is appended. For example, if the parent has a prefix of 3 and you enter 16, the prefix for the child will be 316.

c. Click **Append**.

7. Click **Save**.

Removing a Branch from the Prefix Tree

You can remove a branch from a prefix tree. When you do so, OCECAS removes all the children of that branch.

To remove a branch and all of its children from a prefix tree:

1. Select the change set. See "[Locating and Viewing Details for a Change Set](#)".

2. On the page for the change set, click **All Resources**.

3. In the displayed selection, click **Prefix Trees**.

The page displays the defined prefix trees.

4. To access a specific prefix tree, click on its name under **Name**.

5. In the prefix tree, locate the branch that you want to remove.

6. Click the - icon next to the branch.

Tip: You must move the cursor over the branch for + and - signs to appear.

A dialog box opens to confirm that you want to remove the specified branch.

7. Click **Remove**.

Working with Templates

OCECAS supports the use of templates to build the message bodies of outgoing messages. All Activities associated with an outgoing message reference a message template. You can create templates to generate the message structure, and use the Activities in the control flow to determine when messages are sent.

A web service is treated as template because a web service location is closely tied to the message content. The template, therefore, defines the message body, but also aspects of the message destination.

Message templates are stored as service data. OCECAS groups templates in logical sets in an hierarchy, making them easy to locate. Each template describes a single message. The data includes the user-friendly name given to the template, the parent template set, the message body determined by the locale, and the message type such as plain text, or XML.

You can configure an Activity by update the templates it uses at runtime. The variable parts inserted at runtime are identified by name and type, such as string and a description. A runtime check verifies the content of the message body. Use the control flow editor to create them at the point of use.

About the Default Notification Template

OCECAS provides a default notification template called **AccessTransferInformation**. [Example 3-1](#) shows the template definition.

Example 3-1 Default Notification Template Syntax

```
<?xml version="1.0" encoding="UTF-8"?>
<SRVCC-infos>
<SRVCC-info ATCF-Path-URI="{ATCFPathUri}">
<ATU-STI>{AtuSti}</ATU-STI>
<C-MSISDN>{CMsisdn}</C-MSISDN>
</SRVCC-info>
</SRVCC-infos>
```

Here:

- ATCF-Path-URI is the ATCF path URI
- ATU-STI is the address of the SCC application server
- C-MSISDN is the correlation Mobile Station International Subscriber Directory Number (MSISDN) for the sessions associated with this user.

In runtime, the access transfer information sent from an example SRVCC application server to the access transfer control function (ATCF) contains the following:

Example 3-2 RunTime Example Notification Data

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

```

<SRVCC-info ATCF-Path-URI="sip:termsdggfdwe@actf.visited2.net">
  <ATU-STI>sip:sccas1.home1.net</ATU-STI>
  <C-MSISDN>tel:+1-237-555-1111</C-MSISDN>
</SRVCC-info>
</SRVCC-Infos>

```

Working with Notification Templates

All Activities associated with an outgoing message reference a message template. You can create a Notification template through the All Resources menu

About the Notification Template Data

Table 3–6 lists the data required to define a notification.

Table 3–6 Notification Template Data

Field	Entry	Data
Details	Name	A required field. Enter a unique name.
Details	Description	Enter a description.
Parameters	Name	Each notification definition may have zero or more parameters that will be used by a control flow to populate a locale specific template Enter a unique name.
Parameters	Data Type	Select the data type for this resource from String , Integer , Float , Boolean , Date , MapArray , StringArray , Integer Array , and FloatArray .
Parameters	Description	Enter a description for the parameter.
Locales	Locales that the resource supports	The list of locales. The first entry in the list is Default . Each locale is configured with a template that is used by a control flow to generate the contents of an end user notification. Click on a locale listed under Locales . Update the template definition. To add another locale: <ol style="list-style-type: none"> 1. Click +. 2. Select another locale from the displayed list. 3. Click Add. 4. Repeat, to add another locale. 5. Click Save.

Adding a Notification Template

To add a notification template to a change set:

1. Select the change set. See "[Locating and Viewing Details for a Change Set](#)".
2. On the page for the change set, click **All Resources**.
3. In the displayed selection, click **Notifications**.
The page displays the set of notification templates.
4. Click **New**.
A dialog box opens.
5. In the dialog box configure the notification template data. See "[About the Notification Template Data](#)".

6. Click **Save**.

Updating a Notification Template

To update a notification template to a change set:

1. Select the change set. See "[Locating and Viewing Details for a Change Set](#)".
2. On the page for the change set, click **All Resources**.
3. In the displayed selection, click **Notifications**.
The page displays the set of notification templates.
4. Click on the notification template.
A dialog box opens.
5. In the dialog box update the notification template data. See "[About the Notification Template Data](#)".
6. Click **Save**.

Working with Web Services Templates

The need to query external data sources using standard methods of exchanging information, such as SOAP or JSON, is fairly common. For example, you might need to affect session flow by querying for information on another system regarding products or subscribers. You might, for example, have a service that requires you to securely authenticate a credit card payment. Or you might have a location server that you can query to find out where a call originated and you want to use that to determine the location where the call should be terminated.

By creating web service templates, you can access services that are implemented on external servers and map data to and from a session. When you successfully create a web service template, it becomes available as a selection for an Activity that requires a web service template, such as the Run Web Service Activity.

You can create a Web Service template either by creating the template manually or by importing a WSDL file.

Creating a Web Service Template Manually

To create a Web Service template manually:

1. Select the change set. See "[Locating and Viewing Details for a Change Set](#)".
2. Click the **All Resources** tab.
3. On the **All Resources** menu, click **Web Services**.
4. On the Web Services templates page, click **New** to create a new template.
A dialog displays with links for three pages: Details, Parameters, and Template.
5. On the first page, Details, enter a name for the template and, optionally, a description.
6. Click **Parameters** to enter any template parameters.
These parameters specify the variable parts to be completed when the Run Web Service Activity is selected and configured in a control flow.
7. Click **New** and then enter a name, data type, and description for the parameter.
For example:

- Name: key
 - Data Type: String
 - Description: The value to search for
8. Repeat step 7 for each template parameter that you need to enter.

Note: To delete a parameter, click the trash-can icon next to that row.

9. Click **Template** to enter the web service template. For example:

```
<myService>
  <key xsi:type='xsd:string'>{$key}</key>
  <text xsi:type='xsd:string'>{$text}</text>
</myService>
```

10. Click **Save** to save the template.

Note: You can click **Save** at any time while creating a template to save what you have defined.

You can also create a Web Service template by importing a WSDL file to the server.

Importing a WSDL File

To import a WSDL file to create a Web Service template:

1. Select the change set. See "[Locating and Viewing Details for a Change Set](#)".
2. Click the **All Resources** tab.
3. On the All Resources menu, click **Import WSDL**.
4. In your file system, locate the **.wsdl** file that you want to import and click **Import WSDL**.

The name of the imported WSDL file, along with its size, status and time of day are listed on the page.

The names of the services made available are displayed under the **WEB SERVICES** category in the Changes panel.

Working with Change Sets

This chapter describes how to create, manage, and deploy change sets in Oracle Communications Evolved Communications Application Server (OCECAS).

See “About Control Flows” and “About Change Management” in *Evolved Communications Application Server Concepts* for an overview of change sets and the change management process.

Managing Change Sets

Change sets are collections of configuration and control flow changes. You create them using the **Open Projects** section of the Session Design Center home page. Change sets are also known as *projects*.

To activate a change set, you first deploy it to the **Testing** environment for testing and make any changes required. Once satisfied that it does what you intend, you then deploy the change set to the **Staging** and **Production** environments. The change set can not be modified after you deploy it to **Staging** or **Production**. If there are errors in a change set deployed in the **Staging** or **Production** environments, you can retract it, create a new corrected change set, and deploy the corrected change set in its place.

When you create a change set, you specify a baseline change set to which you are making the changes. The original baseline change set for OCECAS is the **VoLTE, VoWiFi and eSRVCC** change set, which consists of the VoLTE and VoWiFi application. As you continue to create change sets over time, you correspondingly increase the number of potential baseline change sets. The baseline change set is referred to as the *parent* and the new change set is the *child*.

This chapter assumes that you have an OCECAS Session Design Center GUI running.

Creating a Change Set

To create a change set:

1. On the **Home** page of Session Design Center, click **View All Projects**.

2. Click **New**.

The New Change Set dialog box displays.

3. In the **Name** field, enter a unique name.
4. In the **Description** field, enter a short description.
5. To provide a base line for the change set, click the list for **Baseline**. Select a base line.
6. Click **Create**.

Session Design Center creates a change set ID for the new change set and displays it and the name of the change set at the top of the page.

To add a control flow to the change set, or to modify a control flow, see ["Working with Control Flows"](#).

To deploy the change set to the production pipeline, see ["Deploying Change Sets"](#).

Locating and Viewing Details for a Change Set

To locate a change set, do one of the following:

- If the change set is not yet deployed to a runtime environment, Session Design Center displays the change set below **Open Projects**.
- If the change set is not among the recent entries:

Click **View All Projects**.

To search for a change set, enter its ID in the **Search** field, or start entering a string. As you type, the string filters the display to show only the change sets with a matching string in either the name or the description

You also filter the display using the **Filter:All/Open/Deployed** buttons.

Click the change set.

If a change set is in **Open Projects**, or deployed to the **Testing** environment, a pencil icon displayed to the right side of a change set indicating that the change can be edited. An eye icon indicates that the change set is deployed and is available only to view.

Click a change set row to view its details. The row expands to display any description of the change set, the date and time the change set was created, and the name of the baseline change set. Click **Edit Details** to change this information.

Modifying a Change Set

You can modify a change set if its deployment status is **open** or it is deployed to **Testing**, and it has not yet been deployed to staging or production. Some of the ways in which you can modify a change set are:

- Modifying the baseline for a change set. See ["Modifying the Baseline for a Change Set"](#).
- Modifying a control flow in the change set. See ["Modifying a Control Flow"](#).
- Adding a control flow to the change set. See ["Creating a Control Flow"](#).
- Removing a control flow from the change set. See ["Reverting Changes in a Control Flow"](#).

You can also modify a change set in other ways, such as adding media resources, media servers, and notification templates.

Modifying the Baseline for a Change Set

To modify the baseline for a change set:

1. On the **Home** page, click **View All Projects**.
2. Locate the change set. See ["Locating and Viewing Details for a Change Set"](#).
3. Click the arrow to the left of the change set name to expand the details.

4. Click **Edit Details**.
5. For the **Baseline** field, select a different entry.
6. Click **Save**.

Viewing the Changes to a Change Set

To view the changes to a change set:

1. On the **Home** page, click **View All Projects**.
2. Locate the change set. See "[Locating and Viewing Details for a Change Set](#)".
3. Click on the icon in the last column of the row containing the change set.

Tip: A view icon is displayed for a deployed change set and an edit icon for an open change set.

Bundling Change Sets

The ability to bundle change sets allows you to combine change sets and deploy them as a single unit. Bundling makes deployment and rollback of a group of change sets as easy as deploying and rolling back a single change set.

You might want to create a bundle because a project has resulted in the creation of multiple related change sets that you want to deploy as a unit. Another reason could occur when a deployed change set has been retracted from the staging or production environment due to an error and a new change set has been created to fix the error. If this were to happen multiple times, you potentially could have multiple change sets that fix the problem.

To create a bundle:

1. On the Home page, click **View Environment Configuration**.
2. In the left-hand panel, under Open Projects, click **New**.
3. From the New menu, select **Change Set Bundle**.

A New Change Set Bundle dialog opens.

4. In the New Change Set Bundle dialog, enter a **Name** and a **Description** for the new change set bundle.
5. Click **Create** to create the new change set bundle.

The new change-set bundle is added to the list under Open Project Bundles.

Note: A change set bundle does not have a baseline.

Adding Change Sets to a Bundle

To add a change set to bundle, drag the change set from the Open Projects section and drop it on the target bundle. The member change sets in a bundle appear in a list below the name of the bundle.

The following rules apply when adding a change set to a bundle:

- You can only add an open change set to a bundle. You cannot add a change set that has been deployed to the staging or production environments. To add a

change set from the testing environment, you must first revert it and then remove it, which places it back in the Open Projects panel.

- You cannot add a change set whose parent change set is not part of the bundle. You must first add the parent change set and then the children.

For example, for change sets A, B, C, and D, assume that A has the last deployed change set as a parent and that B has A as a parent, C has B as a parent, and D has C as a parent. If you create a bundle and add change sets in the order of A, B, D, you will get an error because D's parent (C) is not part of the bundle. The error message is: "Cannot add change set to bundle. Change set's baseline is not part of the bundle."

- The same rules that apply to non-bundled change sets also apply to bundled change sets.

Working with a Change Set Bundle

After you create a change set bundle, you can deploy it, retract it, and edit it just as you would a change set.

Errors When Deploying a Change Set Bundle

When you deploy a change set, errors appear in two ways.

In the first case, a dialog displays when you attempt to deploy the change set bundle. If the deployment fails, an error displays at the bottom of the dialog and the bundle and all change sets that it contains are not deployed. You must fix the error and deploy the bundle again.

In the second case, the bundle deploys but the heading of the bundle contains text that says "Contains Conflicts." Expand the bundle to show the change sets that it contains and conflict links in red text. Clicking the change set displays a table that shows the details of the conflicts found.

If a Change Set contains conflicts, it should be corrected and re-deployed.

Copying Change Sets

Copying a change set enables you to make changes to a change set that has been deployed to the staging or production environments. A change set in the staging or production environment cannot be changed even if it has been retracted. When you make a copy of a change set, however, you can make changes to the copy, perhaps to fix a problem that existed in the original.

In the testing environment, you can copy a change set after you revert it.

You can copy a change set either in the Projects section of the Session Design Center or in the Environments section.


The following rules apply when copying a change set:

- You cannot copy a change set bundle.
- When copying to an existing change set, the destination change set cannot have the same changes as the source.

To Copy a Change Set in Projects

To copy a change set in Projects.

1. On the Home page, click **View All Projects**.

2. In the list of change sets, locate the change set that you want to copy.
3. Click the copy icon () to the right of the Last Updated date and time.
4. Click the radio button next to one of these two options:
 - **Create a Change Set as Target Named:**
 - **Select an Existing Change Set as Target:**
5. Enter a target name if you are creating a change set or select a target name if you are copying to an existing change set.
6. Click **Copy**.

To Copy a Change Set in Environments

To copy a change set in Environments:

1. Click **View Environment Configuration**.
2. Locate the change set that you want to copy in the testing, staging, or production environment.
3. Click the drop-down menu icon for the change set.

Note: Only change sets that are available for copying display the drop-down menu icon.

4. Select **Copy** from the drop-down menu.
5. Click the radio button next to one of these two options:
 - **Create a Change Set as Target Named:**
 - **Select an Existing Change Set as Target:**
6. Enter a target name if you are creating a change set or select a target name if you are copying to an existing change set.
7. Click **Copy**.

Viewing Copy Progress

To view the progress of the copy, open the change set Changes panel. See "[Viewing the Changes to a Change Set](#)" for more information.

When a copy operation is in progress, a progress bar displays at the top of the Changes panel. As the copy progresses, the bar goes from left to right and the list of changes beneath it grows longer as more items are copied into the change set.

When the copy succeeds, the progress bar goes to 100 percent and then disappears.

Errors When Copying

If you are copying to an existing change set, you can receive an error for the following reasons:

- There are no differences between the source and destination change sets.
- A conflict exists such as a control flow or resource has been changed in both change sets.

If a copy operation fails, the progress bar in the Changes panel stops at a value less than 100 percent and an error displays below the progress bar. The error is typically:

- Copy incomplete, caused by transaction timing out.

Deploying Change Sets

The **View Environment Configuration** link in the **Environments** section of the landing page displays the runtime environments configured for your OCECAS installation.

About the Deploying to the Production Pipeline

For an overview of the OCECAS pipeline, see "About Change Management and the Production Pipeline" in *Evolved Communications Application Server Concepts*.

The production pipeline includes the runtime environments in this order:

1. Testing
2. Staging
3. Production

Change sets become active in an environment when they are deployed in the environment. [Table 4-1](#) shows the possible actions you can take on change sets in a runtime environment.

Table 4-1 Actions Available in the Environments

Action	Testing	Staging	Production
Edit	Available, if it has not been deployed. Retracted change sets are also not available for editing.	Not Available	Not Available
Deploy	Available. You can deploy a change set to the staging environment.	Available. You can deploy a change set to the production environment.	Not Available
Retract	Available. A change set can be retracted from the testing environment.	Available. A change set can be retracted from the staging environment.	Available. A change set can be retracted from the production environment.
Redeploy	Available. A retracted change set can be redeployed into the testing environment.	Available. A retracted change set can be redeployed into the staging environment.	Available. A retracted change set can be redeployed into the production environment.
Remove	Available, if it has not been deployed.	Not available	Not available

Deploying a Change Set

This section assumes that you have a Session Design Center GUI up and running with a change set created but not deployed. See "[Creating a Change Set](#)" for details.

Before you can deploy a change set, review it to ensure that the change set compiles correctly. The **Changes** panel for the change set should not display any error messages.

When you deploy a change set, you can choose to deploy it immediately or specify a later date and time. Once a change set is deployed to the staging or production environment the contents cannot be edited. See "[Correcting Problems in Deployed](#)

[Change Sets](#)".

Deploying a Change Set to the Testing Environment

To deploy a change set to the **Testing** environment:

1. In the **Home** page, click on **Environment Configurations**.
The **Environments** page displays the **Open Projects** panel, the **Testing**, **Staging** and **Production** environment entries.
2. In the **Open Projects** panel, locate the change set.
3. Place your cursor on the change set. The arrow changes to a cross to indicate you have selected it.
4. Select and drag the change set to the **Staging** environment. Release the cursor.
The **Deploying CS-** dialog box opens.
5. Specify when to activate the change set. Select one of:
 - **Activate immediately**
 - **Schedule for activation on**
Enter the date as *YYYY-MM-DD*. use the up and down arrows to provide the hour and minute as *HH:SS*.
6. Click **Confirm**.

When the change set becomes active it is automatically checked for compile errors. It must be error-free before you deploy it to the **Staging** environment.

Deploying a Change Set to the Staging or Production Environment

Before you deploy a change set to staging or production environments, test and confirm that it performs correctly. Once deployed to the staging or production environments you can not edit the change set, only retract it.

To deploy a change set into the next runtime environment in a pipeline:

1. In the **Home** page, click **View Environment Configurations**.
The **Environments** page displays the **Open Projects** panel, and the **Testing**, **Staging** and **Production** environments.
2. Select the change set to deploy.
3. Click on the arrow on the right side of the change set.
4. Do one of the following.
 - If you are in the testing environment, select **Deploy to Staging**.
 - If you are in the staging environment, select **Deploy to Production**.
 The **Deploying CS-** dialog box opens.
5. Specify when to activate the change set. Select one of:
 - **Activate immediately**
 - **Schedule for activation on**
Enter the date as *YYYY-MM-DD*. use the up and down arrows to provide the hour and minute as *HH:SS*.
6. Click **Confirm**.

Correcting Problems in Deployed Change Sets

This section explains how to correct a changes set if it is not working as expected. You have these choices for correcting change sets:

- For minor issues, create a new change set that fixes the problem, then retract the bad change set and deploy the new one. See "[Correcting a Change Set Without Stopping Service](#)".
- If you want to disable the service immediately, *retract* the bad change set first, and then replace it with a corrected version. See "[Correcting a Change Set by Removing the Service Immediately](#)".

A parent change set must be deployed in the testing, staging or production environment in order for the child change set to be deployed in that environment.

Correcting a Change Set Without Stopping Service

You can fix minor change set problems in the background without deactivating that change set. This method applies to issues that are not causing major problems.

To fix a problem without disrupting service:

1. In **Open Projects**, create a new change to replace the one with the problem. See "[Modifying a Change Set](#)".
2. Retract the change set with the problem.
3. Deploy the corrected change set into the staging environment, and then into the production environment. See "[Deploying a Change Set](#)".

Correcting a Change Set by Removing the Service Immediately

To fix a problems that must be resolved by removing the service immediately, retract the change set from the production environment. Then create a replacement change set and deploy it. The idea is to have a replacement ready, or nearly in time to maintain quality of service.

To disable a change set immediately and replace it:

1. From the **Home** page, click **View Environment Configuration**.
1. The **Environments** page displays the **Open Projects** panel with the **Testing**, **Staging**, and **Production** entries.
1. Retract the change set. See "[Retracting a Change Set](#)" for details.
2. In **Open Projects**, create a new change to replace the one with the problem. See "[Modifying a Change Set](#)".
3. Test the change set.
4. Deploy the corrected change set to the staging, and then production environments. See "[Deploying a Change Set](#)".

Retracting a Change Set

To retract a change set from the appropriate runtime environment:

1. From the **Home** page, click **View Environment Configuration**.
The **Environments** page displays the **Open Projects** panel with the **Testing**, **Staging**, and **Production** entries.
2. Locate the change set to retract. See "[Locating and Viewing Details for a Change Set](#)".

3. Click the arrow on the right side of the change set.
4. Select **Retract**.

The change set is no longer active in that environment. OCECAS displays the retraction date and time.

Redeploying a Change Set

If you decide that a retracted change set should be returned to service, you can redeploy it. Redeploying uses the same mechanism as deploying.

Note: before you redeploy a change set, verify that the base line for such a change set is one that is deployed in that environment and does not have issues.

To redeploy a change set into an environment:

1. In the **Home** page, click on **Environment Configurations**.
The Environments page displays the **Open Projects** panel, the **Testing, Staging** and **Production** environment entries.
2. Select the change set to redeploy.
3. Click on the arrow to the right of the change set.
4. Select **Redeploy**.

Working with Subscriber Data

This chapter explains how Oracle Communications Evolved Communications Application Server (OCECAS) stores subscriber information and how applications retrieve that data. It includes a description of the default subscriber data store configuration, and explains how you can customize it for your implementation.

See “About Managing Subscribers and Subscriber Data” for an overview of how OCECAS stores subscriber data records.

About Subscriber Data

OCECAS subscriber data can be stored in multiple sources, called data stores, which are accessed through a configurable *view*. The view retrieves the needed data from the data stores and federates it into a single record to return to the requesting application. The application is not aware of the actual data storage, only the format of the data returned. The view, on the other hand, is aware of the data storage location and format, the format that the application expects, and the mapping between the two.

OCECAS data stores can include an external Home Subscriber Server (HSS), the default ECAS Subscriber Server (ESS) NoSQL database, which is optional, or a data store from an external provider.

Operator’s often will want to integrate an existing HSS. The OCECAS view framework provides the ability to obtain subscriber data from an HSS through the Diameter Sh interface.

Types of Subscriber Data

The types of data accessed from the OCECAS data stores include the following:

- Subscriber profile and services data
- Supplementary services data
- SRVCC data

The following sections describe how these types of data are accessed.

Accessing Subscriber Profile and Services Data

You can store subscriber profile data in any of the supported OCECAS data stores.

When stored in the HSS, the profile and services data are split into separate blocks of transparent data, each with its own service-indication value. Transparent data is data for which the HSS understands the syntax but not the meaning. [Table 5-1](#) describes the service indication values of the profile and services data:

Table 5–1 Service Indication Values for Profile and Services Data

Service Indication	JSON Block	Description
ECAS_USER_PROFILE_1	Profile	Subscriber profile data
ECAS_USER_SERVICES	Services	Subscriber services
ECAS_USER_MESSAGE_ACCOUNT_1	MessageAccount	Transparent data used for Message Waiting Indication Services

In both the ESS and the HSS, each data block is stored as a JSON document. In the case of the ESS NoSQL database, they are stored as a single document; in the HSS they are stored as separate documents.

The following example shows base subscriber data in an ESS NoSQL database:

```
{
  "Description": "IMS Subscriber",
  "ServiceProviders": {
    "MVNO1": {
      "profile_data": {
        "RegistrationExpiry": 0,
        "Language": "en_GB",
        "Translations": {}
      },
      "services": {
        "REGISTRATION": "VoLTE and VoWiFi",
        "VOICE": "VoLTE and VoWiFi"
      }
    }
  }
}
```

Accessing Supplementary Services Data

Supplementary services data can be stored in the HSS or as part of the subscriber profile in the ESS NoSQL database or third-party data store. When it is stored in the HSS, it is stored as a separate RepositoryData entry with one of the service-indication values shown in [Table 5–2](#).

Table 5–2 Service Indication Values for Supplementary Services

Service-Indication Value	Description
MMTEL-SERVICES-BINARY-1	Supplementary services are stored in binary format
MMTEL-Services	Supplementary services are stored in XML format

The data returned from the HSS is converted into JSON and merged into the subscriber profile document that is returned to the application.

Accessing SRVCC Data

OCECAS generally accesses Single Radio Voice Call Continuity (SRVCC) data from the subscriber data store to support the following features:

- To obtain the IMS Voice Over Packet-Switched-Session Support indicator, which indicates whether the subscriber is currently waiting on an IMS Voice over packet-switched-capable access.
- To obtain or update the Session Transfer Number.

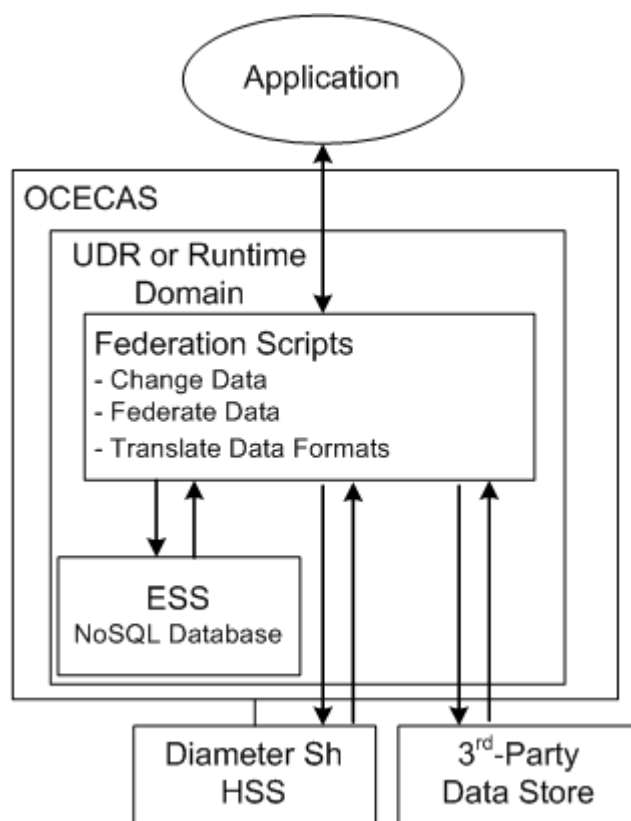
OCECAS retrieves the following non-transparent SRVCC-related fields from the HSS and merges them into the subscriber profile document:

- TADS information
- UE SRVCC capability
- CSRN
- STN SR

Federating the View

Figure 5–1 shows how OCECAS federates data from multiple data stores in a single view.

Figure 5–1 OCECAS Data Stores



If you use the default ESS inside an OCECAS runtime domain, the views and their Groovy federation scripts are loaded each time you start the runtime domain. If you locate the ESS inside a dedicated UDR domain, the views and their federation scripts are loaded each time you start the runtime domain.

If necessary, you can change the federation instructions during runtime. See ["Updating Federation Script Behavior During Runtime"](#) for details.

The view federation script transforms the subscriber profile data, the services data, the supplementary services, and the SRVCC data into the following subscriber profile record. In an HSS, each block of subscriber data, which includes Profile, Services, and MMTelServices, represents a block of transparent data. In the default NoSQL data store, this structure is maintained.

```
{
  "Service":{
    "HomeCountry":"GBR",
    "HomeNetworkIdList":["network.gb.net","ims.mnc044.mcc234.3gppnetwork.org",
      "oracle.com","presence.oracle.com"],
    "ServiceProviderName":"MVN01",
    "ServiceName":"VoLTE and VoWiFi",
    "ControlFlowName":"Session Origination"
  },
  "Profile":{
    "CallCount":0,
    "CallFwdCompany": "",
    "Language":"en_GB",
    "ToNumber": "",
    "Translations": {},
    "VMRedirect": ""
  },
  "MMTel": {
    "OIP":{
      "Active":true
    },
    "TIP":{
      "Active":true
    },
    "OIR":{
      "active":true,
      "Default":"Restricted",
    },
    "TIR":{
      "active":true,
      "Default":"Restricted",
    },
    "ICB":{
      "Active":true,
      "Rules":[
        {
          "RuleDeactivated":false,
          "IdentityList":["sip:notallowed@oracle.com"],
          "DomainList":["notallowed.com"],
          "DomainExceptList":["domain1"],
          "Allow":false
        },
        {
          "RuleDeactivated":false,
          "Anonymous":true,
          "Allow":false
        }
      ]
    },
  },
  "OCB":{
    "Active":true,
    "Rules":[
      {
        "RuleDeactivated":false,
        "Roaming":true,
        "Allow":true
      },
      {
        "RuleDeactivated":false,
        "InternationalExHC":true,

```

```

        "Allow":false
      }
    ]
  },
  "MMTelOperator": {
    "OIR":{
      "Mode":"Permanent",
      "Restriction":"AssertedIdentity"
    },
    "TIR":{
      "Mode":"Permanent"
    }
  }
}

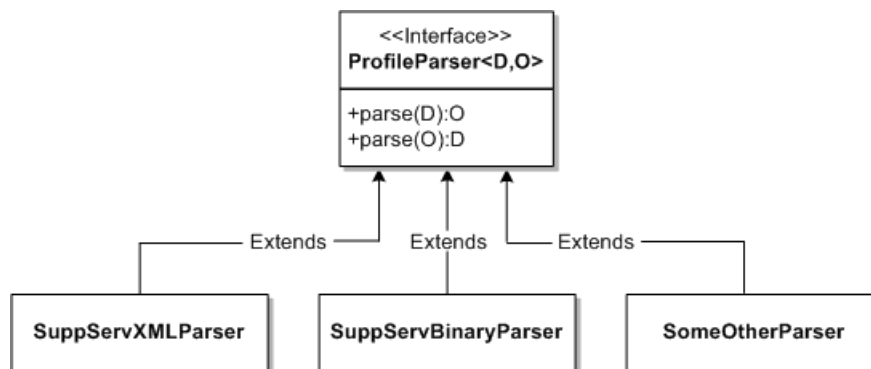
```

Translating HSS Data

Transparent data in the HSS is accessed by the view and federated into a single JSON document before returning it to the application. Each block of transparent data can be in a different format. The ECAS_USER_PROFILE_1 and ECAS_USER_SERVICES_1 blocks are stored in JSON format so the translation is straight-forward. MMTel-Services, however, can be stored in XML, binary, or some proprietary format. Consequently the view must be able to parse the data stored in the HSS and convert it into the JSON format that the application expects.

Adding a Plug-in Parser

To parse the non-JSON data stored in the HSS and convert it to JSON, and to convert it from JSON back to the HSS data types, the view allows you to add a plug-in parser. To create a parser, you must implement the Java ProfileParser interface in the OCECAS platform API. The following diagram illustrates the class structure:



Note: The plug-in parser can also be used elsewhere. For example, you could also use a parser for XML Supplementary Services data to convert between JSON data and XML for the XCAP interfaces.

You can add the parser to the view using the CSP configuration. The following example illustrates a configuration:

```

<csp-config>
  <name>csp</name>
  <pipes>

```

```
<udr-pipe>
  ...
  <parser>
    <name>SuppServXMLParser</name>

<parser-class>oracle.occas.csp.platform.parsers.SuppServXMLParser</parser-class>
  </parser>
  ...
</udr-pipe>
</pipes>
</csp-config>
```

The class is loaded by the `csp.xml` code and made accessible through the external concept, `UserProfileContext`. An access method in `UserProfileContext` allows the Groovy script to get the parser as shown here:

```
ProfileParser parser = context.getProfileParser("SuppServXMLParser")
ObjectNode contextNode = parser.parse(xmlObject)
```

You can add additional methods to load other parsers that are defined in the configuration.

Configuring OCECAS Data Stores

The procedures in this chapter assume that you have already installed OCECAS and your ESS NoSQL database, or third-party data store, and completed the following post-installation tasks, if applicable, in these sections of *Oracle Communications Evolved Communications Application Server Installation Guide*:

- Setting up OCECAS for Subscriber Data
- Configuring Communication Between Your Runtime Domains and the UDR
- Configuring Communication Between Your UDR Domain and the UDR

To complete the following tasks for your data store, see the associated sections of documentation:

- To access data from an HSS, use the instructions in "[Adding an HSS Subscriber Data Store](#)".
- To perform advanced configuration for your ESS NoSQL database, see "[Advanced Configuration of Your ESS NoSQL Subscriber Data Store](#)".
- To replace the default ESS with another data store, see "[Replacing the ESS NoSQL Database](#)" for instructions.
- To make changes to the ESS schema, see "[Changing the Default ESS NoSQL Schema](#)" for details.
- To update your federation scripts without shutting down the runtime environments, see "[Updating Federation Script Behavior During Runtime](#)".
- To provision the ESS with subscriber data using the RESTful interface, see "[Configuring Subscriber Data Quickly for a Test or Demonstration System](#)" and "[Provisioning the ESS with Subscriber Data](#)".
- To understand the details of the default schema for the ESS subscriber store, see "[The Default ESS Subscriber Store Schema](#)".

Adding an HSS Subscriber Data Store

Skip this section if you do not access subscriber data from an HSS.

If you access subscriber data from an existing HSS, use the following instructions to make it available to OCECAS.

Configuring OCECAS to Accept Data from a Diameter Sh HSS

You configure your Diameter Sh HSS using the OCECAS Administration Console to edit the `csp.xml` file in each runtime environment, as described in this section.

See “Managing the Diameter Node” in *Evolved Communications Application Server Administrator’s Guide* for instructions on how to configure the Diameter interface.

To configure a Diameter Sh HSS:

1. Open the OCECAS administration console.
For more information, see “About the OCECAS Administration Console” in *Evolved Communications Application Server Administrator’s Guide*.
2. Click **Lock & Edit**.
3. In the **Domain Structure** pane, navigate to `domain_name` then **Diameter**.
The **General** tab appears.
4. Fill in this tab to meet the needs of your implementation.
5. Click **Save**.
6. Navigate to **Applications** by clicking the **Applications** tab.
7. Click **New**.
8. Fill in the **Application Name**, **Class Name**, and any optional parameters for the new Diameter application.
9. Click **Finish**.
10. Finish configuring Diameter using the **Peers**, **Routes**, and **Message Debug** tabs as necessary for your implementation.
11. Click **Release Configuration**.
12. Open the `domain_home/config/custom/csp.xml` file for editing.
13. If needed, change the `data-source` parameters in the `<provider>` element to specify the HSS to use for read-request data source, read-response data source, update-request data source, and update response data source.
14. Edit the `script_HSS` federation script as necessary to manipulate the HSS data to satisfy the calling application:
 - Translate the HSS data into a format that the calling application requires.
 - Federate the data with any other data stores that your implementation uses.
15. Save and close `csp.xml`.

WARNING: After updating the `csp.xml` file, if you update the configuration through the WebLogic administration console, the changes made in `csp.xml` will be lost.

16. Restart the servers in the runtime environment.
17. Repeat this procedure for each runtime environment that uses the HSS.

Advanced Configuration of Your ESS NoSQL Subscriber Data Store

This section assumes that you have already followed the instructions in *Oracle Communications Evolved Communications Application Server Installation Guide* and installed your default NoSQL ESS and specified where it should connect to a runtime domain. This section takes up where that one left off.

This section requires that you know the format in which your calling applications require data, and know Groovy/Java to modify the federation scripts.

By default, the **csp.xml** file includes a default NoSQL ESS, defined as **udr1** of type **NoSQLProviderFactory**:

Also by default, an adapter is defined that specifies the ESS **nosql1** connection information:

```
<adapters>
<no-sql-adapter>
  <name>nosql1</name>
  <helper-host>localhost:5000</helper-host>
  <store-name>spr.data</store-name>
</no-sql-adapter>
</adapters>
```

You can use these default values, or change them to meet your needs.

To finish configuring your default NoSQL ESS:

1. In a runtime domain, open the *domain_home/config/custom/csp.xml* file for editing.
2. Optional. Edit the `<provider>` element for the ESS, specifying the type of traffic it will be used for. The default ESS **nosql1** is designated for all of the read, update, request, and response messages:

```
<name>udr1</name>
  <provider>
    <name>p1</name>
    <type>NoSQLProviderFactory</type>
    <adapter>nosql1</adapter>
    <read-request-data-source>source_initial</read-request-data-source>
    <read-response-data-source>source_initial</read-response-data-source>
    <update-request-data-source>source_initial</update-request-data-source>
    <update-response-data-source>source_initial</update-response-data-source>
  </provider>
```

Change this as necessary for your implementation.

3. Change the federation scripts as necessary to modify the data from the ESS so that the calling application can accept the data. The exact configuration will be unique to your implementation.
4. Save and close the **cspl.xml** file.
5. Restart the domain that contains the ESS.

Adding Additional Data Sources and Formats to the View

This section explains how to change the default subscriber store configuration.

Replacing the ESS NoSQL Database

The default OCECAS subscriber data storage paradigm retrieves data stored in an HSS that you specify, and if the data changes, it is stored in the ESS NoSQL database. You replace the ESS with a different database if your implementation requires it. However this requires a strong knowledge of the database you wish to use, and significant work to change the Groovy federation scripts in the `csp.xml` file.

See "[The Default ESS Subscriber Store Schema](#)" for details on the default ESS data schema.

Adding a New UDR Using a New Data Type

To add a new UDR provider using a completely different data type, edit the `csp.xml` file, adding a new provider and adapter to specify the UDR name and location and the types of messages that can access them. Then create a new view, data-source, and federation script to translate the data.

Using Different Types of Data from the Same Data Provider

You can obtain different data types from a single data store. In a single pipeline, define one provider and adapter to reference the data store. Then define different view/data source/federation scripts for each of the data types.

Changing the Default ESS NoSQL Schema

You can modify some part of the OCECAS default ESS schema if your implementation requires it. However you have these restrictions:

- Some properties can not be changed because they are specified in the UDR MBean. These include the top level property names:
 - `userIdentifier`
 - `mapData`
 - `JsonData`
- You must correct any references to properties that you change in the data federation scripts in the `csp.xml` file. After correcting this file, restart the domain administration servers that use the `csp.xml` file to make your changes take effect.

Changing the Default ESS Schema

To change the default UDR Schema for a production pipeline:

1. Create a new change set in the test domain for changing the `UdrSubscriberSchema` and record the new change set ID.
2. Use the `SchemaResource` class operations in the RESTful API to change `UdrSubscriberSchema` for the new change set:

```
/api/change-sets/changeset_ID/schemas/UdrSubscriberSchema
```

Where `changeset_ID` is the ID of the change set you created in Step 1.

See the **SchemaResource** class in *Evolved Communications Application Server UDR RESTful API Reference* for details on the RESTful APIs.

3. Open the `domain_home/config/custom/csp.xml` file for editing.
4. Search the Groovy data federation script in `csp.xml` for any of the properties that you changed in the **UdrSubscriberSchema** table, and replace them with the new values you created.
5. Save and close the file.
6. Restart the domain servers.
7. Propagate the changes you made to the staging and production environments in your pipeline.

Updating Federation Script Behavior During Runtime

You can change the federation script behavior at runtime by using this WebLogic Scripting (WLST) command from the default script location in `domain_name/bin`:

```
java weblogic.WLST ./csp.py -u admin_username addsrc federation_scriptname  
federation_script_filename
```

Where:

- `admin_username` is a WebLogic user with administration privileges.
- `federation_scriptname` is the name of the federation script to change. The default is **script_initialProfile**.
- `federation_script_filename` is the replacement script with the instructions you want to use.

For details on WLST, see “Using the WebLogic Scripting Tool” in *Oracle Fusion Middleware WebLogic Scripting Tool*.

About the OCECAS Diameter Sh Interface

See “3GPP_TS_29.328” in *Evolved Communications Application Server Compliance Guide* for a list of the supported Diameter Sh messages. If your implementation uses custom AVPs, add them to the Groovy scripts in `csp.xml`. See ["Configuring OCECAS to Accept Data from a Diameter Sh HSS"](#) for instructions on editing this file.

Configuring Subscriber Data Quickly for a Test or Demonstration System

If you are just creating a test and evaluation system, the fastest and easiest way to add subscriber data to OCECAS is to provision the ESS NoSQL database using the REST interface. This does not make any subscriber information from an HSS available, but it is faster. See ["Provisioning the ESS with Subscriber Data"](#) for instructions.

Provisioning the ESS with Subscriber Data

You create and manage subscribers by writing provisioning tools that use RESTful HTTP commands to interact with RESTful APIs. This allows for easy integration with external systems.

For a complete description of the OCECAS RESTful APIs, see *Oracle Communications Evolved Communications Application Server SDC RESTful API Reference* and *Oracle Communications Evolved Communications Application Server UDR RESTful API Reference*.

This section explains how to provision and manage subscriber data in your ESS, including the following tasks:

- [Connecting to the ESS Server](#)
- [Creating a New Subscriber](#)
- [Getting Subscriber Data](#)
- [Amending Subscriber Data](#)
- [Deleting a Subscriber](#)
- [Getting Subscriber Aliases](#)
- [Updating a Subscriber Alias](#)
- [Getting jsonData for a Subscriber](#)
- [Adding jsonData for a Subscriber](#)

When creating or modifying subscriber data, the data must adhere to the ESS subscriber schema. See "[Understanding the Default ESS Objects](#)" for a description of the schema.

Connecting to the ESS Server

To log in to the ESS server, issue a POST command like the following one, replacing *hostname* and *port* with the host name and port number of the ESS server, and *authorization_value* with user credentials. The default port number of the ESS server is 6052.

```
POST /api/session HTTP/1.1
Host: hostname:port
Authorization: Basic authorization_value==
Cache-Control: no-cache
```

See "[Logging In to the Management Domain](#)" for information on how to obtain an authorization pass code.

Creating a New Subscriber

To create a new subscriber, issue a POST command like the one in this example, replacing *hostname* and *port* with the host name and port number of the ESS server:

```
POST /api/subscriber HTTP/1.1
Host: hostname:port
Content-Type: application/json
Cache-Control: no-cache
```

Include content to describe the subscriber, as shown in this example:

```
{
  "userIdentifiers": [
    {
      "id": "sip:6175551002@oracle.com",
      "type": "END_USER_SIP_URI"
    },
    {
      "id": "tel:6175551002",
      "type": "END_USER_SIP_URI"
    },
    {
      "id": "sip:+6175551002@oracle.com",
```


Amending Subscriber Data

To amend subscriber data, issue a PUT command identifying the subscriber with a GUID. See ["Getting Subscriber Data"](#) for information on obtaining a GUID.

This example changes the subscriber's `jsonData` field. To use this example, replace `hostname` and `port` with the host name and port number of the ESS server. Replace `dWRyMQAAAUuht5PxqzIelnGoEY81150AAAAH` with the subscriber's GUID

```
PUT
/api/subscriber/dWRyMQAAAUuht5PxqzIelnGoEY81150AAAAH/json/oracle.occas.csp.app.sdc
.OCECASUser HTTP/1.1
Host: hostname:port
Content-Type: application/json
Cache-Control: no-cache
```

This command writes a specific `jsonData` field to the subscriber's service data. For example:

```
{
  "ServiceProviders": {
    "MVN01": {
      "profile_data": {
        "TIR": {
          "Active": true,
          "Mode": "Permanent",
          "Default": "Restricted",
          "Restriction": "AssertedIdentity"
        }
      }
    }
  }
}
```

Deleting a Subscriber

To delete a subscriber, issue a DELETE command like the one in the following example, replacing `hostname` and `port` with the host name and port number of the ESS server and identifying the subscriber with a GUID. See ["Getting Subscriber Data"](#) for information on getting a subscriber's GUID.

```
DELETE /api/subscriber/dWRyMQAAAUuht5PxqzIelnGoEY81150AAAAH HTTP/1.1
Host: hostname:port
Cache-Control: no-cache
```

This example deletes the subscriber created in ["Creating a New Subscriber"](#).

Getting Subscriber Aliases

To get all aliases for a subscriber, issue a GET command like the one shown in the following example, specifying the subscriber with a GUID. Replace `hostname` and `port` with the host name and port number of the ESS server and replace `dWRyMQAAAUuht5PxqzIelnGoEY81150AAAAH` with the subscriber's GUID.

```
GET /api/subscriber/dWRyMQAAAUuht5PxqzIelnGoEY81150AAAAH/alias HTTP/1.1
Host: hostname:port
Cache-Control: no-cache
```

This example returns all aliases for the subscriber in an array of `userIdentifiers`:

```
{
```

```
"userIdentifiers": [
  {
    "id": "sip:6175551002@oracle.com",
    "type": "END_USER_SIP_URI"
  },
  {
    "id": "tel:6175551002",
    "type": "END_USER_SIP_URI"
  },
  {
    "id": "sip:+6175551002@oracle.com",
    "type": "END_USER_SIP_URI"
  }
],
}
```

Updating a Subscriber Alias

To update the alias of a subscriber, issue a PUT command like the one shown in the following example, specifying the subscriber with a GUID. Replace *hostname* and *port* with the host name and port number of the ESS server and replace `dWRyMQAAAU6pL51CsE9QWg4Z3LXrX3sAAAAM` with the subscriber's GUID.

```
PUT /api/subscriber/dWRyMQAAAU6pL51CsE9QWg4Z3LXrX3sAAAAM/alias
Host: hostname:port
Content-Type: application/json
Cache-Control: no-cache
```

Attach content to specify the subscriber's alias in a `userIdentifiers` array, as shown in this example:

```
{
  "userIdentifiers" : [ {
    "id" : "sip:subA@192.168.0.1:50640",
    "type" : "END_USER_SIP_URI"
  } ]
}
```

Note: JSON data other than `useridentifiers` is ignored.

A subsequent GET of the subscriber's data returns:

```
{
  "id": "dWRyMQAAAU6pL51CsE9QWg4Z3LXrX3sAAAAM",
  "userIdentifiers": [
    {
      "id": "sip:subA@192.168.0.1:50640", <--- updated
      "type": "END_USER_SIP_URI"
    }
  ],
  "jsonData": {
    "oracle.occas.csp.app.sdc.OCECASUser": {
      "Description": "subscriber: sip:subscriberA024652@192.168.0.1",
      "ServiceProviders": {
        "MVN01": {
          "profile_data": {
            "CallCount": 0,
            "CallFwdCompany": "",
            "Language": "en_GB",

```



```
Content-Type: application/json
Cache-Control: no-cache
```

Content:

```
{
  "Description": "subscriber: sip:subscriberA024652@192.168.0.1",
  "ServiceProviders": {
    "MVNO1": {
      "profile_data": {
        "CallCount": 0,
        "CallFwdCompany": "",
        "Language": "Chinese",
        "ToNumber": "",
        "Translations": {},
        "VMRedirect": ""
      },
      "services": {
        "VOICE": "ExtractStringProfileTagOk"
      }
    }
  }
}
```

A subsequent GET of subscriber data then returns the following data, in which the subscriber's language has been changed to Chinese.

```
{
  "id": "dWRyMQAAAU6pL51CsE9QWg4Z3LXrX3sAAAAM",
  "userIdentifiers": [
    {
      "id": "sip:subA@192.168.0.1:50640",
      "type": "END_USER_SIP_URI"
    }
  ],
  "jsonData": {
    "oracle.occas.csp.app.sdc.OCECASUser": {
      "Description": "subscriber: sip:subscriberA024652@192.168.0.1",
      "ServiceProviders": {
        "MVNO1": {
          "profile_data": {
            "CallCount": 0,
            "CallFwdCompany": "",
            "Language": "Chinese",
            "ToNumber": "",
            "Translations": {},
            "VMRedirect": ""
          },
          "services": {
            "VOICE": "ExtractStringProfileTagOk"
          }
        }
      }
    }
  }
}
```

<--- updated

Accessing Service Data with the RESTful API

This chapter describes how to use the Oracle Communications Evolved Communications Application Server (OCECAS) RESTful APIs to create and access service data.

For a complete description of the OCECAS RESTful API, see *Oracle Communications Evolved Communications Application Server SDC RESTful API Reference*.

About Accessing Service Data

You can create and access OCECAS service data by using HTTP RESTful API commands.

Note: You must first log in to the Management domain (see "[Logging In to the Management Domain](#)") before you can use the other API commands.

This section provides examples for performing the following operations on the OCECAS Management domain:

- [Logging In to the Management Domain](#)
- [Obtaining a CSRF Token](#)
- [Submitting the CSRF Token](#)
- [Creating a Change Set](#)
- [Getting Change Set Details](#)
- [Getting a Change Set](#)
- [Getting Change Set Differences](#)
- [Getting a List of Environments](#)
- [Getting a List of Pipelines](#)
- [Deploying a Change Set](#)
- [Getting Telemetry Records](#)
- [Getting Statistics Definitions](#)
- [Getting a Named Telemetry Record](#)
- [Getting Statistics](#)
- [Exporting a Control Flow](#)

- [Updating a Control Flow](#)
- [Importing a Control Flow](#)

Logging In to the Management Domain

To log in to the OCECAS Management domain, issue a POST command like the one in the following example, replacing *hostname* and *port* with the host name and port number of the management domain server. Replace the encoded authorization code with the encoded authorization code for your username and password:

```
POST /api/session HTTP/1.1
Host: hostname:port
Authorization: Basic dXNlcjplc2VyNHhkYw==
Cache-Control: no-cache
```

The response indicates the outcome of the login attempt.

You can create a script to obtain the encoded authorization pass code. For example, the following steps create a Python script to obtain the encoded authorization pass code and execute the script to display it.

1. Create Python script called **pass.py** using the following commands, replacing *name* and *password* with an actual username and password.

```
cat pass.py
import base64

username="name";
password="password"
base64string = base64.encodestring('%s:%s' % (username,
password)).replace('\n', '')
print base64string
```

2. Enter the following command to display the pass code:

```
python pass.py
```

Obtaining a CSRF Token

A Cross Site Request Forgery (CSRF) token is used to prevent cross site request forgeries. A CSRF token is required in all PUT and POST operations in the service-related RESTful APIs.

To obtain a CSRF token, issue the following GET command after logging into the management domain. Replace *hostname* and *port* with the host name and port number of the OCECAS management domain:

```
GET /api/session HTTP/1.1
Host: hostname:port
Authorization: Basic dXNlcjplc2VyNHhkYw==
Cache-Control: no-cache
```

The CSRF token is returned in the response:

```
{
  "name": "user",
  "fullName": "user",
  "userSession": {
    "name": null,
    "attributes": [
      {
```

```

      "key": "csrfToken",
      "value":
"2c3890b3bb4431a1b47465fbfc58a7d6830c3dbf03b80630767b44225c184ae5"
    },

```

Note: You must submit the CSRF token for all POST and PUT commands in service-related APIs. Otherwise you will receive a '403 Forbidden' error.

Submitting the CSRF Token

For a description of the CSRF token and information on how to obtain it, see ["Obtaining a CSRF Token"](#).

You can submit the CSRF token in a POST or PUT command either in an X-ORA-OCECAS-csrfToken header, as shown here, or in attached JSON content.

```

POST /api/pipelines/pipe/environments/env/deployments/ HTTP/1.1
Host: hostname:port
Content-Type: application/json
X-ORA-OCECAS-csrfToken:
2c3890b3bb4431a1b47465fbfc58a7d6830c3dbf03b80630767b44225c184ae5
Cache-Control: no-cache

```

Submit the CSRF token in JSON content as shown in the following example:

```

POST /api/pipelines/pipe/environments/env/deployments/ HTTP/1.1
Host: hostname:port
Content-Type: application/json
Cache-Control: no-cache

```

Content:

```

{
  "csrfToken": 2c3890b3bb4431a1b47465fbfc58a7d6830c3dbf03b80630767b44225c184ae5
  "name": "Test",
  "description": "A new test change set"
}

```

Creating a Change Set

To create a change set, issue a POST command like the one in the following example, replacing *hostname* and *port* with the host name and port number of the OCECAS management domain:

```

POST /api/change-sets HTTP/1.1
Host: hostname:port
Content-Type: application/json
Cache-Control: no-cache

```

Attach JSON content like that shown in the following example to specify the name of the change set and a description. Replace the `csrfToken` value with the value for your authorization credentials. See ["Obtaining a CSRF Token"](#) for more information.

```

{
  "csrfToken": 2c3890b3bb4431a1b47465fbfc58a7d6830c3dbf03b80630767b44225c184ae5
  "name": "Test",
  "description": "A new test change set"
}

```

Getting Change Set Details

To get change-set details, including the change set ID, issue a GET command like the one in the following example, replacing *hostname* and *port* with the host name and port number of the OCECAS management domain:

```
GET /api/change-sets HTTP/1.1
Host: hostname:port
Cache-Control: no-cache
```

Returns a `changeSets` object that contains a list of currently available change sets.

Getting a Change Set

To get a change set, issue a GET command like the one in the following example, replacing *hostname* and *port* with the host name and port number of the OCECAS management domain, and replacing *ChangeSetID* with the ID of the target change set. See "[Getting Change Set Details](#)" for information about finding a change set ID.

```
GET /api/change-sets/ChangeSetID HTTP/1.1
Host: hostname:port
Cache-Control: no-cache
```

The response returns a `changeSet` object for the named change set.

Getting Change Set Differences

To get a list of differences for a change set, compared with its base change set, issue a GET command like the one in the following example. Replace *hostname* and *port* with the host name and port number of the OCECAS management domain, and replace *ChangeSetID* with the ID of the target change set. See "[Getting Change Set Details](#)" for information about finding a change set ID.

```
GET /api/change-sets/ChangeSetID/diffs HTTP/1.1
Host: hostname:port
Cache-Control: no-cache
```

The response returns the differences for the specified change set, including a list of the control flows and resources that were either added or changed as part of that change set.

Getting a List of Environments

To get a list of environment details, including environment IDs, issue a GET command like the one in the following example, replacing *hostname* and *port* with the host name and port number of the OCECAS management domain:

```
GET /api/environments HTTP/1.1
Host: hostname:port
Cache-Control: no-cache
```

Returns a list of all environments.

Getting a List of Pipelines

To get a list of all pipelines, including IDs, issue a GET command like the one in the following example, replacing *hostname* and *port* with the host name and port number of the OCECAS management domain:

```
GET /api/pipelines HTTP/1.1
```

```
Host: hostname:port
Cache-Control: no-cache
```

Returns a list of all pipelines.

Deploying a Change Set

To deploy a change set, issue a POST command like the one in the following example, replacing *hostname* and *port* with the host name and port number of the OCECAS management domain. Replace *pipe* with the pipeline ID and *env* with the environment ID in the pipeline to which you want to deploy the change set. See ["Getting a List of Pipelines"](#) for information on finding a pipeline ID and ["Getting a List of Environments"](#) for information on finding an environment ID.

```
POST /api/pipelines/pipe/environments/env/deployments/ HTTP/1.1
Host: hostname:port
Content-Type: application/json
Cache-Control: no-cache
```

Attach JSON content that describes the deployment object. For example the following lines define the deployment object for the OCECAS "VoLTE, VoWiFi, and eSRVCC" change set. Replace the `csrfToken` value with the value for your authorization credentials. See ["Obtaining a CSRF Token"](#) for more information.

```
{
  "csrfToken": 2c3890b3bb4431a1b47465fbfc58a7d6830c3dbf03b80630767b44225c184ae5,
  "id": 41,
  "changeSet": {
    "csrfToken": null,
    "id": 2,
    "name": "VoLTE, VoWiFi and eSRVCC",
    "description": "VoLTE, VoWiFi and eSRVCC change set",
    "serviceProviderId": 1,
    "baselineId": 1,
    "pipelineId": null,
    "creationDate": 1437615562000,
    "lastUpdated": 1437615562000,
    "deleted": false,
    "editable": false,
    "inactive": false,
    "active": true,
    "scheduled": false,
    "retracted": false,
    "open": false,
    "erroredObjects": [],
    "username": "VoLTE Installer"
  },
  "baselineId": 1,
  "state": "A",
  "creationDate": 1437615744000,
  "scheduledDate": 1437615744000,
  "lastUpdated": 1437615744000,
  "environment": 1,
  "productionEnvironment": false,
  "pipeline": 1,
  "conflictPotential": null,
  "active": true,
  "scheduled": false,
  "editable": true
}
```

The response returns the `deployment` object for the specified change set.

Getting Telemetry Records

To get telemetry records, including the record names, issue a GET command like the one in the following example. Replace *hostname* and *port* with the host name and port number of the OCECAS management domain:

```
GET /api/telemetry/records HTTP/1.1
Host: hostname:port
Cache-Control: no-cache
```

Returns a `telemetry` object that contains a list of telemetry records.

Getting a Named Telemetry Record

To get the details of a telemetry record for a given name, issue a GET command like the one in the following example, replacing *hostname* and *port* with the host name and port number of the OCECAS management domain, and replacing *name* with the telemetry name. See "[Getting Telemetry Records](#)" for information about finding a telemetry name.

```
GET /api/telemetry/records/name/name HTTP/1.1
Host: hostname:port
Cache-Control: no-cache
```

The response returns the details of the telemetry record of the given name.

Getting Statistics Definitions

To get statistics definitions for a change set, including the names of statistics, issue a GET command like the one in the following example. Replace *hostname* and *port* with the host name and port number of the OCECAS management domain, and replace *changeSetID* with the ID of the change set for which you want statistics definitions. See "[Getting Change Set Details](#)" for information on finding a change set ID.

```
GET /api/statistics/definitions/changeSetID HTTP/1.1
Host: hostname:port
Cache-Control: no-cache
```

Returns a `Statistics` object that contains the statistics definitions.

Getting Statistics

To get statistics, issue a GET command like the one in the following example, replacing *hostname* and *port* with the host name and port number of the OCECAS management domain, and replacing *name* with the statistic name. See "[Getting Statistics Definitions](#)" for information on finding a statistic name.

```
GET /api/statistics/records/name/name HTTP/1.1
Host: hostname:port
Cache-Control: no-cache
```

The response returns a `statistics` object containing all of the statistics records matching the given name.

Exporting a Control Flow

To get a control flow, issue a GET command like the one in the following example, replacing *hostname* and *port* with the host name and port number of the OCECAS management domain. Replace *changeSetID* in the header with the change set ID and *name* with the name of the control flow. You can obtain the control flow name from the response returned in ["Getting Change Set Differences"](#).

All prerequisite configuration data, including service data, external concepts, statistic definitions, EDRs and schemas referenced directly or indirectly by activities of the flow, is included in the exported file.

```
GET /api/change-sets/changeSetID/access-control-flows/name HTTP/1.1
Host: hostname:port
Accept: text/xml
Cache-Control: no-cache
```

The response returns the latest version of the `controlFlow` instance.

Note: The export does not include application triggers and the related VoLTE-specific service data that underpins the application view. This information cannot be computed as a prerequisite of any control flow. If a VoLTE installation has been lost or removed, the applications information will be lacking after all flows are reimported. If necessary, you can reinstate this using application trigger and service data REST requests.

Updating a Control Flow

To update a control flow, issue a PUT command like the one in the following example, replacing *hostname* and *port* with the host name and port number of the OCECAS management domain. Replace *changeSetID* in the header with the change set ID and *name* with the name of the control flow. Attach JSON content to describe the control flow. See ["Importing a Control Flow"](#) for an example of JSON content for a control flow.

```
PUT /api/change-sets/changeSetID/access-control-flows/name HTTP/1.1
Host: hostname:port
Content-Type: text/json
Cache-Control: no-cache
```

Importing a Control Flow

To add a control flow, issue a POST command like the one in the following example. Replace *hostname* and *port* with the host name and port number of the OCECAS management domain. Replace *ChangeSetID* with the ID of the target change set and attach JSON content like that shown in the example to describe the control flow:

```
POST /api/change-sets/ChangeSetID/access-control-flows HTTP/1.1
Host: hostname:port
Content-Type: text/json
Cache-Control: no-cache
```

All prerequisite configuration data, including service data, external concepts, statistic definitions, EDRs and schemas referenced directly or indirectly by activities of the control flow, may be included in the imported file and will be used if present.

You might encounter compile issues in cases where a parent control flow is imported before its child control flows are all imported. The parent will report that one or more

linked control flows cannot be found. To resolve this issue, continue importing the child flows until all are present. Then reimport the parent, which requires a PUT operation because the parent already exists.

It should not be necessary to manually edit any imported control flow. A reimport might be necessary in the case where a child was initially lacking, because the linked control flow name is not saved in the parent control flow. It is saved only in its exported form.

The following JSON content describes a simple control flow with Start, Release, and End Activities. Replace the `csrfToken` value with the value for your authorization credentials. See ["Obtaining a CSRF Token"](#) for more information.

```
{
  "csrfToken": "2c3890b3bb4431a1b47465fbfc58a7d6830c3dbf03b80630767b44225c184ae5",
  "name": "example",
  "description": null,
  "loopsAllowed": false,
  "allowedAsLinkedControlFlow": true,
  "activities": [
    {
      "version": 1,
      "activityNumber": 1,
      "name": "Start",
      "activityType": "Start",
      "fastKey": "ST",
      "location": "420,98,100,60",
      "comments": null,
      "parameterList": [
        {
          "key": null,
          "attributeList": []
        },
        {
          "key": "Loop Limit",
          "attributeList": [
            {
              "name": "int",
              "value": "100"
            }
          ]
        }
      ]
    },
    {
      "key": "Loop Limit External",
      "attributeList": [
        {
          "name": "int",
          "value": "100"
        }
      ]
    },
    {
      "key": "Loops Allowed",
      "attributeList": [
        {
          "name": "check",
          "value": "false"
        }
      ]
    }
  ]
}
```



```

        "linkedFlowName": null,
        "serviceName": null,
        "exit": [
            {
                "branchNumber": 0,
                "branchName": null,
                "linkedControlFlowExitActivityNumber": "",
                "linkedControlFlowExitActivityName": null,
                "connection": [
                    2
                ]
            }
        ]
    },
    {
        "version": 1,
        "activityNumber": 2,
        "name": "Release",
        "activityType": "Release",
        "fastKey": "REL",
        "location": "421,196,100,60",
        "comments": null,
        "parameterList": [
            {
                "key": null,
                "attributeList": []
            },
            {
                "key": "Session Endpoint",
                "attributeList": [
                    {
                        "name": "Entries",
                        "value": "IE:DE"
                    },
                    {
                        "name": "EntriesType",
                        "value": "EventSource"
                    },
                    {
                        "name": "string",
                        "value": "IE"
                    }
                ]
            },
            {
                "key": "Code",
                "attributeList": [
                    {
                        "name": "int",
                        "value": "480"
                    },
                    {
                        "name": "MIN",
                        "value": "100"
                    },
                    {
                        "name": "MAX",
                        "value": "699"
                    }
                ]
            }
        ]
    }
}

```

```

        "name": "CORRECTING",
        "value": "1"
    }
  ]
}
],
"linkedFlowName": null,
"serviceName": null,
"exit": [
  {
    "branchNumber": 0,
    "branchName": null,
    "linkedControlFlowExitActivityNumber": null,
    "linkedControlFlowExitActivityName": null,
    "connection": [
      3
    ]
  },
  {
    "branchNumber": 0,
    "branchName": null,
    "linkedControlFlowExitActivityNumber": null,
    "linkedControlFlowExitActivityName": null,
    "connection": [
      3
    ]
  }
]
},
{
  "version": 1,
  "activityNumber": 3,
  "name": "End",
  "activityType": "End",
  "fastKey": "END",
  "location": "426,315,100,60",
  "comments": null,
  "parameterList": [
    {
      "key": null,
      "attributeList": []
    }
  ],
  "linkedFlowName": null,
  "serviceName": null,
  "exit": []
}
]
}

```

The Default ESS Subscriber Store Schema

This appendix explains the default schema for the Oracle Communications Evolved Application Server (OCECAS) subscriber server (ESS).

Understanding the Default ESS Objects

This section explains the structure of the ESS schema that OCECAS uses to specify and store subscriber data. [Figure A-1](#) shows the top level schema objects. All of the objects in bold font are required, except for the referenced objects in the **profile_data** object.

Elements in the schema relate to data that is accessible from the control flow editor. For example, the `$ref` notation is a key, whose corresponding value indicates the name of the related external concept item.

Figure A-1 ESS Top Level Schema Objects

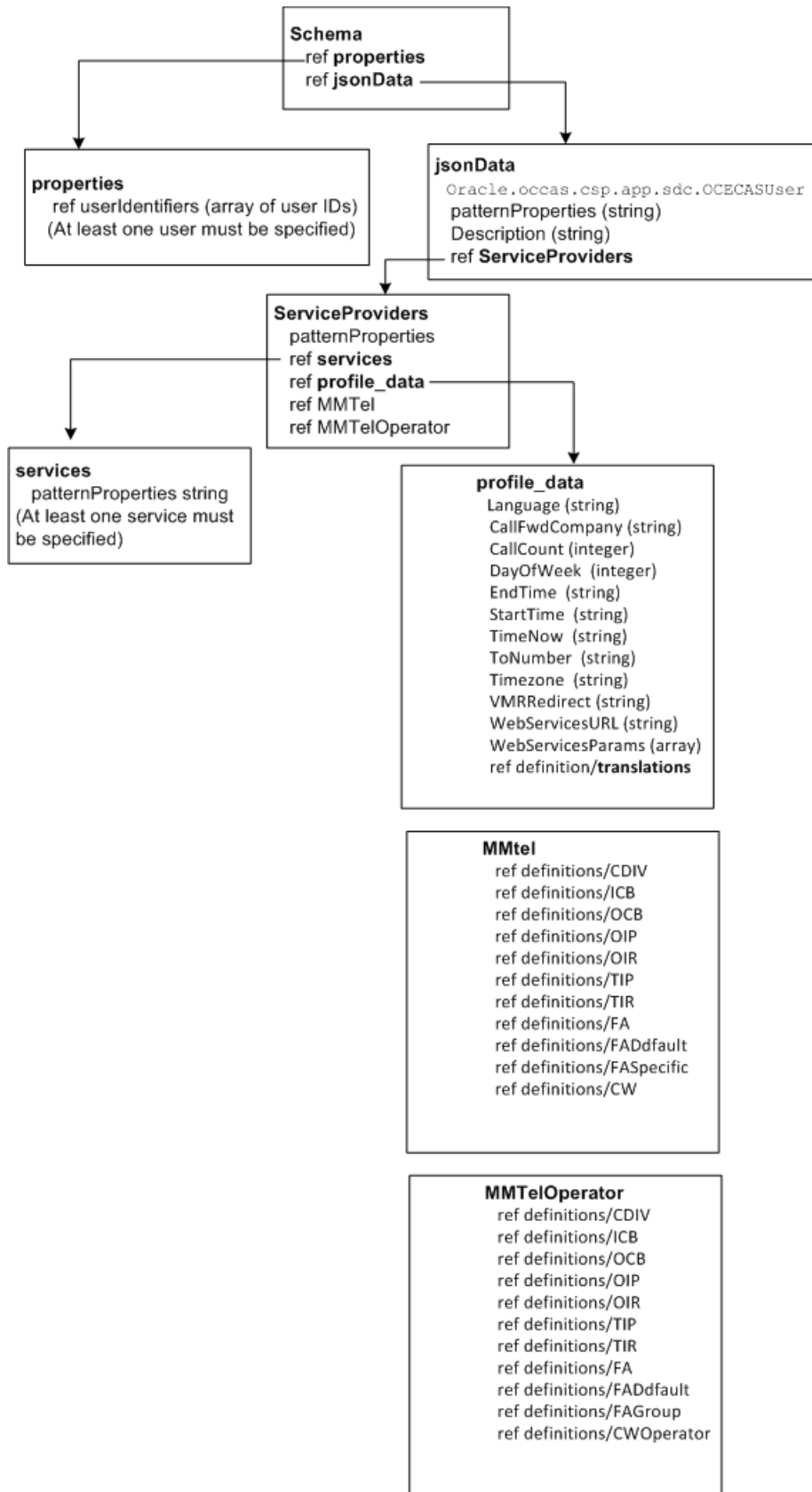


Figure A-2 shows a detail view of the ESS schema **MMTel** object and the objects that it references.

The referenced objects include these supplementary services objects:

- **CDIV** - Communication Diversion
- **ICB** - Incoming Call Barring
- **OCB** - Outgoing Call Barring
- **OIP** - Originating Identification Presentation
- **OIR** - Originating Identification Restriction
- **TIP** - Terminating Identification Presentation
- **TIR** - Terminating Identification Restriction
- **FA** - Flexible Alerting
- **FADefault** - Flexible Alerting Default
- **FASpecific** - Flexible Alerting Specific
- **CW** - Communication Waiting

Figure A-2 ESS Schema **MMTel** Object

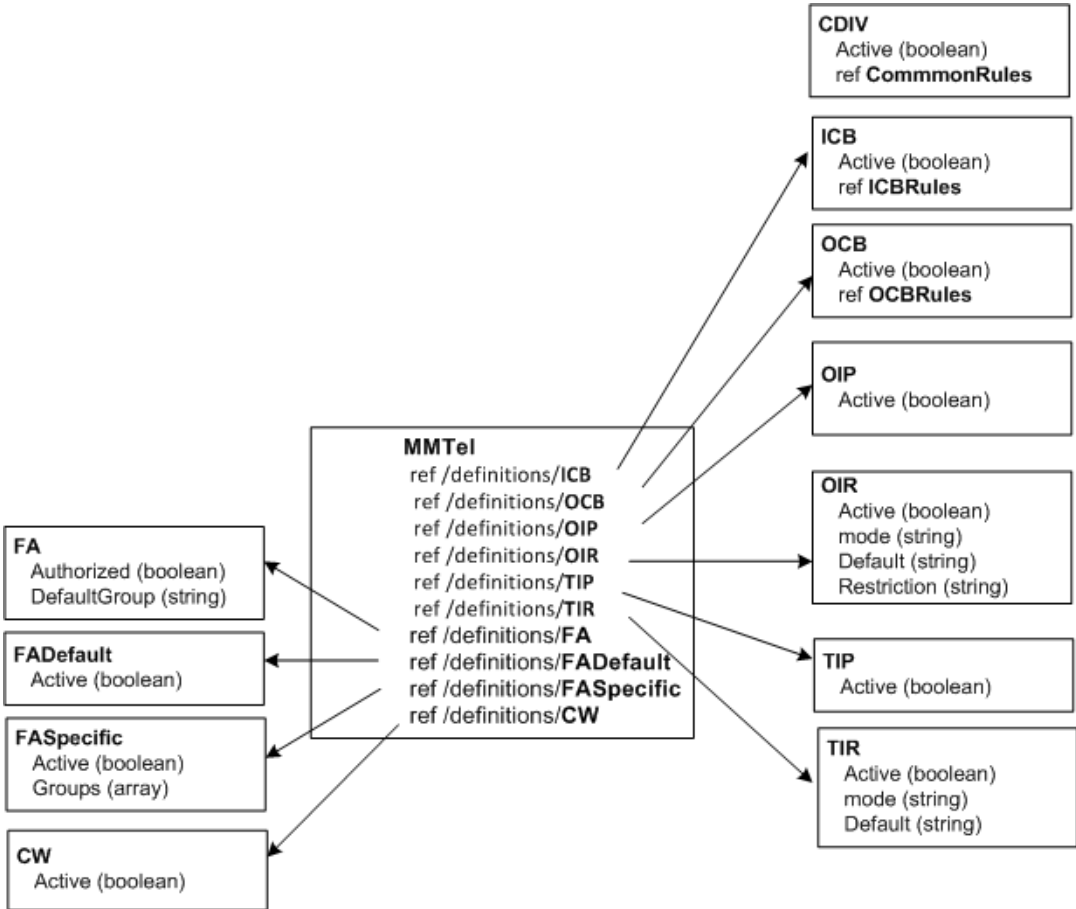
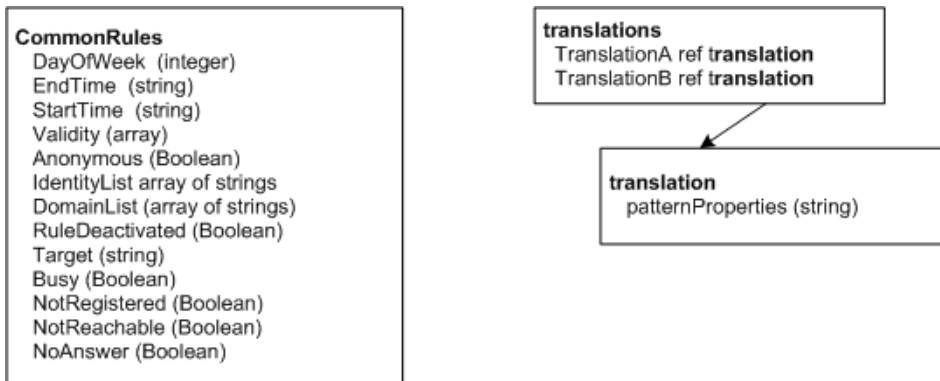


Figure A-3 shows a detail view of the **CommonRules** and **Translations** objects. The **CommonRules** object specifies parameters for using supplementary services that require them. For example start and stop times for the service to take effect.

You use the **Translations** object to change subscriber IDs. This object translates IDs from the **TranslationA** block of name-value pairs, to the different values in the **TranslationB** block of name-value pairs. The change can be a simple number switch, or you could change the number value, or format, or both. In this example, only one name matches (**tel:901**), so that is the only ID translated. It changes from **tel:+133455601001** to **tel:+133455601002**:

```
"Translations": {
  "TranslationA": {
    "tel:901;phone-context=shortcode": "tel:+133455601007",
    "tel:901;noa=unknown;phone-context=oracle.occas.csp": "tel:+133455601001",
    "sip:c1@personal.redirect.com": "sip:called@full.address.com"
  },
  "TranslationB": {
    "tel:902;phone-context=shortcode": "tel:+133455601008",
    "tel:901;noa=unknown;phone-context=oracle.occas.csp": "tel:+133455601002",
    "sip:c2@personal.redirect.com": "sip:called2@full.address.com"
  }
}
```

Figure A-3 ESS Schema CommonRules and translations Objects



Default Subscriber Record

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "description": "Schema for Udr Subscriber data",
  "type": "object",
  "properties": {
    "version": "integer",
    "userIdentifiers": { "$ref": "#/definitions/userIdentifiers" },
    "jsonData": {
      "type": "object",
      "properties": {
        "oracle.occas.csp.app.sdc.OCECASUser": {
          "type": "object",
          "properties": {
            "Description": { "type": "string" },
            "ServiceProviders": { "$ref": "#/definitions/ServiceProviders" }
          },
          "required": ["ServiceProviders"],
          "additionalProperties": false
        }
      },
      "additionalProperties": false
    }
  }
}
```

```

    }
  },
  "patternProperties": {
    "_N_*": { "type": "string" }
  },
  "required": ["userIdentifiers", "jsonData"],
  "additionalProperties": false,

  "definitions": {
    "userIdentifiers": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "id": { "type": "string" },
          "type": { "enum": [
            "END_USER_E164", "END_USER_IMSI", "END_USER_SIP_URI",
            "END_USER_NAI", "END_USER_PRIVATE", "END_USER_GLOBAL_UID"
          ] }
        }
      },
      "required": ["id", "type"],
      "additionalProperties": false
    },
    "additionalItems": false,
    "minItems": 1
  },

  "ServiceProviders": {
    "type": "object",
    "patternProperties": {
      "[a-zA-Z0-9][a-zA-Z0-9]*": {
        "type": "object",
        "properties": {
          "services": { "$ref": "#/definitions/services" },
          "profile_data": { "$ref": "#/definitions/profile_data" },
          "MMTel": { "$ref": "#/definitions/MMTel" },
          "MMTelOperator": { "$ref": "#/definitions/MMTelOperator" }
        },
        "required": ["services", "profile_data"],
        "additionalProperties": false
      }
    },
    "minProperties": 1,
    "additionalProperties": false
  },

  "services": {
    "type": "object",
    "patternProperties": {
      "[a-zA-Z0-9][a-zA-Z0-9]*": {
        "type": "string"
      }
    },
    "additionalProperties": false,
    "minProperties": 1
  },

  "profile_data": {

```

```

    "type": "object",
    "properties": {
      "Language": { "type": "string" },
      "CallFwdCompany": { "type": "string" },
      "CallCount": { "type": "integer" },
      "DayOfWeek": { "type": "integer" },
      "EndTime": { "type": "string" },
      "StartTime": { "type": "string" },
      "TimeNow": { "type": "string" },
      "ToNumber": { "type": "string" },
      "Timezone": { "type": "string" },
      "VMRedirect": { "type": "string" },
      "WebServicesURL": { "type": "string" },
      "WebServicesParams": { "type": "array" },
      "Translations": { "$ref": "#/definitions/translations" }
    }
  },

  "MMTel": {
    "type": "object",
    "properties": {
      "CDIV": { "$ref": "#/definitions/CDIV" },
      "ICB": { "$ref": "#/definitions/ICB" },
      "OCB": { "$ref": "#/definitions/OCB" },
      "OIP": { "$ref": "#/definitions/OIP" },
      "OIR": { "$ref": "#/definitions/OIR" },
      "TIP": { "$ref": "#/definitions/TIP" },
      "TIR": { "$ref": "#/definitions/TIR" },
      "FA": { "$ref": "#/definitions/FA" },
      "FADefault": { "$ref": "#/definitions/FADefault" },
      "FASpecific": { "$ref": "#/definitions/FASpecific" },
      "CW": { "$ref": "#/definitions/CW" }
    },
    "patternProperties": {
      "_N_*": { "type": "string" }
    },
    "additionalProperties": true
  },

  "MMTelOperator": {
    "type": "object",
    "properties": {
      "CDIV": { "$ref": "#/definitions/CDIV" },
      "ICB": { "$ref": "#/definitions/ICB" },
      "OCB": { "$ref": "#/definitions/OCB" },
      "OIP": { "$ref": "#/definitions/OIP" },
      "OIR": { "$ref": "#/definitions/OIR" },
      "TIP": { "$ref": "#/definitions/TIP" },
      "TIR": { "$ref": "#/definitions/TIR" },
      "FA": { "$ref": "#/definitions/FA" },
      "FAGroup": { "$ref": "#/definitions/FAGroup" },
      "CW": { "$ref": "#/definitions/CWOperator" }
    }
  },

  "translations": {
    "type": "object",
    "properties": {
      "TranslationA": { "$ref": "#/definitions/translation" },
      "TranslationB": { "$ref": "#/definitions/translation" }
    }
  }
}

```



```

    },
    "additionalProperties": false
  },
  "translation": {
    "type": "object",
    "patternProperties": {
      "^[a-zA-Z0-9][a-zA-Z0-9_@;.-]*$": {
        "type": "string"
      }
    },
    "additionalProperties": false
  },
  "OIR": {
    "type": "object",
    "properties": {
      "Active": { "type": "boolean" },
      "Mode": { "type": "string" },
      "Default": { "type": "string" },
      "Restriction": { "type": "string" }
    },
    "patternProperties": {
      "_N.*": { "type": "string" }
    },
    "additionalProperties": true
  },
  "OIP": {
    "type": "object",
    "properties": {
      "Active": { "type": "boolean" }
    },
    "patternProperties": {
      "_N.*": { "type": "string" }
    },
    "additionalProperties": false
  },
  "TIR": {
    "type": "object",
    "properties": {
      "Active": { "type": "boolean" },
      "Mode": { "type": "string" },
      "Default": { "type": "string" }
    },
    "patternProperties": {
      "_N.*": { "type": "string" }
    },
    "additionalProperties": false
  },
  "TIP": {
    "type": "object",
    "properties": {
      "Active": { "type": "boolean" }
    },
    "patternProperties": {
      "_N.*": { "type": "string" }
    },
  },

```

```
"additionalProperties": false
},
"CDIV": {
  "type": "object",
  "properties": {
    "Active": { "type": "boolean" },
    "Rules": {
      "type": "array",
      "items": { "$ref": "#/definitions/CommonRules" }
    }
  },
  "patternProperties": {
    "_N_*": { "type": "string" }
  },
  "additionalProperties": true
},
"ICB": {
  "type": "object",
  "properties": {
    "Active": { "type": "boolean" },
    "Rules": {
      "type": "array",
      "items": { "$ref": "#/definitions/ICBRules" }
    }
  },
  "patternProperties": {
    "_N_*": { "type": "string" }
  },
  "additionalProperties": false
},
"OCB": {
  "type": "object",
  "properties": {
    "Active": { "type": "boolean" },
    "Rules": {
      "type": "array",
      "items": { "$ref": "#/definitions/OCBRules" }
    }
  },
  "patternProperties": {
    "_N_*": { "type": "string" }
  },
  "additionalProperties": false
},
"FADefault": {
  "type": "object",
  "properties": {
    "Active": { "type": "boolean" }
  },
  "patternProperties": {
    "_N_*": { "type": "string" }
  },
  "additionalProperties": false
},
"FASpecific": {
```

```

    "type": "object",
    "properties": {
      "Active": { "type": "boolean" },
      "Groups": {
        "type": "array",
        "items": { "$ref": "#/definitions/FAIdentity" }
      }
    },
    "patternProperties": {
      "_N_*": { "type": "string" }
    },
    "additionalProperties": false
  },
  "FAIdentity": {
    "type": "object",
    "properties": {
      "Identity": { "type": "string" },
      "Active": { "type": "boolean" }
    },
    "patternProperties": {
      "_N_*": { "type": "string" }
    },
    "additionalProperties": false
  },
  "FA": {
    "type": "object",
    "properties": {
      "Authorized": { "type": "boolean" },
      "DefaultGroup": { "type": "string" }
    },
    "patternProperties": {
      "_N_*": { "type": "string" }
    },
    "additionalProperties": false
  },
  "FAGroup": {
    "type": "object",
    "properties": {
      "Authorized": { "type": "boolean" },
      "Identity": { "type": "string" },
      "Type": { "type": "string" },
      "Membership": { "type": "string" },
      "Members": {
        "type": "array",
        "items": { "$ref": "#/definitions/FAGroupMember" }
      }
    },
    "patternProperties": {
      "_N_*": { "type": "string" }
    },
    "additionalProperties": false
  },
  "FAGroupMember": {
    "type": "object",
    "properties": {
      "Identity": { "type": "string" },

```

```

    "Active": { "type": "boolean" }
  },
  "patternProperties": {
    "_N_*": { "type": "string" }
  },
  "additionalProperties": false
},

"CW": {
  "type": "object",
  "properties": {
    "Active": { "type": "boolean" }
  },
  "patternProperties": {
    "_N_*": { "type": "string" }
  },
  "additionalProperties": false
},

"CWOperator": {
  "type": "object",
  "properties": {
    "Authorized": { "type": "boolean" },
    "CallingUserNotification": { "type": "boolean" },
    "WaitingTimer": { "type": "integer" }
  },
  "additionalProperties": false
},

"CommonRules": {
  "type": "object",
  "properties": {
    "DayOfWeek": { "type": "integer" },
    "EndTime": { "type": "string" },
    "StartTime": { "type": "string" },
    "Validity": { "type": "array", "items": { "$ref": "#/definitions/Validity"
} } },
  "Anonymous": { "type": "boolean" },
  "IdentityList": { "type": "array", "items": { "type": "string" } },
  "DomainList": { "type": "array", "items": { "$ref":
"/#/definitions/DomainList" } },
  "RuleDeactivated": { "type": "boolean" },
  "Target": { "type": "string" },
  "Busy": { "type": "boolean" },
  "NotRegistered": { "type": "boolean" },
  "NotReachable": { "type": "boolean" },
  "NoAnswer": { "type": "boolean" }
},
  "patternProperties": {
    "_N_*": { "type": "string" }
  },
  "additionalProperties": true
},

"ICBRules": {
  "type": "object",
  "properties": {
    "DayOfWeek": { "type": "integer" },
    "EndTime": { "type": "string" },
    "StartTime": { "type": "string" },

```

```

    "Validity": { "type": "array", "items": { "$ref": "#/definitions/Validity"
  } },
  "Anonymous": { "type": "boolean" },
  "IdentityList": { "type": "array", "items": { "type": "string" } },
  "DomainList": { "type": "array", "items": { "$ref":
"/definitions/DomainList" } },
  "CommunicationDiverted": { "type": "boolean" },
  "Roaming": { "type": "boolean" },
  "RuleDeactivated": { "type": "boolean" },
  "Allow": { "type": "boolean" }
},
"patternProperties": {
  "_N_*": { "type": "string" }
},
"additionalProperties": true
},
"OCBRules": {
  "type": "object",
  "properties": {
    "DayOfWeek": { "type": "integer" },
    "EndTime": { "type": "string" },
    "StartTime": { "type": "string" },
    "Validity": { "type": "array", "items": { "$ref": "#/definitions/Validity"
  } },
    "IdentityList": { "type": "array", "items": { "type": "string" } },
    "DomainList": { "type": "array", "items": { "$ref":
"/definitions/DomainList" } },
    "International": { "type": "boolean" },
    "InternationalExHC": { "type": "boolean" },
    "Roaming": { "type": "boolean" },
    "RuleDeactivated": { "type": "boolean" },
    "Allow": { "type": "boolean" }
  },
  "patternProperties": {
    "_N_*": { "type": "string" }
  },
  "additionalProperties": true
},
"DomainList": {
  "type": "object",
  "properties": {
    "Identity": { "type": "string" },
    "DomainExceptList": { "type": "array", "items": { "type": "string" } },
    "IdentityExceptList": { "type": "array", "items": { "type": "string" } }
  },
  "patternProperties": {
    "_N_*": { "type": "string" }
  },
  "additionalProperties": true
},
"Validity": {
  "type": "object",
  "properties": {
    "From": { "type": "string" },
    "Until": { "type": "string" }
  },
  "patternProperties": {

```

```
    "_N_*": { "type": "string" }  
  },  
  "additionalProperties": true  
}  
  
}  
}
```

Activities Reference

This appendix describes the Activities available for creating control flows in the Session Design Center of Oracle Communications Evolved Communications Application Server (OCECAS).

Add EDR Field Value

Fast key: **AEDR**



Stores into an EDR field a value that is either directly input or comes from an external concept.

The activity has two parameters:

- **EDR Field**, select from the list, which is populated after typing some letters. For possible letters to type in, check the list of the predefined EDR field names below:
 - URL: Web Request URL
 - cst: Change Set ID
 - dep: Deployment ID
 - dom: Domain Name
 - mth: Web Request Method name
 - pfm: Platform Info EDR Group
 - res: HTTP response Code
 - rht: Remote Host
 - rqt: Time of request
 - sedt: End Datetime
 - srv: Server Name
 - tfn: Activities traversed
 - usr: User Name

- web: Web Request EDR Group
 - b2br: B2B Session Result
 - b2et: B2B Session End Time
 - b2st: B2B Session Start Time
 - bar: Call Barred
 - call: Call Info EDR Group
 - cfn: Control Flow Names
 - chs: Chassis Info EDR Group
 - cid: SIP Call ID
 - cld: Called URI
 - clg: Calling URI
 - ctry: Country Code
 - ctyp: Call Type
 - dclid: Diverted Called URI
 - drul: Call Diversion Rule
 - eec: UDR Error Code
 - eer: UDR Error Reason
 - pcl: Protocol
 - pvd: Service Provider
 - rcldv: Redirected Called URI
 - relc: SIP Release Cause
 - rgstn: Registration
 - roam: Call Roaming
 - spmt: SIP Method
 - ssdt: Start Datetime
 - udr: User Data EDR Group
- **EDR Value** A value that can be directly input or comes from an external concept.

The activity has two exits:

- **Success**, when added successfully.
- **Error**, when encountering invalid parameters, edr field, or edr value; or unable to retrieve information from the specified external concept.

Adjust Media

Fast Key: **ADJM**



Mutes or unmutes the media stream for a conference participant.

The activity has two parameters:

- **Session Endpoint**, initiating endpoint or event source.
- **Adjustment**, MUTE or UNMUTE.

The activity has six exits:

- **Success**
- **Error (Server)**
- **Error (Media)**
- **Error (Timeout)**
- **Error (Not Supported)**
- **Error for any other reasons**

Alarm

Fast Key: ALM



Logs an occurrence of the configured alarm. The activity performs the following operations:

1. Gathers the Text and Resolution.
2. Raises the alarm.
3. Exits a branch based on response status.

The activity has three parameters:

- **OID Postfix**, a value that can be directly input or comes from an external concept. It is used together with the OID prefix configured in the Alarm tab in the OCECAS administration console to form a complete unique OID that identifies the alarm.
- **Alarm Text**, the value can be input directly or come from an external concept. It describes what has happened.
- **Alarm Resolution**, the value can be a direct input or from an external concept. It provides the resolution information.

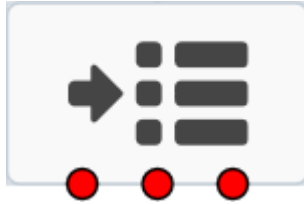
The activity has two exits:

- **Success** when the alarm is raised successfully.

- **Error** when the input is invalid or the activity is unable to retrieve information from the specified external concept.

Array Index

Fast Key: **AIDX**



Sets and stores the index of an array that is then used for accessing that array. The activity performs the following operations:

1. Initiates a Retrieve Context Action request to get the array size. If the operator is Increment or Decrement, it also retrieves the current array index.
2. Processes the Retrieve Context Action response, which consists of calculating and validating the new array index and storing it or deleting the existing array index value.
3. Processes the Store Context Action response.
4. Processes the Delete Context Action response.

The activity has three parameters:

- **Array**, the value is from an external concept of the array type.
- **Index**, a value that is input directly or obtained from an external concept.
- **Operator**, Set, Zero, Max, Increment, Decrement, New and Clear Index.

The activity has three exits:

- **Success** when setting and storing the array index successfully.
- **Not Found** when unable to locate the array while the operator is not New.
- **Error** when encountering invalid parameters or values, or failed use the operator to set the current array index, or failed to store the array index.

Compare

Fast Key: **COMP**



Compares a source external concept with a given value or another external concept. The activity performs the following operations:

1. Verifies ContextKey parameter is valid. Specifies the **/UserData/Service/View/Field** path.

2. Verifies Comparison string is valid.
3. Invokes RetrieveContextValue ActionRequest with specified ContextKey:
 - ReadFromUdr Action intercepts the RCV ActionRequest and inspects the ContextKey. If prefix is '/UserData', then continue.
 - Invoke UDR Facade for given ContextKey.
 - Process UDR response - extract data from returned view.
 - Store extracted data into context map.
 - Defer to standard RetrieveContextValue Action, which populates the ActionResponse and returns to the Activity.
4. Verifies successful query
5. Compares retrieved value with specified comparison string
6. Branches on comparison result

The activity has three parameters:

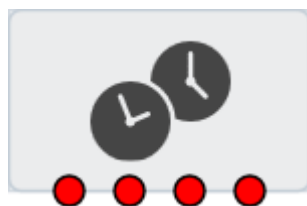
- **Source**, the value is an external concept.
- **Compare With**, the target value, which is input directly or comes from another external concept. It is optional if the comparator is EXIST.
- **Comparator** - regex, =, !=, <, <=, >, >=, and exist.

The activity has four exits:

- **Success** when comparison succeeds.
- **Not Matched** when comparison failed.
- **Not Found** when failed to locate source and the comparator is not EXIST, or failed to locate the target value.
- **Error** when failed to locate source and the comparator is EXIST.

Compare Date Time

Fast Key: CPDT



Compares a range of date time types, including Date, Time of Day, and Date and Time. The activity performs the following operations:

The activity has six parameters:

- **Timezone**, used to adjust the target datetime value.
- **Source**, a value from an external concept.
- **Compare With**, a value that is input directly or comes from an external concept. This is the value that is compared against the source.

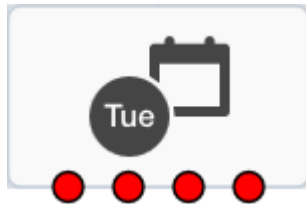
- **Compare Now**, a checkbox that specifies a target value that is the current time. This value takes precedence over the derived target value from the Compare With parameter and is compared against the source.
- **Comparator**: =, !=, <, <=, >, and >=
- **Date and Time Type**: One of the following types for the comparison.
 - Date
 - Time of Day
 - Date and Time

The activity has four exits:

- **Success** when comparison succeeds.
- **Not Matched** when comparison fails.
- **Not Found** when unable to locate the values of all external concepts.
- **Error** when failed to get the target date value or date parameter values are invalid.

Compare Day of Week

Fast Key: CPDW



Compares day-of-week values. The activity performs the following operations:

1. Obtain the input parameters
2. Get required values from Context
3. Compare retrieved value with specified comparison value.
4. Branch on comparison result

The activity has six parameters:

- **Timezone**, adjusts the target datetime value.
- **Source**, a value from an external concept.
- **Compare With**, a value that is input directly or comes from an external concept. The value is used as the target compared against the source.
- **Compare Now**, a checkbox that specifies a target value that is the current time. This value takes precedence over the derived target value from the Compare With parameter and is compared against the source.
- **Comparator**: =, !=, <, <=, >, and >=
- **Date and Time Type**: One of the following types for the comparison:

The activity has four exits:

- **Success** when comparison succeeds.
- **Not Matched** when comparison fails.

- **Not Found** when unable to locate the values of all external concepts
- **Error** when unable to get the target date value or having invalid date parameter values.

Compare List And Value

Fast Key: CPRL



Matches a target string against a list of source strings. The source strings and the target string can be regular expressions.

The activity has three parameters:

- **Source List**, a list of strings as source stored in an external concept.
- **Compare With**, a target string that is either input directly or comes from an external concept.
- **Comparator**: regex, =

The activity has three exits:

- **Success** when comparison succeeds.
- **Not Found** when unable to retrieve values from specified external concepts.
- **Error** when comparison fails.

Copy Value

Fast Key: COPY



Copies the value from one context field to another context field.

The activity performs the following operations:

1. Verifies Source ContextKey parameter is valid.
2. Verifies Destination ContextKey parameter is valid.
3. Invokes RetrieveContextValue ActionRequest with Source ContextKey.
4. Extracts value from retrieved context key.
5. Stores extracted value in destination context key.
6. Verifies store succeeded.

7. Branches on result.

The activity has two parameters:

- **Source**, the source value from an external concept.
- **Destination**, the value is from an external concept.

The activity has two exits:

- **Success** when copying was successful
- **Error** when unable to retrieve the source value, or unable to convert the source value into a data type that the destination requires, or failed to store the destination value.

Date Time Offset

Fast Key: **DTO**



Applies an offset to a Date, Time or DateTime source value.

The activity has six parameters:

- **Compare Type**: Now (comparing with the current time), Context Location.
- **Source**, a value that can be input directly or come from an external concept.
- **Timezone**, optional, a value that can be input directly or come from an external concept. The value contains a string literal value of one of the Canonical ID values of the Olsen tz database. It is only applicable to date and time values. The date time value may itself contain a Timezone, this parameter allows us to convert it to a time in the requested TZ. If the parameter is absent then the default Timezone of UTC is used.
- **Time Offset**, a value that can be input directly, be an integer literal value, or come from an external concept. The value may be negative.
- **Offset Units**, the value can be input directly, or be an integer literal value, or come from an external concept. The value is a string literal. Possible values are:
- **Destination**, the location of the result, which must be a context path. It represents the location to place the incremented or decremented DateTime or TimeOfDay value. The type of the value put in the result location is the same as the type of the source value.

The activity has two exits:

- **Success** when adjusted successfully
- **Error** when receiving invalid action response, or failed to retrieve any context values, or failed to store the adjusted date time value.

Delete

Fast Key: DEL



Allows a control flow to delete a context field entry.

The activity performs the following operations:

1. Verifies that the Source parameter is valid.
2. Invokes DeleteContextValue ActionRequest.
3. Verify that the Delete succeeded.
4. Branch on result.

The activity has one parameter:

- **Source**, an external concept which specifies the context field to be deleted.

The activity has two exits:

- **Success** when the delete operation succeeded.
- **Error** when unable to retrieve the source value or unable to delete the source value.

End

Fast key: END



Denotes the end of a control flow branch.

The activity has no parameters and no exits.

End Charging Session

Fast Key: ECS



Ends a charging session with the charging data function (CDF)/online charging system (OCS).

The activity has four parameters:

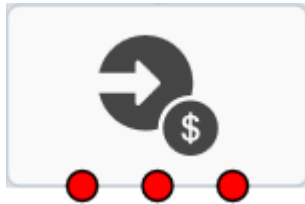
- **Charging Type:** off-line, on-line
- **Request Template,** the value is from an external concept of the charging template type.
- **Answer Template,** the value is from an external concept of the charging template type.
- **Session Name:** Charging Session 1, Charging Session 2, Charging Session 3, Charging Session 4, and Charging Session 5.

The activity has three exits:

- **Success,** when charging ends successfully.
- **No Session,** when there is no charging session in progress.
- **Error,** when charging ends with failure.

Event Charge

Fast Key: EVTC



Sends a charging event to the charging data function (CDF). Both offline charging (Diameter Rf) and online charging (Diameter Ro) are supported.

The activity has three parameters:

- **Charging Type:** On-line, Off-line
- **Request Template,** the value is from an external concept of the charging template type.
- **Answer Template,** the value is from an external concept of the charging template type.

The activity has three exits:

- **Success,** when charging event successful
- **Denied,** when receiving reservation DENIED charging response (on-line charging only)
- **Error,** when receiving NO SESSIOIN, FAILURE, PENDING FAILURE charging response, or failed to retrieve template parameters.

Extract And Store String

Fast Key: EXCT



Extracts a string value via regular expression and stores the result in the context.

The activity has three parameters:

- **Source**, where to extract strings. The value can be from string that is input directly or from an external concept.
- **Comparator**, a value that comes from user input. The input string is a regular expression, which is used to extract a matching string from the source.
- **Destination**, location where the extracted string should be stored. The value is an external concept of String type.

The activity has two exits:

- **Success**, when successfully extracted and stored strings.
- **Error**, when unable to get the configuration of any of the parameters, or unable to retrieve the source value from the specified external concept, or unable to store the list of extracted strings to the destination.

Find and Replace And Store Value

Fast Key: FIND



Finds a given value in a source, replaces it, and stores the updated value into a destination.

The activity has five parameters:

- **Source**, the source string value can be either from a direct user input, or an external concept.
- **Search Value**, the actual string to search in the source, The value is from the user input. The value will not be used as a regular expression during search.
- **Replacement Value**, the replacement string from the user input.
- **Search Option**: All Values, or First Value only.
- **Destination**, where to store the updated source value. The value is an external concept.

The activity has two exits:

- **Success**, when successfully finding, replacing, and storing the value.

- **Error**, when failed to get the configuration of any of the parameters, or failed to retrieve the source value from the specified external concepts, or failed to store the updated value to the destination.

Generate Document and Store

Fast Key: GDOC



Generates a document from a template using a specified content-type and stores it into a destination.

This activity performs the following operations:

1. Gathers the service URL and template
2. Selects the action based on REST/SOAP selection
3. Marshals the response into context
4. Exits branch based on response status

The activity has four parameters:

- **Document Template**, the value is an external concept of the Template type.
- **Language Of User**, the language to use, either input directly or obtained from an external concept. Note that the value must be in the configured languages of the specified document template. Otherwise, the error branch will be taken during the runtime.
- **Destination**, the external concept where the generated document is stored.
- **Content-Type**, the content type, which is an external concept, of the generated document. The value can be used by Send Message Activity when sending outbound messages.

The activity has two exits:

- **Success** when successfully generating and storing the document.
- **Error** when unable to get the configuration of any of the parameters; retrieve the document template from the specified external concept; recognise the specified language; or store the generated document and content type.

Increment Statistic

Fast Key: INCS



Increments the statistic derived from the statistic definition parameter.

The activity performs the following operations:

1. Gathers the Statistic Definition Id.
2. Invokes the Increment Statistic Action to perform the increment.
3. Exits branch based on response status.

The activity has one parameter:

- **Statistic Definition**, specified by an external concept of the Statistic type.

The activity has two exits:

- **Success** when successfully incrementing the statistic.
- **Error** when unable to get a valid parameter configuration or retrieve the statistic definition from the specified external concept.

Load Service Definition

Fast Key: LSD



Loads and runs a derived service retrieved from the UDR. If a service cannot be determined during control-flow design time, but is only determined during run-time, this activity can be used to achieve that.

The activity has one parameter:

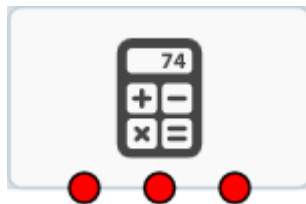
- **Service Definition**, the service to be loaded and run, which is derived from an external concept.

The activity has one exit:

- **Service Not Loaded**, when unable to get the configuration of the Service Definition parameter, or unable to retrieve the value from the specified external concept.

Maths

Fast Key: MATH



Enables basic numerical operations in a control flow. The available operations are: addition, subtraction, multiplication, division and modulus. All operations are performed with two operands, one of which is the contents of a context field (Source).

The second operand can be either a literal value or the contents of another context field (Operand). The calculated result is stored into a context field (Destination) which can differ from the source field.

The operands can be any combination of integer, decimal or string values. All operands are converted into decimal (Double) format prior to the calculation. The result can be stored as an integer, decimal or string value. When storing the result as an integer, the calculated value is rounded down towards zero. See "The Java Language Specification," section 5.1.3, for details.

The activity performs the following operations:

1. Verifies that the following activity parameters are valid:
 - Source ContextKey parameter.
 - Operator parameter.
 - Operand ContextKey parameter.
 - Destination ContextKey parameter.
2. Invokes RetrieveContextValue Action Request with specified context keys.
3. Extracts and validate values from retrieved context keys.
4. Performs the required numerical operation.
5. Invokes StoreContextValue Action Request to store the result into the destination context field.
6. Verifies the store succeeded.
7. Branches on result.

The activity has four parameters:

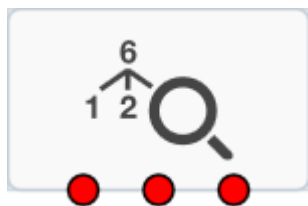
- **Source**, the value is from an external concept and may have type Integer, Float or String.
- **Operator**, Add, Subtract, Multiply, Divide and Modulus.
- **Operand**, the value is from an external concept and may have type Integer, Float or String.
- **Destination**, the value is from an external concept and may have type Integer, Float or String.

The activity has three exits:

- **Success**, when the numerical operation was performed successfully.
- **Not Found**, when unable to locate a context key parameter.
- **Error**, when encountering invalid parameters or values, or the numerical operation fails - for example, when dividing by zero error.

Prefix Tree Lookup

Fast Key: PTLU



Looks up a value in a specified prefix tree.

The activity has six parameters:

- **Prefix Tree**, specifies which prefix tree to use. The value is an external concept of the PrefixTree type.
- **Search Key**, specifies the value to look up in the specified prefix tree. The value can be directly input or come from an external concept.
- **Prefix**, Optional. It specifies a context location to store the matched prefix value.
- **Path**, Optional. It specifies a context location to store the matched path value. The path value is the path from the root element of the tree to the leaf that was matched.
- **Exact Match**. A boolean value that indicates whether or not exact matching should take place. The exact matching is performed when comparing the matched prefix value to the dynamic exit mapping definitions.

The activity can have more than three exits as it can define dynamic exits.

- **Default Match**, when found a value in the prefix tree but the value is not mapped to a dynamic exit.
- **Not Found**, when failed to find the search key in the prefix tree.
- **Error**, when failed to retrieve values from specified external concepts, or failed to store matches prefix value or path.
- **Prefix Exits**. A parameter map that contains dynamic exits mapping to chosen prefix values.

Release

Fast Key: **REL**



Releases a SIP session with a SIP cause value.

The activity has two parameters:

- **Session Endpoint**, specifies which endpoint to release: Initiating Endpoint or Destination Endpoint.
- **Code**, specifies the release cause, for example 480.

The activity has two exits:

- **Disconnected**, when a SIP session has been successfully released.
- **Error**, when parameters were invalid or unable to retrieve parameter values, or the release action is unsuccessful.

Remote Copy

Fast Key: **RCPY**



Copies the value of a context field associated with a different SipApplicationSession to another context field.

The activity performs the following operations:

1. Verifies that Source ContextKey parameter is valid.
2. Verifies that Destination ContextKey parameter is valid.
3. Verifies that Session ID is valid
4. Invokes RetrieveRemoteContext Action Request with Source ContextKey.
5. Extracts value from retrieved context key.
6. Stores the extracted value in destination context key.
7. Verifies store succeeded.
8. Branches on result.

The activity has three parameters:

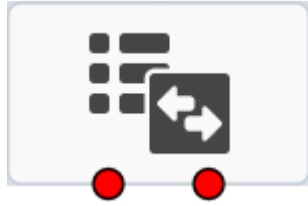
- **Source**, specifies a context location to retrieve the source value.
- **Destination**, specifies a context location to store the retrieved value from the source.
- **Source Session**, specifies a context location to retrieve the SipApplicationSession with which the source is associated.

The activity has two exits:

- **Success**, when it successfully copied the value from a different SipApplicationSession to the specified place.
- **Error**, when invalid parameters were encountered, or unable to retrieve values from the specified external concepts, or unable to store the value into the specified destination.

Retrieve Session List

Fast Key: **RSL**



Retrieves a list of SIP Application Session IDs for the specified session key.

The activity performs the following operations:

1. Verify Source ContextKey parameter is valid.
2. Verify Destination ContextKey parameter is valid.
3. Invoke RetrieveContextValue Action Request with Source ContextKey.
4. Extract value from retrieved context key.
5. Store extracted value in the session as an index key.
6. Verify store succeeded.
7. Branch on result.

The activity has two parameters:

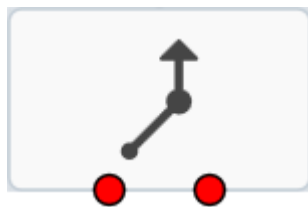
- **Source**, a String type external concept where the search session key is stored.
- **Destination**, a StringArray type external concept where the retrieved IDs are stored.

The activity has two exits:

- **Success**, when successfully retrieved the list of SIP Application IDs and stored in the destination.
- **Error**, when invalid parameters encountered, or failed to retrieve the search session key from the specified external concept, or failed to store the value into the specified destination.

Route

Fast Key: **RTE**



Redirects a call to the original destination endpoint.

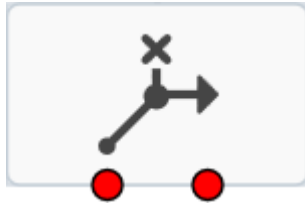
The activity has no parameters.

The activity has two exits:

- **Success**, when the call is successfully routed.
- **Error**, when failed to route the call to the original destination endpoint.

Route Changed

Fast Key: **RTEC**



Redirects a call to the destination address.

The activity has one parameter:

- **Destination Endpoint**, where to route the call. The value can be directly input, or come from an external concept.

The activity has two exits:

- **Success**, when successfully route the call to the specified destination endpoint.
- **Error**, when an invalid parameter encountered, or unable to retrieve the destination endpoint from the specified external concept, or unable to route to the specified destination endpoint.

Run Control Flow

Fast Key: **RCF**



Runs a control flow. The control flow is selected during control flow design time where only application control flows will be listed. The message control flows, such as the 'SIP INVITE' control flow, will not be available.

The activity has one parameter:

- **Control Flow**, the value is from a filtered drop-down list when the designer types. Hint: Typing a space will show all available control flows.

The activity has no default exits and more than one dynamic exit.

- **Dynamic exits**, of which there can be more than one. These are derived from the configured Exit Activities in the selected control flow. If the selected control flow is changed, the number of dynamic exits will be updated on the Run Control Flow Activity and additional exits may need to be connected.

Run Service Definition

Fast Key: **RSD**



Runs a control flow that is associated with a specified service. The service and the control flow are selected during control flow design time.

The activity has two parameters:

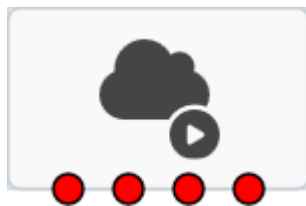
- **Service**, which is chosen from a drop-down list.
- **Control Flow**, which is chosen from a filtered drop-down list when the designer types. Hint: Typing a space will show all available control flows.

The activity has no default exits and more than one dynamic exit.

- **Dynamic exits**, which can be more than one. These are derived from the configured Exit Activities in the selected control flow. If the selected control flow is changed, the number of dynamic exits will be updated on the Run Service Definition Activity and additional exits may need to be connected.

Run Web Service

Fast Key: **RWS**



Runs a web service, either SOAP or REST.

The activity performs the following operations:

1. Gathers the service URL and template
2. Selects the action based on REST/SOAP selection
3. Marshals the response into context.
4. Exits branch based on response status.

The activity has five parameters:

- **Web Service Type**: SOAP or REST
- **Web Service Template**, the value is from an external concept of Template type. Then the designer needs to specify external concepts for each variable used in the template.
- **Language Of User**, which language to use, which can be directly input or come from an external concept. Note that the value must be in the configured languages of the specified template. Otherwise, the error branch will be taken during the runtime.
- **Web Service Response**, an external concept where the web service response will be stored.

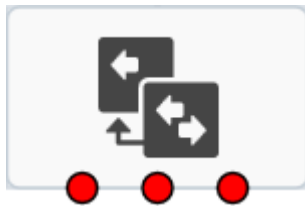
- **Message Timeout**, an integer value for setting up the timeout

The activity has four exits:

- **Success**, the client successfully executes the specified web service and receives a SUCCESS response.
- **Error (Response)**, the client successfully executes the specified web service, but receives a FAILURE response from the server.
- **Failure (Transient)**, a failure caused by temporary conditions, such as a timeout waiting for the response. The request may work if tried again.
- **Failure (Permanent)**, a failure that indicates there is no point trying the request again. This could be because the request couldn't be sent due to:
 - invalid parameters
 - failed to retrieve the context keys used by the specified template
 - failed to retrieve the template content body or to find the URL in the template content body
 - processed a request of an unknown service type.
 - encountered invalid operations for SOAP or invalid method for REST Or a response was received from the server, but couldn't be parsed (e.g. invalid JSON or XML).

Send Inter-Session Event

Fast Key: SISE



Allows an event to be sent from one SIP Application Session to another. The activity performs the following operations:

1. Verifies that the following activity parameters are valid:
 - Key
 - Event Name
2. Invokes RetrieveContextValue Action Request with specified context keys.
3. Extracts and validate values from retrieved context keys.
4. Invokes SendInterSessionEvent Action Request.
5. Validates the SendInterSessionEvent response.
6. Branches on result.

The activity has the following parameters:

- **Session Key**, a String type external concept where the session key is stored.
- **Event Name**, a String type external concept that identifies the type of event to be sent to the remote session.

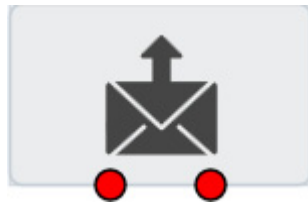
- **Event Data**, a Map type external concept that identifies where the event data is held.

The activity has three exits:

- **Success**, when the event notifications were performed successfully.
- **Not Found**, when unable to locate a context key parameter, or when no SIP Application Sessions matched the session id.
- **Error**, when encountering invalid parameters or values.

Send Message

Fast Key: **SEND**



Sends a SIP MESSAGE towards the Access Transfer Control Function (ATCF) containing SRVCC-related information. It is normally sent following receipt of a third-party registration from the S-CSCF, which indicates support for the ATCF role.

The activity has three parameters:

- **SIP Request-URI**, the value can be from a direct input or from an external concept.
- **Content to Send**, the value is from an external concept.
- **Content Type**, the value can be directly input or come from an external concept.

The activity has two exits:

- **Success**, when on receipt of 200 OK.
- **Error**, when invalid parameters encountered, for instance request-URI is empty; or content is provided without content type (note: empty content is allowed); or content type is provided but content is empty; or response received other than 200 OK.

Send Subscription Notify

Fast Key: **SNOT**



Sends a notification on an existing subscription notifier session, previously started with a Start Notifier Session activity.

The activity performs the following operations:

1. Verifies parameters. All must be present. If state is "Terminated" then reason must be set to something other than "None".
2. Fetches context parameters from context map. The content parameter will always be held in context map.
3. Verifies that all parameters have been obtained from context and activity.
4. Triggers the notification by sending SendNotifyParameters in a resume request to the underlying SipSessionAction. The SipSessionAction will:
 - a. Build the NOTIFY request with the content and headers collected by the activity.
 - b. Send the NOTIFY request to the subscriber.
 - c. Wait for the NOTIFY response from the subscriber.
 - d. Send an appropriate SendNotifyResponse back to the activity.
5. Waits for SendNotifyResponse from the SipSessionAction.
6. Branches on result.

The activity has five parameters:

- **Session Endpoint**, the alias of the SIP session to send the notification on.
- **Subscription State**, one of "Active", "Terminated" or "Pending". Refer to RFC 6665 for definitions.
- **Termination Reason**, one of "None" (to be used when state is not "Terminated"), "Deactivated", "Probation", "Rejected", "Timeout", "Giveup", "No Resource" or "Invariant". Refer to RFC 6665 for definitions.
- **Notify Content Type**, the notification Content-Type header. A string context field or literal value, for example "application/dialog-info+xml".
- **Notify Content**, the notification body content. A string context field. It is envisaged the existing Generate Document Activity will be used to build up the content from a template prior to invoking this activity in a similar manner to how the Send Message activity operates.

The activity has two exits:

- **Success**, notification sent successfully.
- **Error**, the notification failed to send, parameter errors, context fields missing, and so on.

Start

Fast Key: **ST**



This activity denotes the beginning of a Control Flow.

The activity has three parameters:

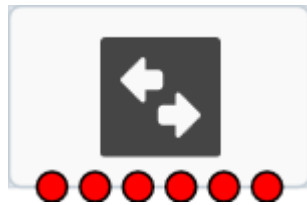
- **Loop Limit:** an integer value from the user input to restrict the number of times the activity can be entered. Default: 100.
- **Loop Limit External:** an integer value from the user input to restrict the number times the activity can be entered between network interactions. Default to 100.
- **Loops Allowed.** a Boolean value from the user input. Default: OFF.

The activity has one exit:

- **Success,** control flow started.

Start Back to Back Session

Fast Key: SBBS



Starts a back to back session between initiating endpoint and destination endpoint. It initiates a call to the destination address using the Back to Back User Agent (B2BUA) Session Action, which then acts as a B2BUA until the call ends. The action returns a response containing the ID of the event that caused the call to finish, for example Busy, No Answer, and so on. The exit appropriate to the failure reason is taken if defined; otherwise, the Session Ended exit is taken.

The activity has four parameters:

- **Session Endpoint,** where you can specify the session alias for both Initiating Endpoint and Terminating Endpoint.
- **Destination Endpoint,** the value can be directly input, or come from an external concept. It is the destination address for the INVITE request.
- **Timeout,** the time to wait for the response to the INVITE, an integer value that can be directly input or come from an external concept.
- **Forwarded,** a boolean value, true if SIP status code 181 should be sent for call being forwarded.

The activity has six exits:

- **Ringin****g,** when the RINGING response has been received from the started B2BUA session
- **Busy,** when the BUSY response has been received from the started B2BUA session
- **Not Reachable,** when the NOT REACHABLE response has been received from the started B2BUA session
- **Redirected,** when the REDIRECTED response has been received from the started B2BUA session
- **Initiating Endpoint Ended,** when the PARTY A HANGUP response has been received from the started B2BUA session
- **Error,** when invalid parameters encountered, or unable to retrieve values from the specified external concept, or the activity received the ERROR response from the started B2BUA session

Start Charging Session

Fast Key: SCS



Starts a charging session with the charging data function (CDF)/online charging system (OCS).

The activity has four parameters;

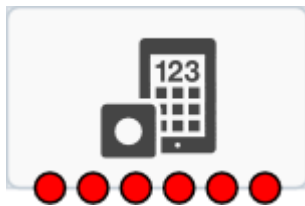
- **Charging Type:** On-line or Off-line
- **Request Template,** the value is from an external concept of the charging template type.
- **Answer Template,** the value is from an external concept of the charging template type.
- **Session Name:** Charging Session 1, Charging Session 2, Charging Session 3, Charging Session 4, or Charging Session 5.

The activity has four exits:

- **Success,** when charging is updated successfully.
- **Denied,** when reservation is denied (on-line charging only)
- **Error,** when encountered invalid parameters or failed to retrieve values from the specified external contexts, or received NO SESSION, FAILURE, PENDING FAILURE charging response
- **No Charge,** when charging is not applicable.

Start Collecting Digits

Fast Key: SCOL



Starts collecting digits on the media server.

The activity has seven parameters:

- **Session Endpoint:** Initiating Endpoint, Destination Endpoint, or Event Source.
- **Language Of User,** which language to use, the value can be from a direct input or from an external concept. Note that the value must be in the configured languages of the specified prompt template. Otherwise, the error branch will be taken during the runtime.
- **Number Of Digits,** an integer value from the user input

- **Stop Digit**, any string from the user input indicating the end of collecting
- **Store Digits To**, a context location where the collected digits will be stored.
- **Prompt**, the prompt media template resource, the value is from an external concept.
- **Prompt Parameters**, a list of optional parameters for playing prompt, including Max Duration, Repeat Count, Repeat Interval, and Queue Announcement.
- **Collect Parameters**, a list of optional parameters for collecting digits, including Max Duration, Initial Digit Timeout, Inter-Digit Timeout, Buffer Digit Collection, and Barge In.

The activity has six exits:

- **Success** when successfully collected and stored the digits from the user.
- **Error (Server)**, when received ERROR SERVER CONNECTION media response.
- **Error (Media)**, when received ERROR NOT FOUND media response
- **Error (Timeout)**, when received ERROR TIMEOUT media response
- **Error (Not Supported)**, when received ERROR NOT SUPPORTED media response
- **Error**, when encountered invalid parameters or failed to retrieve values from external concepts, or received other ERROR media response.

Start Conference Session

Fast Key: CONF



Starts a conference session.

The activity has four parameters:

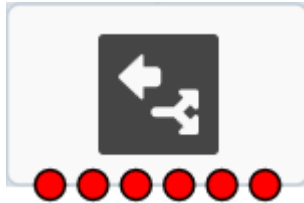
- **Creator Endpoint**: Initiating Endpoint only
- **Conference Endpoint**, the value can be from a direct input or from an external concept.
- **Media Server**, the value is an external concept of the template type.
- **Media Parameters**, a list of optional media mixer parameters, including Max Ports and Max Active Inputs.

The activity has three exits:

- **Created**, when successfully created a conference session.
- **Creator Hang Up**, when the creator hangs up the call.
- **Error**, when encountered invalid parameters or failed to retrieve values from the specified external concepts, or received the ERROR response.

Start Forked Session

Fast Key: SFOS



Starts a forked session between initiating endpoint and a set of destination endpoints defined in the activity. It initiates calls to the destination address using the ForkedSessionProcessor via the SipSessionAction. That action acts as an intermediary SipApplicationSession between the initiating endpoint and the actual terminating endpoint sessions.

The activity has six parameters:

- **Initiating Endpoint**, where you can specify the session alias for the initiating caller.
- **Destination Endpoint**, where you can specify the session alias to apply to the caller that answers.
- **Group Identity List**, containing a list of all destination addressees to fork.
- **User Type**, single user or multiple user.
- **Timeout**, the time to wait for the response to the INVITE, an integer value can be directly input or come from an external concept.
- **Fork Type**, indicating if the fork should be sequential or parallel.

The activity has five exits:

- **Ringin****g**, when received the first RINGING response from one of the forked session
- **RBusy**, when received the BUSY response from all forked session
- **RNot Reachable**, when received the NOT REACHABLE response from all forked sessions
- **RInitiating Endpoint Ended**, when received the PARTY A HANGUP response from the successful forked session.
- **RError**, when encountered invalid parameters or failed to retrieve values from the specified external concept, or received the ERROR response from the successful forked session

Start Subscription Notifier Session

Fast Key: SNTS



Starts ECAS running as a SIP Subscription Notifier as a result of receiving a SIP SUBSCRIBE request. The activity performs the following operations:

1. Verifies parameters. Minimum Expiry must be > 0 and $<$ Maximum Expiry. **Note:** There are currently no context parameters to fetch so there is no RetrieveContext step.
2. Switches the underlying SipSessionAction to NOTIFIER mode by sending NotifierParameters in a resume request. The SipSessionAction will:
 - a. Validate the SUBSCRIBE request against the specified expiry range and reject or cap the expiry.
 - b. Respond to the SUBSCRIBE request.
 - c. Send an appropriate NotifierResponse back to the activity.
3. Waits for NotifierResponse from the SipSessionAction.
4. Branches on result.

The activity has three parameters:

- **Session Endpoint**, the alias of the SIP session which received the SUBSCRIBE request.
- **Minimum Expiry**, the minimum expiry time to accept (seconds). A SUBSCRIBE with $0 < \text{Expires} < \text{MinExpiry}$ will be rejected with a 423 Interval Too Brief error.
- **Maximum Expiry**, the maximum expiry time to accept (seconds). A SUBSCRIBE with $\text{Expires} > \text{MaxExpiry}$ will be capped so $\text{Expires} = \text{MaxExpiry}$.

The activity has three exits:

- **Success**, the subscription is ok and accepted.
- **Terminated**, the special case where initial SUBSCRIBE Expires = 0. The subscription is ok but the flow should build and send a single notification with state Terminated then exit.
- **Error**, the subscription was rejected, session setup failed, parameter errors or others

Start Playing Media

Fast Key: **SPLY**



Starts playing an announcement on the media server.

The activity has four parameters:

- **Session Endpoint:** Initiating Endpoint, Destination Endpoint, Conference Endpoint, or Event Source.
- **Language of User:** which language to use, the value can be directly input or come from an external concept. Note that the value must be in the configured languages

of the specified document template. Otherwise, the error branch will be taken during the runtime.

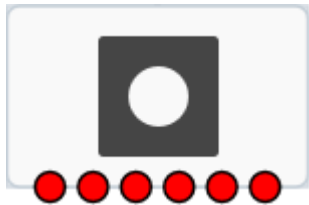
- **Media Resource**, which media to play, the value is from an external concept.
- **Media Parameters**, a list of optional parameters for playing the media, including Max Duration, Repeat Count, Repeat Interval, and Queue Announcement.

The activity has six exits:

- **Success** when successfully played the media.
- **Error (Server)**, when received ERROR SERVER CONNECTION media response.
- **Error (Media)**, when received ERROR NOT FOUND media response
- **Error (Timeout)**, when received ERROR TIMEOUT media response
- **Error (Not Supported)**, when received ERROR NOT SUPPORTED media response
- **Error**, when encountered invalid parameters or failed to retrieve values from external concepts, or received other ERROR media response.

Start Recording Message

Fast Key: SREC



Plays a prompt, then starts recording a message on the media server.

The activity has seven parameters:

- **Session Endpoint**: Initiating Endpoint, Destination Endpoint, Conference Endpoint, or Event Source.
- **Language of User**: which language to use, the value can be from a direct input or from an external concept. Note that the value must be in the configured languages of the specified document template. Otherwise, the error branch will be taken during the runtime.
- **Stop Digit**: any string from the user input indicating the end of recording.
- **Prompt**, the prompt media template resource, the value is from an external concept.
- **Store Recording To**, a context location where the recorded message will be stored.
- **Prompt Parameters**, a list of optional parameters for playing prompt, including:
 - Max Duration
 - Repeat Interval
 - Queue Announcement
- **Record Parameters**, a list of optional parameters for recording message, including:
 - Max Duration
 - Mix Duration

- Append Recording
- Start Beep
- Start in Paused mode
- Stop on Silence
- Initial Timeout
- Final Timeout.

The activity has six exits:

- **Success** when successfully recorded the media message.
- **Error (Server)**, when received ERROR SERVER CONNECTION media response.
- **Error (Media)**, when received ERROR NOT FOUND media response.
- **Error (Timeout)**, when received ERROR TIMEOUT media response.
- **Error (Not Supported)**, when received ERROR NOT SUPPORTED media response.
- **Error**, when encountered invalid parameters or failed to retrieve values from external concepts, or received other ERROR media response.

Start Timer

Fast Key: **STMR**



Starts a timer for a configurable duration. A Timer Expiry event is raised when the duration of the timer has elapsed. The Wait For Event activity is used to trap a Timer Expiry event. Up to four timers can be created.

The activity has two parameters:

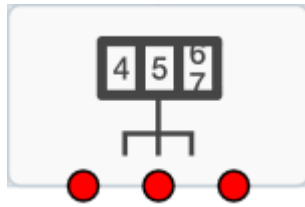
- **Duration**, the duration of the timer in milliseconds.
- **Timer**, specifies the timer to be created.

The activity has two exits:

- **Success**, when the timer has been successfully started.
- **Error**, upon failure to retrieve the duration value from the specified external concept or the retrieved value is invalid.

Statistic Branching

Fast Key: **STB**



Branches on statistic values. The target values is either directly input or comes from an external concept. The activity performs the following operations:

1. Verifies parameters are valid.
2. Invokes Statistics Action Request with specified definition ID.
3. Verifies successful query.
4. Compares retrieved value with specified comparison string.
5. Branches on comparison result.

The activity has four parameters:

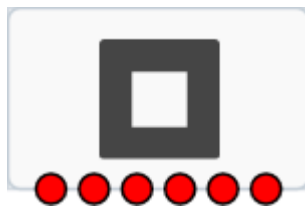
- **Statistic Definition**, the value is from an external concept of the Statistic type.
- **Start Time Offset**, an integer value from the user input.
- **Compare With**, the target, its value is from either a direct input or another external concept.
- **Comparator**: =, !=, <, <=, >, >=.

The activity has three exits:

- **Success**, when the comparison result is expected.
- **Not Found**, when unable to retrieve the compared target, or the comparison result is unexpected.
- **Error**, when encountered invalid parameters or failed to retrieve values from the specified external concepts, or encountered an invalid comparator.

Stop Media

Fast Key: **STPM**



Stops all media on the media server.

The activity has one parameter:

- **Session Endpoint**: Initiating Endpoint, Destination Endpoint, Conference Endpoint, or Event Source.

The activity has six exits:

- **Success** when successfully played the media.
- **Error (Server)**, when received ERROR SERVER CONNECTION media response.

- **Error (Media)**, when received ERROR NOT FOUND media response.
- **Error (Timeout)**, when received ERROR TIMEOUT media response.
- **Error (Not Supported)**, when received ERROR NOT SUPPORTED media response.
- **Error**, when encountered invalid parameters or failed to retrieve values from external concepts, or received other ERROR media response.

Store

Fast Key: **STRE**



Stores a value in a context field.

The activity has two parameters:

- **Source**, the value to store from the user input.
- **Destination**, a context location where the input value will be stored.

The activity has two exits:

- **Success**, when successfully stored the input value into the specified context location.
- **Error**, when encountered invalid parameters or failed to store the value.

Store Session Key

Fast Key: **STSK**



Copies the value from a context field into the current session as an index key.

The activity performs the following operations:

1. Verifies that the Source ContextKey parameter is valid.
2. Invokes RetrieveContextValue Action Request with Source ContextKey.
3. Extracts the value from the retrieved context key.
4. Stores the extracted value in the session as an index key.
5. Verifies that the store succeeded.
6. Branches on the result.

The activity has one parameter:

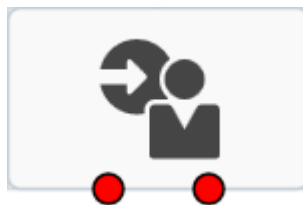
- **Source**, the value is from an external concept.

The activity has two exits:

- **Success**, when it has successfully extracted and stored the value in the session.
- **Error**, when an invalid parameter is encountered, when unable to retrieve the value from the specified source location, or when unable to store the value into the session.

Subscribe Event Package

Fast Key: **SEP**



Sends a SIP SUBSCRIBE request. It is unusual in that it creates an instance of `SipSessionAction`. `SipSessionActions` are normally created by the `SessionControlFramework` in response to messages received by the `SipServlet`, for example an incoming SIP INVITE message.

The activity performs the following operations:

1. Fetches endpoint parameter from activity.
2. Creates a new `SipSessionAction` to handle the subscription
3. Verifies that we have a new `SipSessionAction`.
4. Verifies other activity parameters. All must be present.
5. Fetches context parameters from the context map.
6. Verifies that the parameter values have been successfully retrieved from the context or the activity.
7. Triggers the SIP SUBSCRIBE request by sending `SubscriberParameters` in a resume request to the `SipSessionAction` that was created earlier. The `SipSessionAction` will:
 - a. Build the SIP SUBSCRIBE request with the content and headers collected by the activity.
 - b. Send the SIP SUBSCRIBE request.
 - c. Wait for the SIP SUBSCRIBE response from the notifier.
 - d. Send an appropriate `SubscriberResponse` back to the activity
8. Branches on result.

The activity has six parameters:

- **Session Endpoint**, the alias of the SIP session to send the notification on. This will usually be 'SUB'.
- **Subscribe Content Type**, the subscription Content-Type header. A string context field or literal value, for example "application/tcap-component-portion+xml"

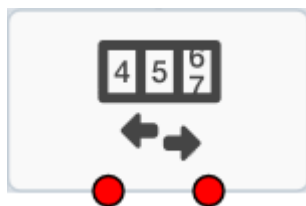
- **Subscription Content**, the subscription body content. A string context field. It is envisaged that the existing Generate Document Activity will be used to build u the content from a template prior to invoking this activity in a similar manner to how the Send Message Activity operates.
- **Event**, The name of the Event Package for the subscription. A string context field or literal value.
- **Request URI**, the uri that the SIP SUBSCRIBE request will be sent to. A string context field or literal value.
- **Expiry**, The subscription Expiry header. An integer context field or literal value. Must not be negative.

The activity has two exits:

- **Success**, SIP the SUBSCRIBE request was successful.
- **Error**, SIP the SUBSCRIBE request failed or was rejected.

Sync Statistic

Fast Key: **SYNS**



Syncs the statistic cache with the server. The activity performs the following operations:

1. Invokes the StatisticsRequestAction to perform a sync.
2. Invokes the TelemetryRequestAction to perform a sync.
3. Exits branch based on response status.

The activity has no parameters.

The activity has two exits:

- **Success**, when successfully sync the statistic cache with the server.
- **Error**, when failed to sync the statistic cache with the server.

Telemetry

Fast Key: **TELM**



Starts or stops a telemetry record.

The activity performs the following operations:

1. Gathers the Telemetry Record Id.
2. Invokes the Telemetry Action to perform the start/stop.
3. Exits branch based on response status.

The activity has two parameters:

- **Record Option:** Start or Stop.
- **Telemetry Record,** the value is from an external concept of the Telemetry type.

The activity has two exits:

- **Success,** when successfully started or stopped the specified telemetry.
- **Error,** when encountered invalid parameters including invalid record options, or failed to retrieve the telemetry record name from the specified external concept, or failed to start or stop the specified telemetry.

Translate and Store Value

Fast Key: TSST



Performs the following operations:

1. Obtain the context locations for 'Source Location', 'Destination Location' and the 'Source List Location'.
2. Read the 'Source Location' value
3. Translate if in the 'Source List Location'
4. Write the 'Destination Location' value

The activity has three parameters:

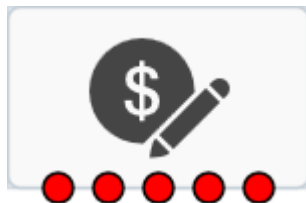
- **Source List,** the value is from an external concept of the Map type.
- **Source,** the value is from an external concept.
- **Destination,** a context location where the translated result will be stored.

The activity has three exits:

- **Success,** when translated and stored the result into the destination location successfully.
- **Not Found,** when failed to retrieve values from the specified Source location or failed to find the corresponding translate value in the Source List location.
- **Error,** when encountered invalid parameters, or failed to store the translated result into the destination.

Update Charging Session

Fast Key: UCS



Sends an update to an already established charging session with the charging data function (CDF).

The activity has four parameters:

- **Charging Type:** On-line or Off-line.
- **Request Template,** the value is from an external concept of the charging template type.
- **Answer Template,** the value is from an external concept of the charging template type.
- **Session Name:** Charging Session 1, Charging Session 2, Charging Session 3, Charging Session 4, or Charging Session 5.

The activity has five exits:

- **Success,** when charging is updated successful.
- **Denied,** when reservation is denied (on-line charging only).
- **No Session,** when there is no charging session in progress.
- **Error,** when encountering invalid parameters or unable to retrieve values from the specified external contexts, or received charging response of NO SESSION, FAILURE, or PENDING FAILURE.
- **No Charge,** when charging is not applicable.

Update Profile

Fast Key: UPDP



Updates a subscriber's profile data in the UDR.

The activity performs the following operations:

1. Invokes an Update Profile action.
2. Branches on result.

The activity has no parameters.

The activity has three exits:

- **Success Response**, when received a SUCCESS response.
- **No data To Update**, when received NO DATA TO UPDATE response.
- **Activity Error**, when received an ERROR response.

Wait For Event

Fast Key: WFEV



Branches on an event result. The activity performs the following operations:

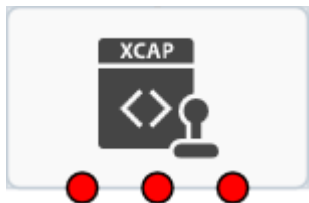
1. Validates and converts the activity parameters to internal event type and source objects.
2. Initiates a Wait For Event Action request to perform the event wait / claim.
3. Processes the Wait For Event Action response.
4. Exits according to the response result.

The activity has no parameters.

The activity has two fixed exits and more than one dynamic event exits.

- **Session Ended**, when the SESSION ENDED response has been received.
- **Error**, when the Wait for Event action is unsuccessful.
- **Event-based exits**, which can be more than one. It will be taken when the specified event in the given session occurs successfully.

XCAP Authorize



The activity performs the following operations:

1. Verifies that ContextKey parameters are valid.
2. Gathers the XML, HTTP method and node selector from context on which to operate.
3. Authorizes the method based on source XML, HTTP method and node selector.
4. If there is an authorization failure, stores the error in the error location.

The activity has five parameters:

- **Source**, an external concept.

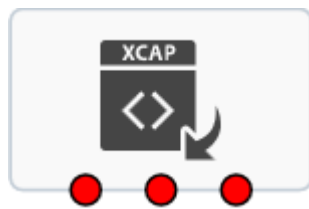
- **AUID**, an external concept, which can be a context literal value.
- **HTTP method**, an external concept.
- **Node selector**, an external concept.
- **Error location**, an external concept.

The activity has three exits:

- **Success** when operation call successful.
- **Error** from XCAP operation. The XML Error Response will be set.
- **Error** when failed to locate a context parameter.

XCAP Fetch And Store Document

Fast Key: XFTC



Loads subscriber data as XCAP XML into the context Map.

The activity performs the following operations:

1. Verifies that ContextKey parameters are valid which specify the "/UserData/Service/View/Field" xpath.
2. Gathers JSON from source (HSS presumably).
3. Invokes the pluggable XCAP AU fetch action.
4. Stores the result in either the destination or error location.

The activity has four parameters:

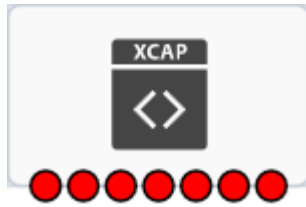
- **Source**, an external concept.
- **AU ID** of the pluggable action. Its value is either directly input or comes from another external concept.
- **Destination location**, an external concept.
- **Error location**, an external concept.

The activity has three exits:

- **Success** when operation call successful.
- **Unable** to find subscriber data to get.
- **Other Error**.

XCAP Process Document

Fast Key: XPRC



Processes the XCAP inbound configuration and the Subscriber data as XML documents.

The activity performs the following operations:

1. Verifies that ContextKey parameters are valid, which specify the "/UserData/Service/View/Field" xpath.
2. Gathers the XML from context on which to operate.
3. Processes the configuration instruction.
4. Stores the result in either the destination, or error location.

The activity has nine parameters:

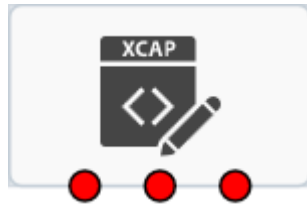
- **Source**, an external concept.
- **AUID**, of the pluggable action; its value is either directly input or comes from another external concept.
- **HTTP method**, an external concept.
- **Node selector**, an external concept.
- **Namespaces**, an external concept.
- **Update location**, an external concept.
- **Destination location**, an external concept.
- **Response location**, an external concept.
- **Update keys destination**, an external concept.

The activity has seven exits:

- **Success** when operation call successful.
- **Success Creation**, when operation call creation successful.
- **Conflict Error** from XCAP operation. (XML Error Response will be filled in.)
- **Missing Element Error** from XCAP operation.
- **Bad Request** from XCAP operation.
- **Unauthorized** from XCAP operation.
- **Error** when failed to locate a context parameter.

XCAP Update Document

Fast Key: XUPD



Updates subscriber data from XCAP XML into the context Map.

The activity performs the following operations:

1. Verifies that ContextKey parameters are valid (which specify the "/UserData/Service/View/Field" xpath).
2. Gathers JSON from source (HSS presumably).
3. Invokes the pluggable XCAP AU update action.
4. Stores the result in either the destination, or error location.

The activity has four parameters:

- **Source**, an external concept.
- **AU ID** of the pluggable action, its value either is directly input or comes from another external concept.
- **Destination location**, an external concept.
- **Error location**, an external concept.

The activity has three exits:

- **Success** when operation call is successful.
- **Conflict Error** from XCAP operation (probably Validation failure).Fills in XML Error Response.
- **Other Error**.

External References for Supported Interactions

This appendix describes scenarios and corresponding call-session flows that OCECAS supports.

Table C-1 Redirection - Single PMN

Scenario	Reference
Redirection to alternative DE: Call Forward Unconditional	3GPP TS 24.604 A.1.1
Redirection to alternative DE: Call Forward Not Logged In	3GPP TS 24.604 A.1.5
Redirection to alternative DE: Call Forward on Busy	3GPP TS 24.604 A.1.4
Redirection to alternative DE: Call Forward on No Answer	3GPP TS 24.604 A.1.3
Redirection to alternative DE: Call Forward on Not Reachable	3GPP TS 24.604 A.1.5

Table C-2 VoLTE - Single PMN

Scenario	Reference to Call Flows
VoLTE UE Attachment and IMS Registration,	GSMA FCM.01 3.2.2 & 3GPP TS 24.229 5.7.1.1
VoLTE UE Initiated Detach and IMS Deregistration,	GSMA FCM.01 3.2.2
Basic VoLTE UE to VoLTE UE Voice Call Establishment, Originating Side	GSMA FCM.01 3.2.3
Basic VoLTE UE to VoLTE UE Voice Call Establishment - Terminating Side	GSMA FCM.01 3.2.4
Basic VoLTE UE to VoLTE UE Voice Call Clearing - Initiated	GSMA FCM.01 3.2.5
Basic VoLTE UE to VoLTE UE Voice Call Clearing - Received	GSMA FCM.01 3.2.6
Basic VoLTE UE to VoLTE UE Multimedia (Voice/Video) Call Establishment, Originating Side	GSMA FCM.01 3.2.7
Basic VoLTE UE to VoLTE UE Multimedia (Voice/Video) Call Establishment, Terminating Side	GSMA FCM.01 3.2.8
Basic VoLTE UE to VoLTE UE Multimedia (Voice/Video) Call Clearing - Initiated	GSMA FCM.01 3.2.9
Basic VoLTE UE to VoLTE UE Multimedia (Voice/Video) Call Clearing - Received	GSMA FCM.01 3.2.10
Basic VoLTE UE to VoLTE UE - Adding a video media stream - Originating Side	GSMA FCM.01 3.2.11

Table C-2 (Cont.) VoLTE - Single PMN

Scenario	Reference to Call Flows
Basic VoLTE UE to VoLTE UE - Adding a video media stream - Terminating Side	GSMA FCM.01 3.2.12
Basic VoLTE UE to VoLTE UE - Removing a video media stream - Originating Side	GSMA FCM.01 3.2.13
Basic VoLTE UE to VoLTE UE - Removing a video media stream - Terminating Side	GSMA FCM.01 3.2.14

Table C-3 VoLTE- Single PMN (Failures)

Scenario	Reference to Call Flows
No Answer from DE	RFC 3665
Busy indicator from DE	RFC 3665
No route to DE - IE detaches before session establishment with DE - Deregistration during call	RFC 3665

Table C-4 VoLTE - Roaming

Scenario	Reference to Call Flows
Roaming VoLTE UE Attachment and IMS Registration	GSMA FCM.01 5.2.1
Roaming VoLTE UE Initiated Detach and IMS Deregistration	GSMA FCM.01 5.2.2
Roaming VoLTE UE to VoLTE Voice Call Establishment, Originating Side	GSMA FCM.01 5.2.3
Roaming VoLTE UE to VoLTE UE Voice Call Establishment, Terminating Side	GSMA FCM.01 5.2.4
Roaming VoLTE UE to VoLTE UE Voice Call Clearing - Initiated	GSMA FCM.01 5.2.5
Roaming VoLTE UE to VoLTE Voice Call Clearing - Received	GSMA FCM.01 5.2.6
Roaming VoLTE UE to VoLTE UE Multimedia (Voice/Video) Call Establishment, Originating Side	GSMA FCM.01 5.2.7
Roaming VoLTE UE to VoLTE UE Multimedia (Voice/Video) Call Establishment, Terminating Side	GSMA FCM.01 5.2.8
Roaming VoLTE UE to VoLTE UE Multimedia (Voice/Video) Call Clearing - Initiated	GSMA FCM.01 5.2.9
Roaming VoLTE UE to VoLTE UE Multimedia (Voice/Video) Call Clearing - Received	GSMA FCM.01 5.2.10
Roaming VoLTE UE to VoLTE, Adding a video media stream	See following list
- Video Stream added by the roaming UE	GSMA FCM.01 5.2.11
- Video Stream added by the other party	GSMA FCM.01 5.2.11
Roaming VoLTE UE to VoLTE, Removing a video media stream	See following list
- Video Stream removed by roaming UE	GSMA FCM.01 5.2.12
- Video Stream removed by other party	GSMA FCM.01 5.2.12

Table C-5 SRVCC - Enhanced

Scenario	Reference To Call Flows
eSRVCC Attachment and IMS Registration	GSMA FCM.01 7.4.1 & 3GPP TS 24.237 A.3.3
UE Voice Call Establishment - Originating Side	GSMA FCM.01 7.4.2 & 3GPP TS 24.237 A.4.2
UE Voice Call Establishment - Terminating Side	GSMA FCM.01 7.4.3
PS-CS SRVCC - PS-CS Handover Cancellation - PS-CS Handover Terminating Leg	GSMA FCM.01 7.4.4 & 3GPP TS 24.237 A.6.1
PS-CS SRVCC enhancements using ATCF and with media anchored	3GPP TS 24.237 A.18.3
PS-CS SRVCC enhancements using ATCF and session traverses IBCF	GSMA FCM.01 7.4.4 & 3GPP TS 24.237 A.18.6
PS-CS SRVCC for active call with speech and video using vSRVCC procedure	3GPP TS 24.237 A.19.2

Table C-6 Terminating Access Domain Selection

Scenario	Reference
Call termination in IM CN	3GPP TS 24.292 10.4.3
Call termination to CS network via MSC Server enhanced for ICS	3GPP TS 24.292 10.4.5
Call termination over CS	3GPP TS 24.292 10.4.7

Table C-7 Supplementary Services

Scenario	Reference to Call Flows or Specification
Originating Identification Presentation	See following list
- no subscription tests	3GPP TS 24.607 4.5.2.9
- Privacy "id" tests	3GPP TS 24.607 4.5.2.9
- Privacy "header" tests	3GPP TS 24.607 4.5.2.9
- Privacy "user" tests	3GPP TS 24.607 4.5.2.9
Originating Identification Restriction	See following list
- "permanent mode" subscription tests	3GPP TS 24.607 4.5.2.4
- "temporary mode" subscription tests	3GPP TS 24.607 4.5.2.4
- no subscription tests	3GPP TS 24.607 4.5.2.4
Terminating Identification Presentation	See following list
- no subscription tests	3GPP TS 24.608 4.5.2.4
- subscription tests	3GPP TS 24.608 4.5.2.4
Terminating Identification Restriction	See following list
- "permanent mode" subscription tests	3GPP TS 24.608 4.5.2.9

Table C-7 (Cont.) Supplementary Services

Scenario	Reference to Call Flows or Specification
- "temporary mode - default restricted" subscription tests	3GPP TS 24.608 4.5.2.9
- "temporary mode - default not restricted" subscription tests	3GPP TS 24.608 4.5.2.9
- no subscription tests	3GPP TS 24.608 4.5.2.9
Communication Forwarding	See following list
- Unconditional (CFU)	3GPP TS 24.604 4.5.2.6 A.1.1
- on Not Logged-in (CFNL)	3GPP TS 24.604 4.5.2.6 A.1.5
- on Busy (CFB)	3GPP TS 24.604 4.5.2.6 A.1.4
- on Not reachable (CFNRc)	3GPP TS 24.604 4.5.2.6
- on No Reply (CFNR)	3GPP TS 24.604 4.5.2.6 A.1.3
- media condition	3GPP TS 24.604 4.9.1.3
- NoReplyTimer	3GPP TS 24.604 4.9.1.1A
Communication Deflection	3GPP TS 24.604 A.1.2
Incoming Call Barring (ICB)	See following list
- barring of all incoming calls	3GPP TS 24.611 4.5.2.6.1
- barring of incoming calls when roaming	3GPP TS 24.611 4.5.2.6.1
- barring of anonymous calls (ACR)	3GPP TS 24.611 4.5.2.6.2
Outgoing Call Barring (OCB)	See following list
- barring of all outgoing calls	3GPP TS 24.611 4.5.2.4.1
- barring of outgoing international calls	3GPP TS 24.611 4.5.2.4.1
- barring of outgoing international calls excl. home country	3GPP TS 24.611 4.5.2.4.1
Communication Hold (HOLD)	See following list
- hold without announcement	3GPP TS 24.610 A.1.1
- hold with announcement	3GPP TS 24.610 A.1.2
- resume without announcement	3GPP TS 24.610 A.2.1
- resume with announcement	3GPP TS 24.610 A.2.2
Ad-Hoc Multi Party Conference	GSMA IR.92 2.3.3
- create a conference with a conference factory URI	3GPP TS 24.147 A.3.2
- user requesting IMS to join another user	3GPP TS 24.147 A.4.4.1
- user leaving the conference	3GPP TS 24.147 A.6.2.1
- user subscribing to the conference event package	3GPP TS 24.147 A.5.2.1