Avaya Aura Contact Center Communication Control Toolkit API Comparison

Notice

While reasonable efforts have been made to ensure that the information in this document is complete and accurate at the time of printing, Avaya assumes no liability for any errors. Avaya reserves the right to make changes and corrections to the information in this document without the obligation to notify any person or organization of such changes.

Documentation disclaimer

"Documentation" means information published by Avaya in varying mediums which may include product information, operating instructions and performance specifications that Avaya generally makes available to users of its products. Documentation does not include marketing materials. Avaya shall not be responsible for any modifications, additions, or deletions to the original published version of documentation unless such modifications, additions, or deletions were performed by Avaya. End User agrees to indemnify and hold harmless Avaya, Avaya's agents, servants and employees against all claims, lawsuits, demands and judgments arising out of, or in connection with, subsequent modifications, additions or deletions to this documentation, to the extent made by End User.

Link disclaimer

 Avaya is not responsible for the contents or reliability of any linked Web sites referenced within this site or documentation provided by Avaya. Avaya is not responsible for the accuracy of any information, statement or content provided on these sites and does not necessarily endorse the products, services, or information described or offered within them. Avaya does not guarantee that these links will work all the time and has no control over the availability of the linked pages.

Warranty

Avaya provides a limited warranty on its Hardware and Software ("Product(s)"). Refer to your sales agreement to establish the terms of the limited warranty. In addition, Avaya's standard warranty language, as well as information regarding support for this Product while under warranty is available to Avaya customers and other parties through the Avaya Support Web site: http://support.avaya.com. Please note that if you acquired the Product(s) from an authorized Avaya reseller outside of the United States and Canada, the warranty is provided to you by said Avaya reseller and not by Avaya.

Licenses

 THE SOFTWARE LICENSE TERMS AVAILABLE ON THE AVAYA WEBSITE, HTTP://SUPPORT.AVAYA.COM/LICENSEINFO/ ARE APPLICABLE TO ANYONE WHO DOWNLOADS, USES AND/OR INSTALLS AVAYA SOFTWARE, PURCHASED FROM AVAYA LLC., ANY AVAYA AFFILIATE, OR AN AUTHORIZED AVAYA RESELLER (AS APPLICABLE) UNDER A COMMERCIAL AGREEMENT WITH AVAYA OR AN AUTHORIZED AVAYA RESELLER. UNLESS OTHERWISE AGREED TO BY AVAYA IN WRITING, AVAYA DOES NOT EXTEND THIS LICENSE IF THE SOFTWARE WAS OBTAINED FROM ANYONE OTHER THAN AVAYA, AN AVAYA AFFILIATE OR AN AVAYA AUTHORIZED RESELLER; AVAYA RESERVES THE RIGHT TO TAKE LEGAL ACTION AGAINST YOU AND ANYONE ELSE USING OR SELLING THE SOFTWARE WITHOUT A LICENSE. BY INSTALLING, DOWNLOADING OR USING THE SOFTWARE, OR AUTHORIZING OTHERS TO DO SO, YOU, ON BEHALF OF YOURSELF AND THE ENTITY FOR WHOM YOU ARE INSTALLING, DOWNLOADING OR USING THE SOFTWARE (HEREINAFTER REFERRED TO INTERCHANGEABLY AS "YOU" AND "END USER"), AGREE TO THESE TERMS AND CONDITIONS AND CREATE A BINDING CONTRACT BETWEEN YOU AND AVAYA LLC. OR THE APPLICABLE AVAYA AFFILIATE ("AVAYA").

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

Concurrent User License

Concurrent User License (CU). End User may install and use the Software on multiple Designated Processors or one or more Servers, so long as only the licensed number of Units are accessing and using the Software at any given time. A "Unit" means the unit on which Avaya, at its sole discretion, bases the pricing of its licenses and can be, without limitation, an agent, port or user, an e-mail or voice

Table of Contents

Validity Information

- Unless superseded by a later document issue, this document applies to AACC release: **7.0 or later**.
- Document Issue: **01.01 with appendix**
- Issue Date: **November 22, 2023**

Introduction

Purpose
The following document provides a cross-reference of functionality provided in the Avaya Aura^{TM} Contact Center (AACC) Communication Control Toolkit (CCT) Service Oriented Architecture Open Interfaces (SOA-OI) with the standard CCT .NET interface. This cross-reference can be used as a decision making aid for the user to determine which interface is suitable for the customer's individual requirements.

Available CCT API's
Currently there are 2 main application programming interfaces (API) available for accessing and manipulating communications control information within AACC:

- **CCT .NET API**
    - The CCT .NET API is the preferable API when working with Microsoft Windows 10 and 11.
    - The CCT .NET API requires .NET Framework 4.0 to be installed on the target machine.
    - This API is used by existing AACC client applications such as Avaya Aura Agent Desktop (AAAD), and the CCT Reference client application.
    - Detailed reference information including tutorials on the CCT .NET API can be found in the latest *AACC CCT SDK Programmers Reference Guide*.

- **CCT SOA-OI Interface**
    - For environments that are not exclusively using Microsoft operating systems the CCT SOA-OI provides an alternative platform independent solution.
    - This open interface provides a number of services which enable customer clients to access and manipulate fine grained CCT data as a service on their Enterprise Service Bus (ESB).
    - The SOA OI interface is subdivided into two separate API.
        - SOA OI SOAPAPI
        - SOA-OI REST API

CCT .NET API
The CCT SDK is a communications control API which is implemented as a set of .NET types and interfaces that provides the user with a set of objects that can be used to develop communications applications. These applications communicate with the CCT server. This API uses the Microsoft .NET Framework to allow you to quickly build and deploy robust applications that take advantage of the Microsoft common language runtime environment as well as security and connectivity features provided using the Windows Communication Foundation (WCF).

By using the CCT API the client application has the following capabilities. It can

- securely connect to the Communication Control Toolkit server

- obtain terminal and address information from the Communication Control Toolkit server

- install event handlers on objects within the Communication Control Toolkit server to be notified of significant communications events (for example, incoming contact, contact disconnected, and so on)

- query the current capabilities of various communications objects within the Communication Control Toolkit server

- get or set the values of properties on the various communications objects (for example, the "do not disturb" state of a terminal)

- invoke methods on the communications objects to perform some function (for example, originate a call to some destination address)

Advantages:
Some advantages of using the CCT .NET interface include:

- CCT .NET provides standard Net.TCP transport channels for high performance and secure transport of data to and from the server.

- The transport channel provides for user authentication using standard Windows user authentication (SSPI) and can be used in environments where the client and server are separated by a firewall.

- Any windows user that can be authenticated by the CCT server can be configured on the server as a CCT user.

- CCT SDK API provides for creation of single sign-on (SSO) style applications.

- All communication transmitted over the client server connection is encrypted using standard encryption mechanisms built into Microsoft Windows OS.

- The .NET Framework provides a number of basic features that make it easier to deploy a variety of applications. These features include:
  - **No-impact applications -** This feature provides application isolation and eliminates DLL conflicts. By default, components do not affect other applications.

  - **Private components by default -** By default, components are deployed to the application directory, and are visible only to the containing application.

  - **Controlled code sharing -** Code sharing requires you to explicitly make code available for sharing, rather than being the default behavior.

  - **Side-by-side versioning -** Multiple versions of a component or application can coexist. You choose which versions to use, and the common language runtime enforces versioning policy accordingly.

- o **XCOPY deployment and replication -** Self-described and self-contained components, and applications, can be deployed without registry entries or dependencies.

- o **On-the-fly updates -** Administrators use hosts, such as ASP.NET, to update program DLLs, even on remote computers.

- o **Integration with the Microsoft Windows Installer -** Advertisement, publishing, repair, and install-on-demand are all available when deploying your application.

- o **Enterprise deployment -** This feature provides easy software distribution, including the use of Active Directory.

- o **Downloading and caching -** Incremental downloads keep downloads smaller, and components can be isolated for use only by the application for zero-impact deployment.

- o **Partially trusted code -** Identity is based on the code rather than the user. Policy is set by the administrator, and no certificate dialog boxes appear.

- Hot Desking is explicitly facilitated using the toolkit workstation property. Using .NET hot-desking greatly reduces consumption of CCT licenses.

Disadvantages
CCT .NET API is not platform independent.

CCT SOA Open Interface

Advantages:
SOA is an architectural style whose goal is to achieve a loose coupling among interacting software agents. This means that applications existing on different Operating Systems written in different programming languages can communicate and exchange data with each other. The CCT SOA Open Interfaces provides a number of advantages that make it easier to deploy a variety of applications.

These advantages include:

- **Interoperability** – CCT SOA-OI web services offer developers an opportunity to create non-proprietary solutions.

- **Standardized protocol** – The CCT SOA-OI uses industry standard protocols such as SOAP/REST/WSDL for communication.

- **Loose Coupling –** SOA has a loose coupling of services with operating systems, programming languages and other technologies and hence does not place additional requirements on the client to download Avaya libraries.

- **Platform independent –** SOA is not associated with a specific OS, programming language or technology and hence leaves the clients free to develop in their language(s) of choice.

- **Common Integration Point –** SOA is becoming a common integration technology between different technologies and applications.

Disadvantages / Limits of Usage
- Where a user is sensitive to latency, such as scenarios where the WS client and server are separated over a wide area network, it is preferable to use the CCT .NET interface. In contrast to SOA, the .NET interface has an extensive built-in caching mechanism to compensate for high-latency networks that eliminate unnecessary server calls if the information requested by the client is already available locally. The SOA interface does not provide this caching. Clients developed using the SOA services can implement a caching mechanism if they so wish.

- There are differences in the underlying architecture between the SOA OI and .Net CCT SDK which lead to limits on:

  1. The supported number of concurrent logged in agents
  2. The contact-rate (measured in contacts per hour) that the CCT OI SDK can support.

  These limits are outlined in the respective customer documentation.

  Where an application interacting with the AACC is being developed with requirements that exceed either (1) the supported agent count or (2) the contact rate, then the CCT OI SDK is not a suitable choice. This is also true if current requirements come within 15% of these limits and the client application is expected to scale up to greater numbers in the near future.

SOA-OI SOAP API
The SOA SOAP based API offers an alternative detailed feature set that enables a user to expose the full contact center call model over their service bus. .

The list of services provided in the SOAP API is as follows:

Service Name	Description
Address	This service models an Address in the system It provides methods to read and set the attributes of the address.
Agent	This service models an Agent in the system It provides agent related functionality such as login/logout as well as methods for querying or manipulating data related to a given agent.
AgentTerminal	This service provides access to agent terminal connection related functionality.
AgentTerminalConnection	This service models an agent terminal in the system It provides methods to read and set the attributes of the terminal.
Connection	This service models a Connection in the system It provides methods to read and set the attributes of the connection.

Contact	This service models a Contact in the System and as such provides functionality for querying and manipulating a given contact.
RoutePointAddress	This service provides functionality to query control status of RoutePointAddresses as well as take and release control etc.
RoutePointConnection	This service provides functionality associated with RoutePointConnections such as querying capabilities, routing and applying media treatment etc.
TerminalConnection	This service is used to provide functionality associated with TerminalConnections such as answer/hold, conferencing, transferring, retrieving associated contacts and connections etc.
Terminal	This service provides functionality associated with Terminals such as querying and setting associated states, retrieving associated Address and TerminalConnection objects etc. This service models a User in the system It provides methods to read attributes of the user such as Address(s), Terminal(s) etc.
User	This service models a User in the system It provides methods to read attributes of the user such as Address(s), Terminal(s) etc.
NotificationProducer	This service offers clients the ability to subscribe for contact center notifications.
NotificationConsumer	A Notification Consumer represents the endpoint to which the Notification Producer sends events.
Session	This is a summary service and represents a subset of all the other services exposed and as such can be used in place of them in many cases. SessionService exposes a higher level API thus making it easier to use.

Service Capabilities

All services include a method to read the current capabilities of that service (e.g. can Answer, can transfer, can originate etc.). It is recommended that user applications initially read the service capabilities available on their system, and use this information to pre-check that a capability is available before attempting to perform capability specific read/write method calls over the SOA interface.

SOA-OI Session Service

SOA-OI session service offers a subset of functionality from the API that is typically used in rapid client or simple server based solutions. The session service abstracts the user from the full call model allowing them to work with the more familiar concept of Agent's, Addresses and Terminal without requiring an in-depth knowledge of Connections and Terminals Connections.

High availability

AACC supports "High Availability" for fault tolerant and mission critical contact centers. The following levels of campus high availability are supported :

- Mission Critical High Availability for SIP-enabled Contact Centers.
- Hot-standby High availability for AML-based Contact Centers.
- Warm standby High Availability.

For full details and definitions of these various levels, please refer to latest "Avaya Aura Contact Center Fundamentals and Planning" documentation.

For Mission Critical High Availability campus solutions CCT SOA OI sessions and notifications are seamlessly retained. No further action is required on the client side. The user can register for the *start/stopHASwitchEventListener* so that they receive a courtesy notification of switchover if they so require. This feature is provided as part of the User Service. As this is a courtesy notification to indicate switchover no

identifier information is passed back in the notification to the service user. It's up to the service user to determine how they should handle this notification.

Hot Standby and Warm Standby campus solutions do not preserve Web services sessions during a switchover. Clients developed using CCT SOA-OI must be programmed to detect and respond to HA switchovers. For information and guidelines on programming Web services to handle HA switchovers, see the SDK documentation. It is not recommended that the customer use the start/stopHASwitchEventListener in this scenario as there is no guarantee given that they will receive the notification on/after switchover. It is recommended that the customer uses the polling method as described in the SOA SDK documentation in this instance.

In all solutions using Remote Geographic Node (RGN), clients developed using Web services must be programmed to be able to connect to the RGN IP address if the RGN is brought online when the campus site is unavailable

SOA-OI REST API
The REST interface currently provides a subset of the functionality available via the SOAP or .NET interfaces

Calls to the REST API are typically lightweight and are easier to embed in web applications code in HTML/JavaScript than the SOAP API. The REST interface consists fundamentally of URIs with HTTP calls. In their simplest form REST calls can be made directly from a standard web browser. Due to this lightweight nature, the REST API can be targeted at pure thin-client development and Web 2.0 developers.

In contrast to REST, using the SOAP API requires more complexity to handle the request and generate the response but offers a greater range of extensions and stronger tool support. Additionally SOAP provides better support for sending attachments such as binary data.
Unlike the SOAP-based web services, the REST web service interfaces are not described using a WSDL. Instead the interface is documented as a standalone wiki-style web page, *Avaya_REST_API.html* that accompanies the CCT REST API.

Currently only subscription for call related events are available using the REST API..

High availability
SOA REST API does not provide any support for high availability campus or remote solutions.

Feature Comparison

Feature Capacity
The following table gives an indication of maximum capacity values supported by the CCT SOA-OI interfaces in the contact center. These values are limited by server platform and telephone switch capacity. Please refer to customer documentation for equivalent capacity figures for various scenarios using the CCT .NET interface.

		CCT .NET[1]	SOA-OI[2] SOAP
SIP	Maximum count of logged in Agents	3000	1000
	Maximum contact rate per hour	45000	10,000
CS1K	Maximum count of logged in Agents	5000	1500
	Maximum contact rate per hour	100000	15,000

[1] Note that the CCT .NET capacity varies depending on underlying switch & server type used. These figures are indicative only. Please refer to latest customer documentation for relevant figures for your system.

[2] Comparable figures for REST API will be made available at a later date.

Overview of Supported Functions

The following table attempts to give a high level comparison of features provided in the CCT .NET and various SOA open interfaces. As contact center configurations may vary, please check that the relevant service capabilities on your system.

Basic CCT Functions	CCT .NET	SOA –OI SOAP	SOA-OI (Session Service)	SOA-OI REST
Make Call	✓	✓	✓	✓
Hold Current Call	✓	✓	✓	
Un-hold Call	✓	✓	✓	
Drop Current Call	✓	✓	✓	✓
Disconnect (Release)	✓	✓	✓	✓
Single Step (Blind) Transfer Call[3]	✓	✓	✓	
Initiate Supervised Transfer	✓	✓	✓	✓
Complete Transfer	✓	✓	✓	✓
Initiate Conference Call	✓	✓	✓	
Complete Conference Call	✓	✓	✓	
Call Forward	✓	✓		
Cancel Call Forward	✓	✓		
Join Conference	✓			
Get Call Capabilities	✓	✓		
Get Data	✓	✓	✓	

[3] Single Step (Blind) Transfer not supported in SIP systems.

Delete Data	✓	✓	✓	
Append Data	✓	✓	✓	
Do Not Disturb (Make Set Busy)	✓	✓		
Get/Set UUI	✓	✓		
Send DTMF (for example, credit card number to IVR)	✓	✓	✓	✓
Mute/Unmute[4]	✓			
Consult	✓	✓		
Park/Unpark[4]	✓	✓		
Message Waiting Indicator[4]	✓	✓		
Answer	✓	✓	✓	✓
Agent Login	✓	✓	✓	✓
Agent Logout	✓	✓	✓	✓
Supervisor Force Agent Logout	✓			
Supervisor Force Agent Ready / Not Ready	✓			
Ready Status	✓	✓	✓	
Not Ready Reason Code	✓	✓	✓	
Set Activity Code	✓	✓	✓	
Agent Whisper[4]	✓	✓		
Observe Call	✓	✓		
Set Call Treatment	✓	✓		
Barge In	✓	✓		
Call Supervisor	✓	✓	✓	
Emergency Key	✓	✓	✓	
Intrinsic support	✓	✓	✓	
Hot Desking	✓			

[4] Feature currently not implemented by any service provider.

Overview of Supported Events

The following chapter attempts to give a high level comparison of asynchronous notification events supported in the CCT .NET, and SOA SOAP based interface. As contact center configurations may vary, please check the relevant service capabilities on your system.

Currently only subscription for call related events are available using the REST API.

CCT .NET and SOA-OI SOAP

Notification events for the following connection state changes are supported in both the CCT .NET and SOA-OI SOAP interfaces:

ACTIVE	ESTABLISHED	OFFERED
ALERTING	FAILED	PARKED
ANCHORED	IDLE	RETRIEVING
CONTROLLED	INITIATED	ROUTING
DIALING	NETWORK_ALERTING	QUEUED
DISCONNECTED	NETWORK_REACHED	UNKNOWN

Notification events for the following terminal connection state changes are supported in both CCT .NET and SOA-OI SOAP interfaces:

IDLE	INUSE
RINGING	BRIDGED
ACTIVE	DROPPED
HELD[5]	UNKNOWN

Both CCT .NET and SOA-OI SOAP interfaces also provide clients with the ability to register for property change event notifications on Address, Contact, User, and Terminal entities.

CCT .NET ISession interface provides a means of setting up "Session-Global" event handlers for any of the events present on other resource-defining interfaces. This can be used in the case where a user is interested in a certain type of event for all of the resources it has access to. The SOA SOAP notification interface provides a similar feature by using the entity wild-card feature when subscribing for events.

CCT .NET Only
In addition to the above, CCT .NET also provides the ability to register additional state change notification events for User, Terminal and Address entities as well as property change notifications for Connection and Terminal Connection entities.

Additional support for remote connection state changes is only supplied in the CCT .NET interface.

CCT .NET also exclusively provides "Force Answer" events. These occur when a terminal connection associated with an agent terminal controlled by the CCT SDK Client session has been in a Ringing state for longer than a specified call forcing timeout period.

---

[5] Note: SOA-OI provides ability to listen for an additional AUTOHELD state. The .NET interface reports AUTOHELD as a normal HELD event.

Disclaimer

Appendix. Additional information on use of the Avaya Communications Control Toolkit SDK's for Aura Communication Manager deployments

This table shows the applicability of the items listed in this Appendix to each of the CCT SDK's.

Applicability	CCT.Net SDK	CCT SOAP based OI	CCT REST based OI
Item 1	Yes	Yes	Yes
Item 2	Yes	Yes	Yes
Item 3	Yes	Yes	Yes
Item 4	No	Yes	Yes

Item 1- Communications Control Toolkit SDK support for multiple applications.

The CCT SDK ClientProfile Enumeration section defines profiles that can be selected by the client application to define the desired behavior of the Communication Control Toolkit session when interacting with the client application and the back end switching platform.

⊟ **Members**

Member name	Value	Description
FullControl	0	This default profile should be used by client applications that require full control of the communications end-points. This profile would typically be used by applications which are designed to interact with a single Contact Center agent and his or her associated voice and/or multimedia devices.
MonitorOnly	1	This profile provides the client application with the ability to monitor Contact Center agents and their associated communications end-points but prevents the application from performing any control over these objects.
CallRecorder	2	This profile is similar to the **Monitor** profile but is specifically designed to be used by call recording applications that would normally monitor large numbers of agents in the Contact Center but which would not perform any agent and/or end-point control except for call recording related operations.

A maximum of one **FullControl** profile Communications Control Toolkit SDK developed Application is supported.

When Avaya Aura Contact Center Agent Desktop is in use it is the sole **FullControl** CCT application of the Agent, subsequently it is not supported to deploy any additional CCT SDK developed Applications with **FullControl** profile for the same Agent.

The only profiles supported for any additional CCT SDK developed Applications are **MonitorOnly** and **CallRecorder**.

If a CCT application uses FullControl on an Agent, no other FullControl CCT applications (including Avaya Contact Center Agent Desktop) may be used to control the same Agent.

No other application can control this Agent or Supervisor in parallel with the CCT SDK developed application instance. This restriction includes multiple instances of a CCT SDK developed application.

If a CCT API developed application is controlling an Agent or Supervisor (known as User in the Contact Control Service), it must be the sole application controlling this Agent or Supervisor. No other application using AACC API's can control this Agent or Supervisor in parallel with the CCT API developed application instance. This restriction includes multiple instances of a CCT API developed application. The applications referred to above include, Avaya Agent Desktop, any Contact Center SDK Reference Client or any application developed against the Contact Center SDK's listed below:

• CCT Open Interfaces SOAP SDK

• CCT Open Interfaces REST SDK

• CCT .NET SDK

• CCMM Web Communications SDK (all)

• CCMM Agent Web Services SDK

• Enterprise Web Chat SDK

• Contact Control Service (CCS) SDK's

Item 2 – Agent and Communications Control Toolkit SDK mapping

The Communication Control Toolkit SDK is designed to support a one-to-one relationship between each Agent and any application monitoring the Agent via a session associated with a CCT User profile. The SDK is not designed to monitor multiple Agents via a single session (CCT User profile). Use of such a mapping is not supported. Applications monitoring multiple agents must utilize multiple sessions with a CCT-User profile for each agent.

Item 3 - Windows versions supported by Communications Control Toolkit SDK

Avaya Aura Contact Center 7.1.2 including the Communications Control Toolkit SDK are supported on Windows 10 and 11.

Item 4 – SIP Call Recording User Account

The Contact Center Commissioning for Avaya Aura® Unified Communications guide is published on the AACC Support portal: https://support.avaya.com/support/en/products/P0793/avaya-aura-contact-center

This document contains a section "Configuring SIP Call Recording" describes configuration of the SIP Call Recording user account. This is a special user account restricted to Avaya and 3rd party Call Recorders and Speech Analytics applications.

This Recording User applies to the Communication Control Toolkit Open Interface and is not relevant to the Communications Control Toolkit .Net SDK. Refer to the Communication Control Toolkit Open Interface for further information. Communication Control Toolkit Open Interfaces (devconnectprogram.com)