Cisco Collaboration Platforms

8.6.1

Introduction

In this topic, you will learn about Cisco's suite of on-premise and cloud-based collaboration solutions. These efficient means of communication enable businesses to increase productivity and decrease costs.

Cisco Unified Communications Manager (Unified CM) was originally designed to route calls over an IP network, and Cisco quickly added more features to meet the growing needs of businesses. Unified CM is:

- Used to configure and automate device provisioning, call routing, and profiles and settings management in a single solution.
- Deployed in hospitals, banks, universities, and government agencies to manage the increasing number of devices and user profiles.

Cisco created many APIs to further extend its capabilities, including AXL and UDS. These APIs provide Cisco partners and developers an easier way to automate and manage the devices, user profiles, and calls.

Contact Center provides robust customer support for call centers with agent desktop, supervisor, and reporting capabilities. Contact Center can be used with Unified CM or as a standalone. Contact Center is used to manage and route high-volume calls, text, video, and data to the right agent at the right time with the right information.

Finesse is an agent and supervisor desktop. The Finesse API provides Cisco partners and developers with more flexibility and customization of the desktop to deliver customer-driven care.

Webex encompasses Meetings, Teams, and Devices. Webex simplifies the needs of voice, video, and data into one solution so that people can collaborate more efficiently:

- Webex Teams is a meetings and messaging app designed to improve collaboration. The Webex Teams API enables you to create chat bots and integrations to streamline activities.
- Webex Devices include digital whiteboards, telepresence units, and room controls. Customize and personalize the Webex Devices with the xAPI.

8.6.2

Cisco Unified Communications Manager

Cisco Unified Communications Manager is also known as Unified CM, CUCM or CallManager. Cisco Unified Communications Manager:

- Is an IP-based communications solution to support mobile and remote workers in small, medium, or enterprise-level businesses.
- Is an integration for voice, video, and data into a single on-premise solution for call control and session management.
- Is highly extensible, with various APIs for configuration, management, monitoring, and call control.

Purpose

The primary function of Unified CM is to manage phone users, IP phones, directory numbers, and to connect and manage calls to the desired destinations. With Unified CM you can:

- Define gateways, trunks, remote destinations, and more telephony-related information.
- Configure a full range of call handling tasks, including hold, transfer, call forward, and starting conference calls. Unified CM stores configuration data in an internal database, with a Publisher→Subscriber cluster architecture.

Unified CM Advanced Features

Unified CM's advanced features include:

- **AXL** - SOAP APIs (XML requests) for managing various aspects of CUCM via programs rather than the web UI. AXL is for administration and configuration of CUCM.
- **UDS** - Enables end users to update their own personal settings stored on CUCM.
- **Serviceability** - RisPort provides real-time registration and IP address information for phones. PerfMon provides real-time monitoring of Cisco Unified CM hardware and software.
- **Platform Administrative Web Services (PAWS)** - Lets you programmatically perform operations such as upgrades instead of performing the steps manually.
- **Softphone compatibility** - Softphone devices can allow a Jabber user to make and receive phone calls.
- **Extension Mobility** - Users can log into any phone and that phone will switch from its default directory number to the user's directory number while the user is logged in.
- **Cisco Jabber Voice & Video SDK** - A JavaScript library/add-on and Chrome/Firefox extension that enables you to create a web page that can act as a softphone (audio and optional video). You can also use the web page to leverage your physical phone.
- **Java Telephony API (JTAPI) or Telephony API (TAPI)** - Enables you to write programs that automate the way calls are handled.
- **Client Matter Codes (CMC)** - Enables you to manage call access and accounting. CMC forces the user to enter a code to specify that the call relates to a particular client matter. You can assign client matter codes to customers, students, or other populations for call accounting and billing purposes.
- **Forced Authentication Codes** - Limit access to certain directory numbers users by requiring users to enter an authentication code first.

- **Hunt Lists** - Lists of directory numbers. If you call a number in a Hunt List and that number is not available (busy, for example), it rings the next number in the list, and so on, until it either reaches a callee or runs out of numbers in the list.
- **Music/Video on Hold** - Define your own music and/or video to play when a user is on hold.

Extensions

Unified CM integrates with other services:

- **Cisco Instant Messaging and Presence (Cisco IM&P)** - Requires a server installed with Cisco IM&P, also known as CUP. The Cisco IM&P server synchronizes presence and instant messaging (IM) information between Cisco Unified Communications and other applications.
- **Voicemail** - Advanced voicemail management, requires a server installed with Cisco Unity Connection (CUC).
- **Contact Center** - Routes customer service requests to a pool of agents to provide efficient customer support.

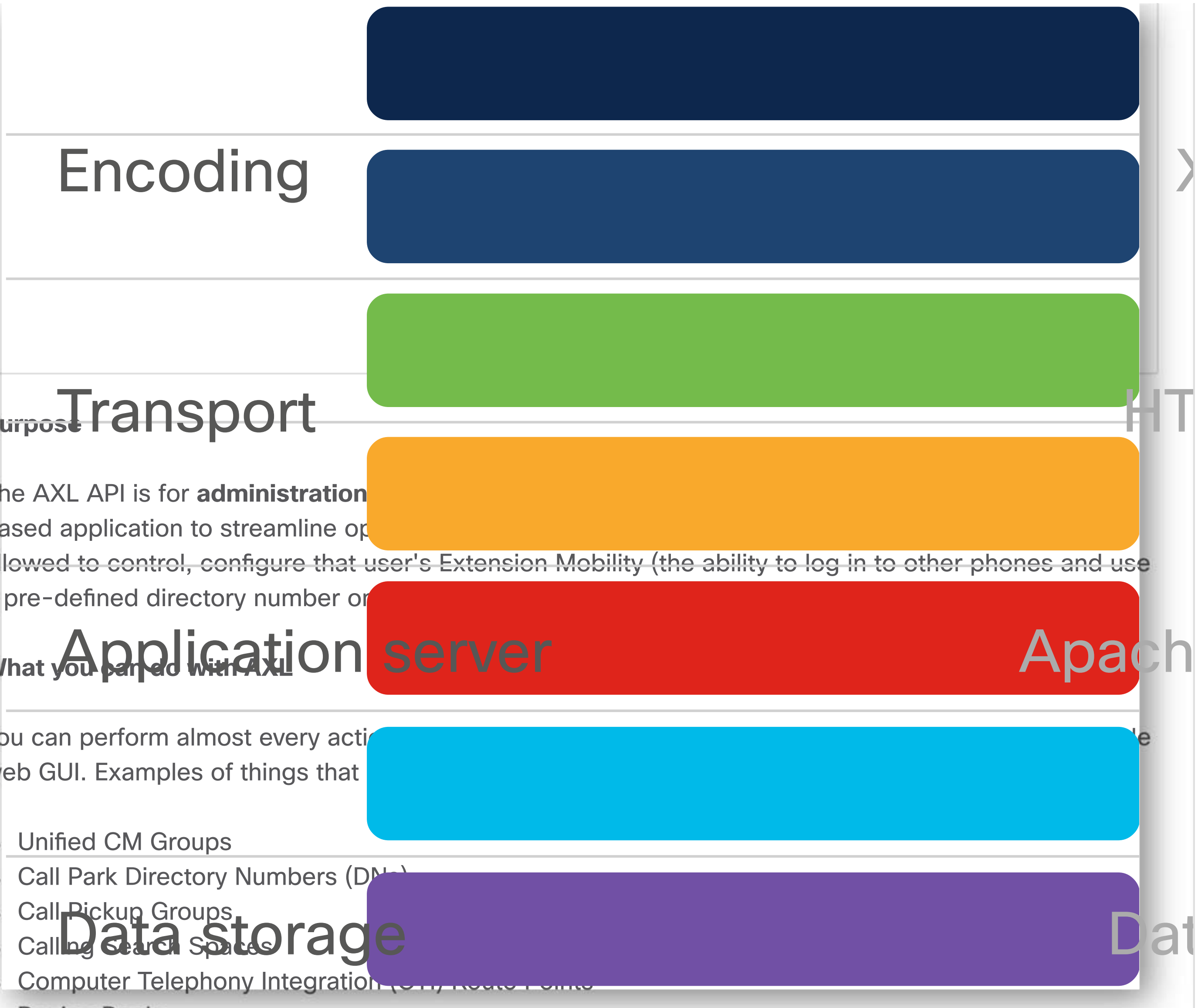
8.6.3

Administrative XML Layer

Administrative XML Layer (AXL) is an XML/SOAP-based interface that provides a mechanism for inserting, retrieving, updating, and removing data from the Unified Communication configuration database. Developers can use AXL and the provided Web Services Description Language (WSDL) to create, read, update, and delete objects such as gateways, users, devices, route-patterns and much more.

AXL and the Unified Communication Configuration Database

Application	App
API	AXL (Admi
Protocol	S



Purpose

The AXL API is for **administration** based application to streamline op allowed to control, configure that user's Extension Mobility (the ability to log in to other phones and use a pre-defined directory number or

What you can do with AXL

You can perform almost every acti e web GUI. Examples of things that

- Unified CM Groups
- Call Park Directory Numbers (DNs)
- Call Pickup Groups
- Calling Search Spaces
- Computer Telephony Integration (CTI) Route Points
- Device Pools
- Device Profiles
- Dial Plan Tags
- Dial Plans
- Digit Discard Instructions
- Directory Numbers
- Gateways (Analog, T1, PRI)
- Locations
- Media Gateway Control Protocol (MGCP) Devices
- Phones
- Process Nodes
- Process Node Services
- Regions
- Route Filters
- Route Groups
- Route Lists
- Route Partitions
- Service Parameters
- Translation Patterns
- Users
- Voice Mail Ports

How it works

AXL is a SOAP interface, meaning it is based on the exchange of XML documents (defined in a schema, or Web Services Description Language (WSDL). Several APIs and development frameworks can consume the AXL WSDL to produce 'native' code/libraries. These allow you to use AXL without handling XML. For simple requests, it can be simpler to make AXL requests by creating XML strings and sending via HTTP POST.

The methods for the objects begin with:

- `list`
- `add`
- `update`
- `get`
- `remove`

For example `listPhone` lists all phones, and `addUser` creates a new user.

SQL queries

You can also perform direct SQL queries to update or retrieve data in the Unified CM configuration database using `ExecuteSQLupdate` or `ExecuteSQLquery`. You need to refer to the Unified CM Database Dictionary to create your SQL statement. Please note there is no backwards compatibility between Unified CM releases when using SQL queries.

If you want to download the WSDL to produce a 'native' library, download it from Unified CM:

Step 1. Log into Unified CM as an administrator.

Step 2. Select `Application` -> `Plugins` from the menu.

Step 3. Click the `Find` button. The first entry in the list is the Cisco AXL toolkit.

Step 4. Download the file `axlsqltoolkit.zip` and unzip it in a working directory. This kit contains AXL schema for a WSDL and XSD files for a variety of Unified CM versions, so you have a nice collection to pick from for your particular version of Unified CM.

Versioning

The AXL schema version is backwards-compatible for up to two major releases. This means developers can use AXL schema version 10.0(1) for Unified CM release 10.0(1) through 12.5(1). The schema version can be specified in the SOAP action header of the request. If it is not specified, the system uses the oldest supported schema. You can see the version being specified in the sample request below. You can access more details in the versioning docs.

Sample request

This sample request returns a list of phones and all their settings. By requesting `name` and specifying a wildcard character `%`, you can retrieve a list of all phone names.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns="http://www.cisco.com/AXL/API/12.5">
   <soapenv:Header/>
```

```
    <soapenv:Body>
       <ns:listPhone>
          <searchCriteria>
             <name>%</name>
          </searchCriteria>
       </ns:listPhone>
    </soapenv:Body>
 </soapenv:Envelope>
```

Advanced features

AXL has some advanced features. One of the most useful is the Change Notification Feature. You can use this SOAP request repeatedly to see what changes have been made to the system since the last time you ran the request. You can request to see changes about specific categories, like Phone or User, or just see all changes.

Related SOAP-based Administration APIs

UC Manager Serviceability includes similar SOAP-based API requests for retrieving information about phones such as the registration status, the IP address, and more.

It also includes a performance monitoring API, and an API to manage and get the status of CUCM services. To learn more about these services, view the Serviceability developer documentation.

Read more

More information about AXL is available in the AXL developer documentation.

8.6.4

User Data Services

User Data Services (UDS) is a REST-based API that provides a mechanism for inserting, retrieving, updating and removing data from the Unified Communication configuration database. Developers can use the UDS API to create, read, update, and delete user resources, including devices, subscribed services, and speed dials.

Purpose

The UDS API is designed for end users to configure settings. For example, the API provides authenticated access to enable end users to update their personal settings for their own devices.

What you can do with UDS

One example of how you can use the UDS API is to create a directory search or manage user preferences and settings in your web application. Because the API provides functionality for end users to login, those end users can then change their settings to enable and disable features like "Call Forward All" and "Do Not Disturb". Examples of actions with UDS include:

- Directory search for users
- Manage Call Forward, Do Not Disturb, and Speed Dial settings, including visual and audible alert preferences
- Set Language and Locale
- Subscribe to IP Phone Service Applications
- Reset PIN or password credentials
- Configure Remote Destinations used with Cisco Mobility & Single Number Reach

How it works

UDS is a REST-based interface that sends and receives XML-formatted data. UDS implements the four common HTTP request methods: `GET`, `POST`, `PUT`, and `DELETE`.

For example `GET` /speedDials returns a list of the user's speed dials, and `PUT` /speedDials updates that user's speed dials.

UDS supports Single Sign-on (SSO) and Basic Authentication for authentication. Not all UDS API requests require authentication, but the UDS resources that do require authentication are:

- credentials
- device(s)
- extension(s)
- remoteDestination(s)
- speedDial(s)
- subscribedService(s)
- user
- userPolicy

Sample request

This sample XML request returns a list of your user's settings and preferences:

```
GET https://{host}:8443/cucm-uds/user/{userId}
Accept: application/xml
```

This is the response:

```
<user version="{version}" uri="https://{host}:8443/cucm-uds/user/{userId}">
    <id>{userPkid}</id>
    <userName>{userId}</userName>
    <firstName>firstName</firstName>
    <lastName>lastName</lastName>
    <middleName>middleName</middleName>
    <nickName>nickName</nickName>
    <phoneNumber>8134567</phoneNumber>
    <homeNumber>8134455</homeNumber>
    <mobileNumber>8131111</mobileNumber>
    <mobileConnect>true|false</mobileConnect>
    <userLocale uri="https://{host}:8443/cucm-uds/installedLocales" value="0"
appliesToAllDevices="true|false">English, United States</userLocale>
    <email>someone@example.com</email>
    <directoryUri>someone@example.com</directoryUri>
    <msUri>someone@example.com</msUri>
    <department>department</department>
    <manager>manager</manager>
    <title>title</title>
    <pager>8137777</pager>
    <primaryExtension uri="https://{host}:8443/cucm-
uds/user/{userId}/userExtension/{extensionPkid}">
        <description>extensionDescription</description>
        <directoryNumber>81134953</directoryNumber>
        <callForwardAllDestination>
            <sendToVoiceMailPilotNumber>true|false</sendToVoiceMailPilotNumber>
            <destination>4352134</destination>
        </callForwardAllDestination>
        <messageWaitingVisualAlert>true|false</messageWaitingVisualAlert>
        <messageWaitingVisualAlertPreference
appliesToAllLineAppearances="true|false" value="">Use System
Default</messageWaitingVisualAlertPreference>
        <messageWaitingAudibleAlertPreference
appliesToAllLineAppearances="true|false"
value="1">On</messageWaitingAudibleAlertPreference>
        <onACallRingPreference appliesToAllLineAppearances="true|false" value="0"
>Use System Default</onACallRingPreference>
        <notOnACallRingPreference appliesToAllLineAppearances="true|false"
value="0" >Use System Default</notOnACallRingPrefence>
        <voiceMailPilotNumber>5555</voiceMailPilotNumber>
        <label appliesToAllLineAppearances="true|false">label</label>
        <logMissedCalls
appliesToAllLineAppearances="true|false">true|false</logMissedCalls>
    </primaryExtension>
    <accountType useLdapAuth="true|false">ldap|local</accountType>
    <homeCluster enableCalendarPresence="true|false"
enableImAndPresence="true|false">true|false</homeCluster>
    <devices uri="https://{host}:8443/cucm-uds/user/{userId}/devices"/>
    <credentials uri="https://{host}:8443/cucm-uds/user/{userId}/credentials"/>
    <userExtensions uri="https://{host}:8443/cucm-
uds/user/{userId}/userExtensions"/>
    <enableDoNotDisturb
appliesToAllDevices="true|false">true|false</enableDoNotDisturb>
    <callForwardAllDestination appliesToAllExtensions="true|false">
        <sendToVoiceMailPilotNumber>true|false</sendToVoiceMailPilotNumber>
        <destination>43521</destination>
    </callForwardAllDestination>
    <speedDials uri="https://{host}:8443/cucm-uds/user/{userId}/speedDials"/>
    <serviceProfile uri="http://{host}:6970/SPDefault.cnf.xml"/>
</user>
```

Read more

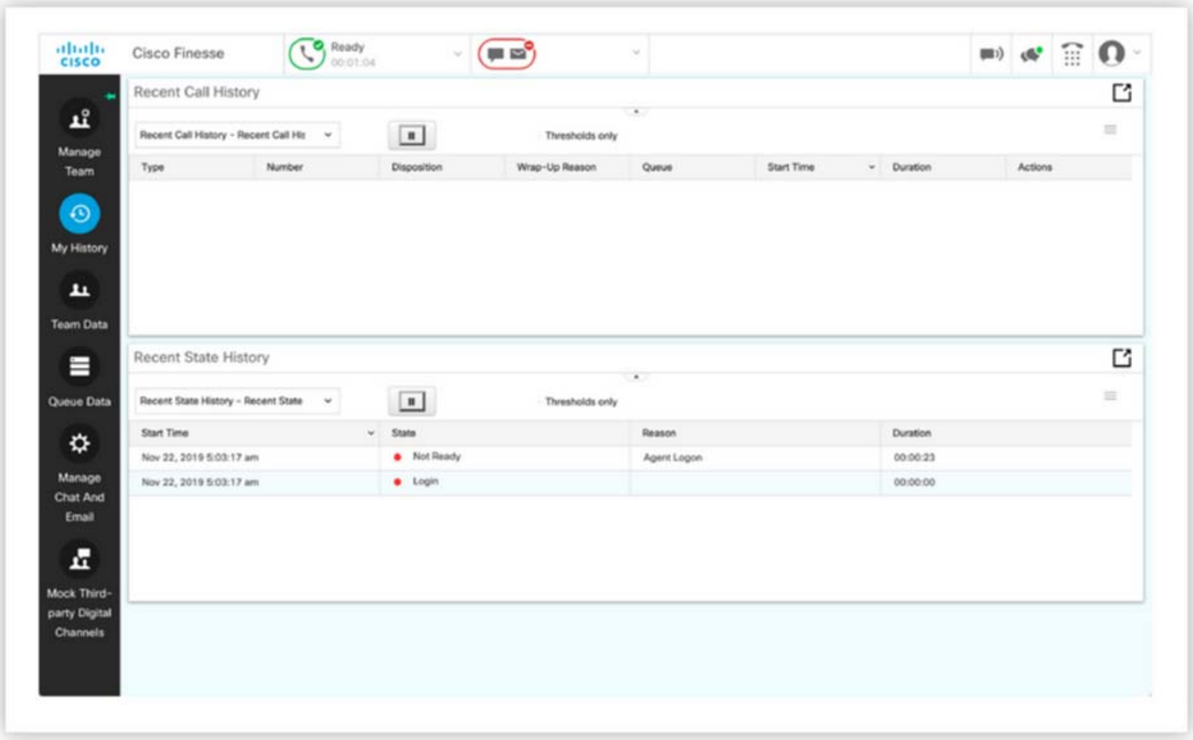
For more information about UDS, refer to the UDS developer documentation.

8.6.5

Cisco Finesse

Finesse is Cisco's browser-based contact center agent and supervisor desktop. Finesse has REST APIs and JavaScript APIs that can be used to build fully custom agent desktops, integrate contact center functionality into applications, and integrate applications into the Finesse agent and supervisor desktop. This integration can be accomplished in the form of OpenSocial gadgets.

Finesse Agent and Supervisor Desktop

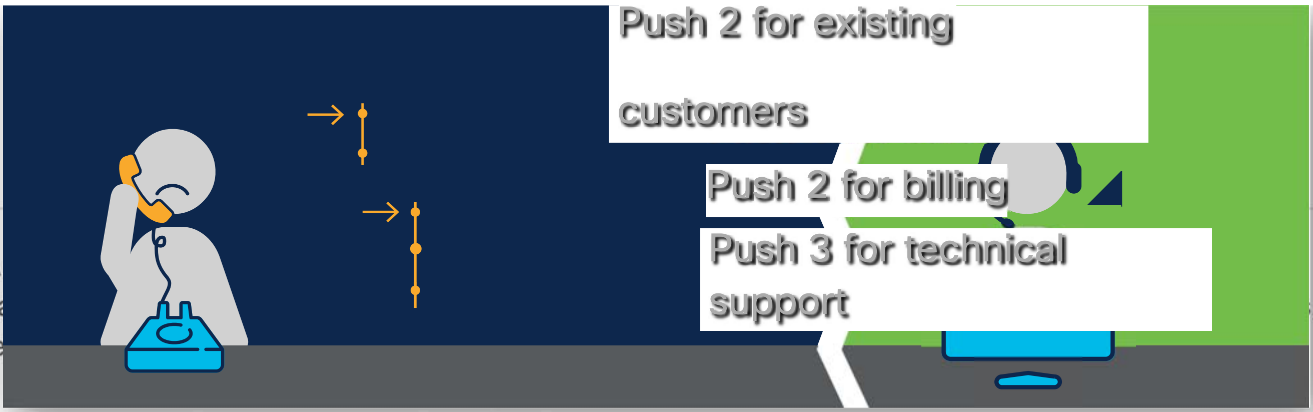


Type	Number	Disposition	Wrap-Up Reason	Queue	Start Time	Duration	Actions

Start Time	State	Reason	Duration
Nov 22, 2019 5:03:17 am	● Not Ready	Agent Logon	00:00:23
Nov 22, 2019 5:03:17 am	● Login		00:00:00

What is a Contact Center?

Customer Calling Customer Service Rep



Contact centers have two categories of tasks:

- **Inbound tasks -** When customers initiate communication with customer service. Examples include a customer asking questions about their bill, or needing IT help, or a new customer trying to sign up.
- **Outbound tasks -** A customer interaction made from the contact center agent to a customer. Examples include telemarketing, reminders for an upcoming appointment or service, or a follow-up on a previous customer service interaction.

The workers of a contact center are called agents and their managers are supervisors. They assist the customers and are also known as customer service representatives. They use an application called an agent desktop for the following tasks:

- Manage their incoming work (call, text, chat, email)
- Get the customer data that was provided through a menu prompt or form
- Get customer information from a customer relationship management or an internal database
- Enter notes into the customer's account/file for future reference
- Get help from a knowledge base in order to assist the customer
- Request help from peers or the supervisor

Contact Center Systems

Contact center systems are very complex. This is a high level overview of the features needed to understand Finesse.

Contact center systems perform what is called skill-based routing, using an automatic call distributor (ACD) to intelligently route inbound customer interactions (such as calls, text, chat, email) to customer service agents by matching them based on skills and specialties using sophisticated algorithms.

Contact center systems also provide agent and supervisor management. The systems monitor agent state so that the ACD knows who is available to receive tasks. The following are examples of agent states:

- **Not Ready -** The agent is not available to take incoming task (call, text, chat, email)
- **Ready -** The agent is available to take incoming task from the queue (call, text, chat, email)
- **Talking -** The agent is currently handling a call from the queue.
- **Work -** The agent just finished handling the task and is wrapping up work by adding notes, completing forms, and so on. In this state, the agent will not receive a new task.
- **Logout -** The agent is signed out and not working at the moment.

To provide additional detail into the agent state, there are reason codes that describe why an agent is in a state. There are typically two types of reason codes: Not Ready and Logout.

Finesse deployments

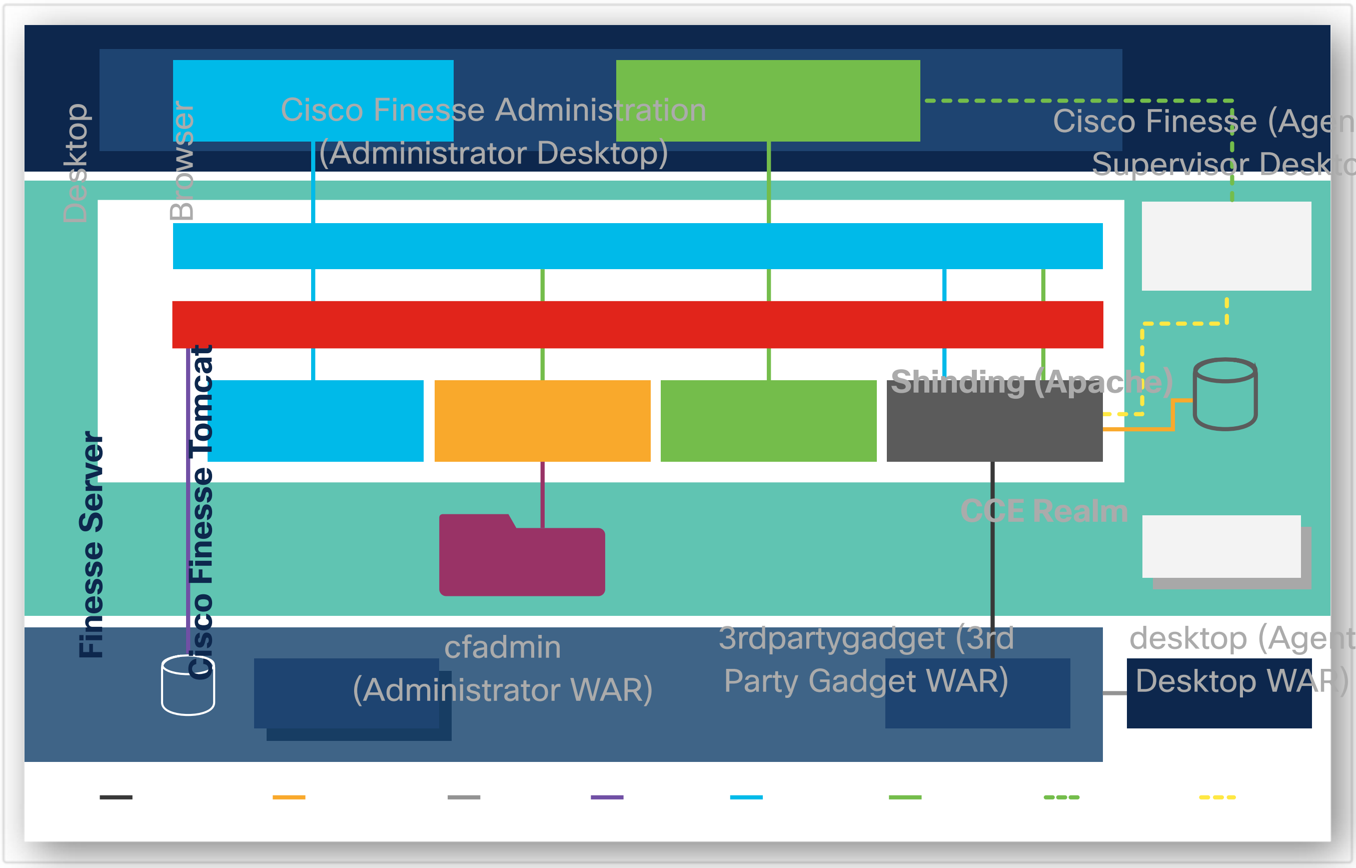
Finesse has two different deployments. In a contact center solution, Finesse sits on top of Contact Center Enterprise or Unified Contact Center Express and cannot function without one of these contact center systems.

Finesse communicates with the contact center system via the CTI protocol. This protocol is different between the Contact Center Enterprise and Contact Center Express systems, but Finesse provides the same interface for the two deployments while maintaining the behavioral differences.

Contact Center Enterprise

Standalone Finesse that is used with a Contact Center Enterprise system is one type of deployment.

Finesse with UCCE Architecture Diagram

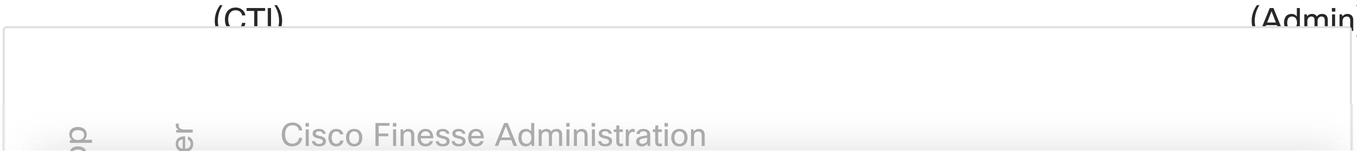

Desktop
Browser
Cisco Finesse Administration (Administrator Desktop)
Cisco Finesse (Agent a Supervisor Desktop)
Finesse Server
Cisco Finesse Tomcat
Shinding (Apache)
CCE Realm
cfadmin (Administrator WAR)
3rdpartygadget (3rd Party Gadget WAR)
desktop (Agent Desktop WAR)
3rdpartygadget
opt directory
Other services
UCCE
AWBD
GED-188 (CTI)
Hibernate
JTAPI
JDBC
HTTP/S (Admin)

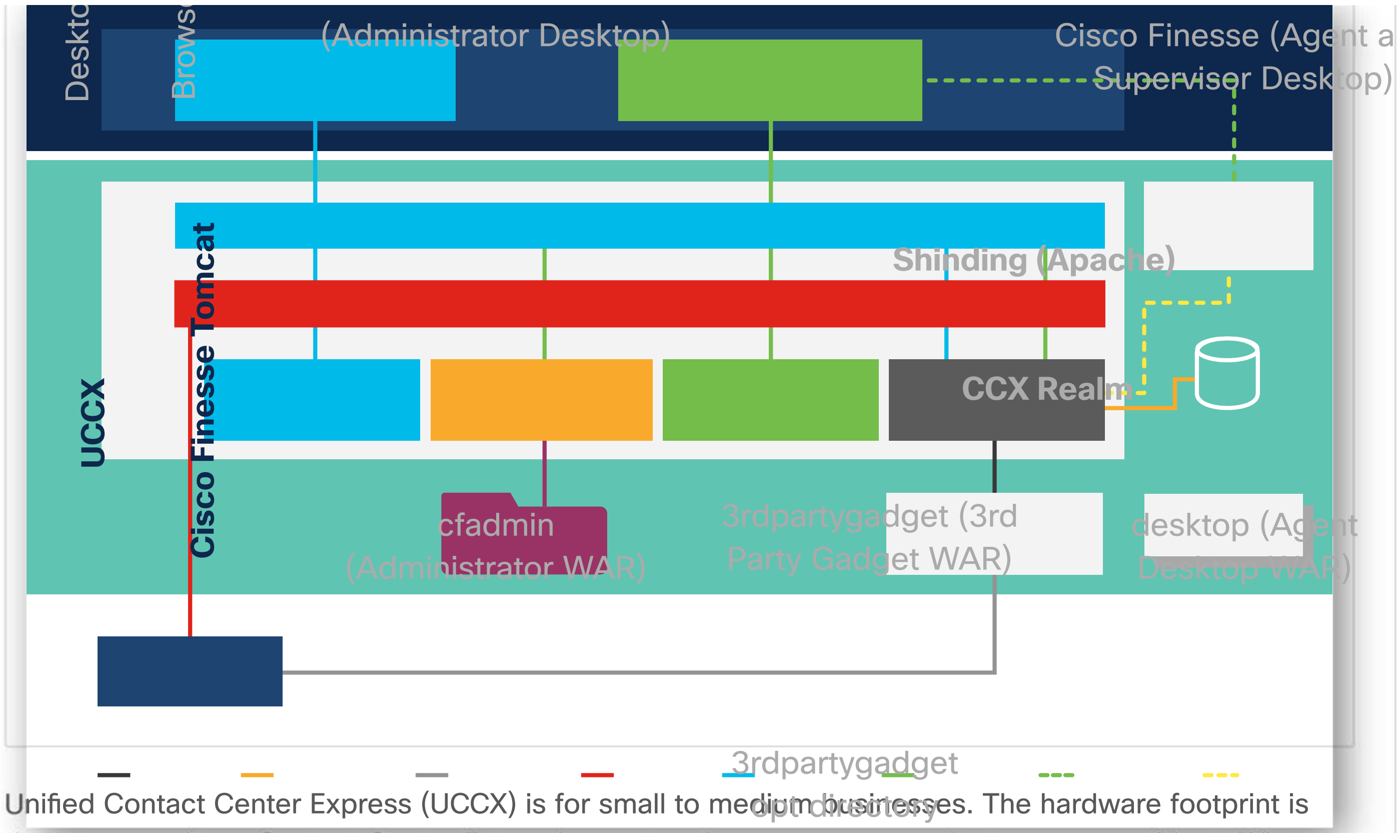
There are two versions of Contact Center Enterprise (CCE), Unified Contact Center Enterprise (UCCE) and Packaged Contact Center Enterprise (PCCE). The difference between the two is mainly hardware footprint, the ease of installation and the number of supported agents. The audience for a Contact Center Enterprise system is large businesses.

Unified Contact Center Express

Co-resident Finesse that is used with Unified Contact Center Express.

Finesse with UCCX Architecture Diagram





Unified Contact Center Express (UCCX) is for small to medium businesses. The hardware footprint is tiny compared to a Contact Center Enterprise system because most products are co-resident. When products are co-resident, it means that it utilizes one server/VM and therefore, one installer. Because UCCX is for smaller businesses, there are not as many features as there are in a CCE deployment.

Finesse APIs

Finesse was built with integrations in mind. It provides an extensive list of REST and JavaScript APIs for developers to integrate Finesse's functionality into applications or applications to be integrated into the Finesse agent desktop.

Finesse REST APIs

Finesse provides REST APIs for performing agent and supervisor actions programmatically. They can be used to build custom agent desktops, integrate into existing applications, and/or build a script to automate tasks. Because the APIs are HTTP-based, they can be used in both thick and thin applications. The details of each REST API can be found in the Finesse Developer Guide, but here is a high-level description of some API functionality:

- **User -** Represents an Agent, Supervisor or Administrator and enables you to retrieve or update user details and state information.
- **Dialog -** Represents a call and the participants. If the media type is voice, it enables you to make calls, take action on calls, and make outbound-related actions.
- **Media -** Represents a user's state in a non-voice Media Routing Domain (MRD) and enables you to get information about a user and manage user state.
- **Team -** Represents a team of Users and enables you to get team details and lists of team messages.
- **SystemInfo -** Represents current state of the system and enables you to retrieve System Details such as XMPP Server, PubSub Domains, Node IP Addresses, Status, and Deployment Type.
- **ClientLogs -** Enables you to send client-side logging to the Finesse Server.

The Finesse REST APIs that use the GET verb are synchronous, which means that the Finesse server will return a response with the requested information immediately. The application must wait for a response before proceeding. The rest of the Finesse REST APIs are asynchronous, which means that the Finesse server will acknowledge that the request was made, but will send a notification with the requested information via the Finesse Notification Service. The asynchronous Finesse REST APIs goes hand in hand with the Finesse Notification Service.

Finesse Notification Service

The Finesse Notification Service is an instance of an OpenFire server that runs as a separate process on the platform. The notification service provides event notification from the Finesse server to any client that is subscribed to a particular resource. OpenFire uses the XMPP protocol as the data communication channel. Applications must communicate with the Finesse Notification service with XMPP or BOSH (Bi-directional streams Over Synchronous HTTP) for web applications.

Finesse JavaScript APIs

The Finesse agent and supervisor desktop is an OpenSocial container that has the ability to host OpenSocial gadgets. Finesse provides JavaScript APIs for building custom OpenSocial gadgets to be used in the out-of-the-box Finesse agent and supervisor desktop.

The Finesse JavaScript library is a layer built on top of the Finesse REST API and Finesse notification service communication. It abstracts the details of the request and notifications by wrapping them into JavaScript classes, methods, and callbacks. By doing so, developers using the Finesse JavaScript APIs do not need to deal with the BOSH connection setup and making the REST API requests directly. The Finesse JavaScript API contains almost all of the same APIs as the Finesse REST APIs. The details of each JavaScript API can be found in the Finesse JavaScript Library.

Finesse Use Cases

There are endless possibilities with what can be built with the Finesse APIs. Here are some common use cases;

- **Problem -** A company needs a contact center agent desktop that is custom to their agents' needs. The provided Finesse agent desktop is not sufficient. **Solution -** The company can build a 100% fully functioning agent desktop that is branded and contains every single feature that can be found in the out of the box Finesse agent desktop. They will need to use most of the User, Dialog, Team APIs in conjunction with the Finesse Notification Service.
- **Problem -** A company uses a customer relationship management (CRM) as their main application for their contact center agents. They want to add agent state and basic call control into the CRM to avoid needing to flip between two applications. **Solution -** The company will use the User and Dialog APIs in conjunction with the Finesse Notification Service. The User APIs will add the agent state capabilities while the Dialog APIs will add the basic call control capabilities.
- **Problem -** A company wants to add a "Click to call" feature to their application. **Solution -** The company will use the Dialog--Create a New Dialog API to add this functionality
- **Problem -** A telemarketing company wants a custom agent desktop that would have only outbound capabilities. **Solution -** The company can build a 100% fully functioning outbound-only agent desktop. They will need to add agent sign in/out, agent state, outbound features, and specific call control for outbound calls. Supported outbound features include receiving an outbound call, accept/close/reject/reclassify outbound calls, and schedule callbacks.
- **Problem -** A company wants a supervisor specific desktop because their supervisors do not take incoming customer calls. **Solution -** The company can build a supervisor desktop that only contains

supervisor capabilities. Finesse has APIs for the following features for their team: team messages, silent monitor, barge, view and change agent's state.

- **Problem -** A company is using the Finesse out of the box agent desktop but wants to add agent state workflows. **Solution -** The company can build a custom gadget for the agent state workflow. The custom gadget has the ability to receive all of the User notification. With that data, the gadget can trigger the workflow accordingly.
- **Problem -** A company is using the Finesse out of the box agent desktop and wants to integrate a different application that has REST APIs into the agent desktop. **Solution -** The company can build a custom gadget and integrate in the application by using the application's REST APIs. The gadget has the ability to call external REST APIs, parse the responses, and display the data accordingly.
- **Problem -** A company wants to integrate a webpage into the Finesse agent desktop. **Solution -** If the webpage allows itself to be loaded in an iframe, they can build a custom gadget that is as simple as just an iframe that loads that webpage.
- **Problem -** A company wants to change the color of the light on their IoT capable headset when the agent states change. **Solution -** The company will build a custom gadget that calls the headset's APIs to change the color of the light when the agent state changes. The gadget has the ability to receive all of the User notifications so it will know when the agent state changes.

Learn more about Finesse

For more details about Finesse, such as a technical overview including Finesse architecture and free sandboxes, take a look at the DevNet Finesse site.

To find the documentation for the Finesse REST APIs, take a look at the Finesse Developer's Guide.

To find the documentation for the Finesse JavaScript APIs, take a look at the Finesse JavaScript Library.

8.6.6

Webex Teams

Cisco Webex Teams is an online collaboration solution to connect people and teams through chat, voice, and video. With the Webex Teams app, you gain access to secure virtual work spaces. You also use messaging and file sharing with third-party app integrations.

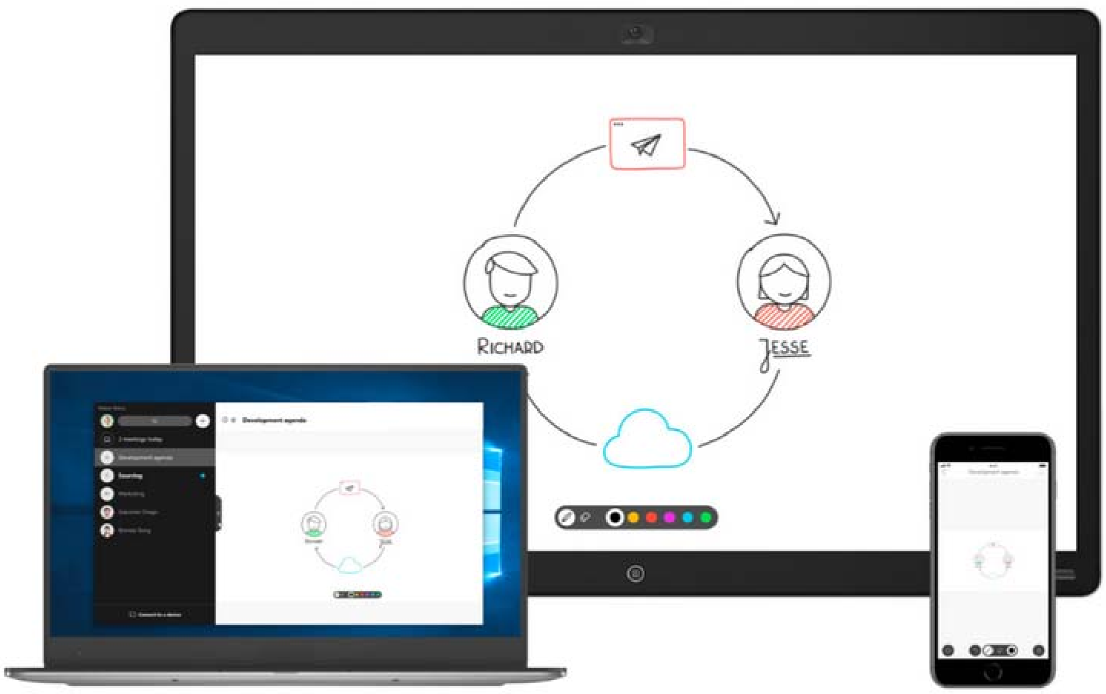
Benefits of Webex Teams

Webex Teams lets you use a single app to contain content and information for team collaboration. Teams also integrates with Cisco Webex devices. Information remains safe and secure with advanced security and compliance built-in.

Tour of Webex Teams

Mobile, desktop, and web-based clients are available for the Webex Teams platform.

Webex Teams Clients



Features of Webex Teams

Webex Teams includes the following capabilities and features:

- **Create spaces**
    - Invite people and teams to specific Webex Teams spaces.
    - Send messages, share files, and create or edit whiteboards, securely, with one person or a group of people.

- **Create meetings**
    - High-quality video meetings with screen sharing and white boarding across all of your devices.
    - In-meeting tools, such as the ability to mute others, add guests, and turn on recording, help improve effectiveness and engagement.
    - Chats, files, and whiteboards shared during meetings can be reviewed by everyone later.

- **Work inside and outside your company the same way**: Add people to your shared spaces from your company directory or enter an email address directly.
- **Place calls**
    - Native in-app voice and video calling capabilities to reach other Webex Teams and standards-based SIP endpoint users.
    - These can be enhanced with comprehensive PBX calling features and Cisco IP phones when combined with one of our calling solutions - Cisco Unified Communications Manager, Cisco Hosted Collaboration Solution (HCS), or Cisco Webex Calling.

Security

Your users and sensitive information is kept safe with extensive controls to help you configure and control your security policies.

- Protect messages, files, and whiteboard drawings with end-to-end encryption.
- Manage your own encryption keys on-premises.

- Your policies are maintained even when your employees are collaborating with others outside your company, through integration with your chosen DLP solution.

Integrations

Webex Teams includes pre-built solutions with third-party applications from vendors such as Microsoft, Google Cloud, and Salesforce. With Microsoft Office 365, Microsoft Exchange, and Google Calendar integrations, you can view your meeting list in Webex Teams:

- You will be able to schedule meetings.
- Users can share and edit files from Microsoft OneDrive and SharePoint Online directly in Webex Teams spaces.

Other integrations can be set up using the Webex App Hub to connect Webex Teams with work happening in other tools such as Service Now, Trello, Asana, Salesforce, and JIRA.

Webex Teams works seamlessly with Webex devices to give you the best possible video meeting and teamwork experiences, without complicated setup procedures, cabling, or productivity disruption.

Webex Hybrid Services bridge cloud and on-premises services to smooth your transition to the cloud, while maximizing return on investment (ROI). Includes integrations with on-premises assets such as your calendar service, directory, calling system, conferencing resources, and video devices.

Features of the Webex Teams API

The Webex Teams API is an extensive set of APIs that allow you to interact with the entire Webex Teams platform. From managing organizations, teams, people, rooms, memberships, and messages to creating conversational bots or embedded video calls, you can extend the capabilities of Webex Teams:

- Use the Webex API to create webhooks enabling fine-grained communication with, and control of, applications and services in response to specific events in Webex Teams.
- Create bots that emulate Webex users and mediate with external applications to provide services (often using webhooks).
- Use Webex Embedding APIs for Java, node.js, browsers, iOS, and Android, along with Webex Widgets, to embed Webex voice/video calling and messaging functionality into desktop, web, and mobile applications.

All of this reference information is available in the Webex Teams documentation.

Organizations

Organizations represent a set of people in Webex Teams. Organizations may manage other organizations or be managed themselves. Organization resources can **only** be accessed by an administrator.

METHOD	URL	Description
GET	https://api.ciscospark.com/v1/organizations	List Organizations
GET	https://api.ciscospark.com/v1/organizations/{orgId}	Get Organization Details

Teams

Teams are groups of people with a set of rooms that are visible to all members of that team. The Teams API resources are teams to be managed, created, updated, and deleted.

METHOD	URL	Description
GET	https://api.ciscospark.com/v1/teams	List Teams
POST	https://api.ciscospark.com/v1/teams	Create a Team
GET	https://api.ciscospark.com/v1/teams/{teamId}	Get Team Details
PUT	https://api.ciscospark.com/v1/teams/{teamId}	Update a Team
DELETE	https://api.ciscospark.com/v1/teams/{teamId}	Delete a Team

People

People are registered users of Webex Teams.

METHOD	URL	Description
GET	https://api.ciscospark.com/v1/people	List People
POST	https://api.ciscospark.com/v1/people	Create a Person
GET	https://api.ciscospark.com/v1/people/{personId}	Get Person Details
PUT	https://api.ciscospark.com/v1/people/{personId}	Update a Person
DELETE	https://api.ciscospark.com/v1/people/{personId}	Delete a Person
GET	https://api.ciscospark.com/v1/people/me	Get My Own Details

Rooms (Spaces)

Note: Notice that the resource in the API endpoint refers to "rooms." The user interface refers to "spaces." These are interchangeable terms.

Rooms are virtual meeting places where people post messages and collaborate to get work done. The Rooms API can manage, create, update, and delete rooms.

METHOD	URL	Description
GET	https://api.ciscospark.com/v1/people	List People
POST	https://api.ciscospark.com/v1/people	Create a Person
GET	https://api.ciscospark.com/v1/people/{personId}	Get Person Details
PUT	https://api.ciscospark.com/v1/people/{personId}	Update a Person
DELETE	https://api.ciscospark.com/v1/people/{personId}	Delete a Person
GET	https://api.ciscospark.com/v1/people/me	Get My Own Details

Memberships

Memberships represent a person's relationship to a room. The Memberships API resources allow you to list members of any room that you are in, create or revoke memberships, and memberships can be updated to make someone a moderator of a room.

METHOD	URL	Description
GET	https://api.ciscospark.com/v1/memberships	List Memberships
POST	https://api.ciscospark.com/v1/memberships	Create a Membership
GET	https://api.ciscospark.com/v1/memberships/{membershipId}	Get Membership Details
PUT	https://api.ciscospark.com/v1/memberships/{membershipId}	Update a Membership
DELETE	https://api.ciscospark.com/v1/memberships/{membershipId}	Delete a Membership

Team Memberships

Team Memberships represent a person's relationship to a team.

METHOD	URL	Description
GET	https://api.ciscospark.com/v1/team/memberships	List Team Memberships
POST	https://api.ciscospark.com/v1/team/memberships	Create a Team Membership
GET	https://api.ciscospark.com/v1/team/memberships/{membershipId}	Get Team Membership Details
PUT	https://api.ciscospark.com/v1/team/memberships/{membershipId}	Update a Team Membership
DELETE	https://api.ciscospark.com/v1/team/memberships/{membershipId}	Delete a Team Membership

Messages

Messages are how we communicate in a room. In Webex Teams, each message is displayed on its own line along with a timestamp and sender information. Use this API to list, create, and delete messages.

Message can contain plaintext, rich text, and a file attachment.

METHOD	URL	Description
GET	https://api.ciscospark.com/v1/messages	List Messages
GET	https://api.ciscospark.com/v1/messages/direct	List Direct Messages
POST	https://api.ciscospark.com/v1/messages	Create a Message
GET	https://api.ciscospark.com/v1/messages/{messageId}	Get Message Details
DELETE	https://api.ciscospark.com/v1/messages/{messageId}	Delete a Message

8.6.7

Lab – Construct a Python Script to Manage Webex Teams

In this lab, you will use Webex Teams APIs to authenticate, manage people, manage rooms, manage memberships to rooms, and send a message.

You will complete these objectives:

- Part 1: Launch the DEVASC VM
- Part 2: Get Your Webex Teams Access Token
- Part 3: Test Your Access Token
- Part 4: Manage People in Webex Teams
- Part 5: Manage Rooms in Webex Teams
- Part 6: Manage Memberships in Webex Teams
- Part 7: Manage Messages in Webex Teams

Construct a Python Script to Manage Webex Teams

8.6.8

Webex Devices

Cisco Webex Devices provide access to all of Webex's features. Webex Boards, Room Devices, and Desk Devices enable collaboration through video, calling, and programmability.

Types of Webex Devices

Meet Webex Devices.

Get the most out of Cisco Webex Meetings and Cisco Webex Teams with tools designed for better team collaboration.

All-in-one whiteboard, wireless presentation screen, and video conferencing system for smarter team collaboration.	Intelligent video conferencing devices for meeting rooms of all sizes.	Simple-to-use and compact video conferencing devices designed for desktops.

Webex Board Series

The Cisco Webex Board Series are all-in-one team collaboration devices for meeting rooms and spaces. Benefits include:

- A fully touch-based device that simplifies the meeting experience
- Cloud-based platform that allows you to save and continue work before, during, and after the meeting
- Cloud registration makes it affordable and easy to deploy with end-to-end encryption
- Wireless presentations with no more need for dongles and wires
- Digital whiteboards
- High quality and high fidelity video and audio conferencing
- High-resolution 4K screen

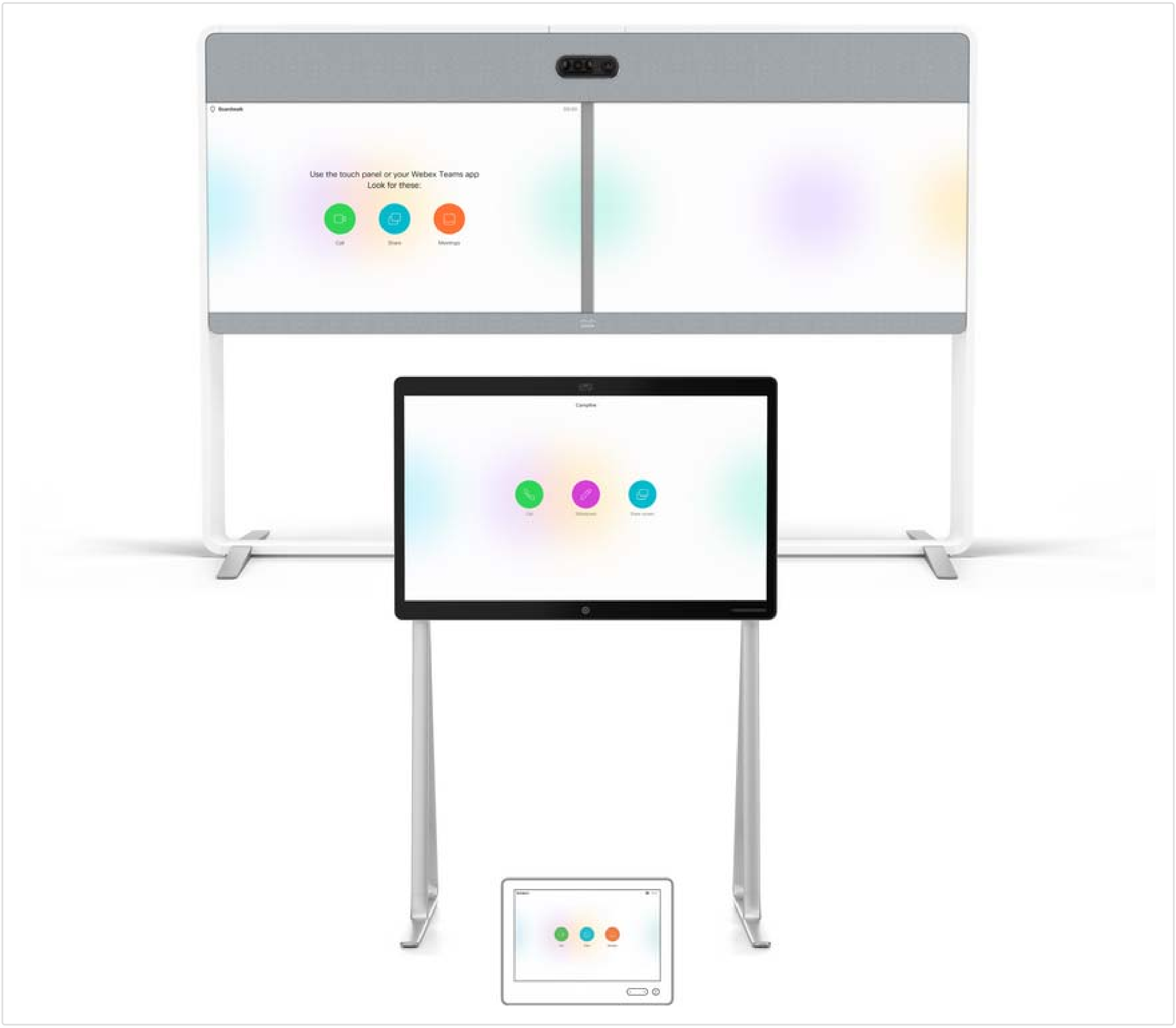
Webex Boards

Webex Room Series

The Cisco Webex Room Series brings integrated video conferencing systems to every room. Benefits include:

- 55- or 70-inch 4K screen(s)
- Intelligent viewing capabilities, such as automatic framing and speaker tracking
- Dual screens, dual content sources, wireless sharing, and 4K content for presentations
- AI based noise suppression, voice-activated controls, and people counting
- APIs and macros allow for expressive meeting personalization
- Built for both on-premises and cloud deployment
- Digital whiteboards

Webex Room Device

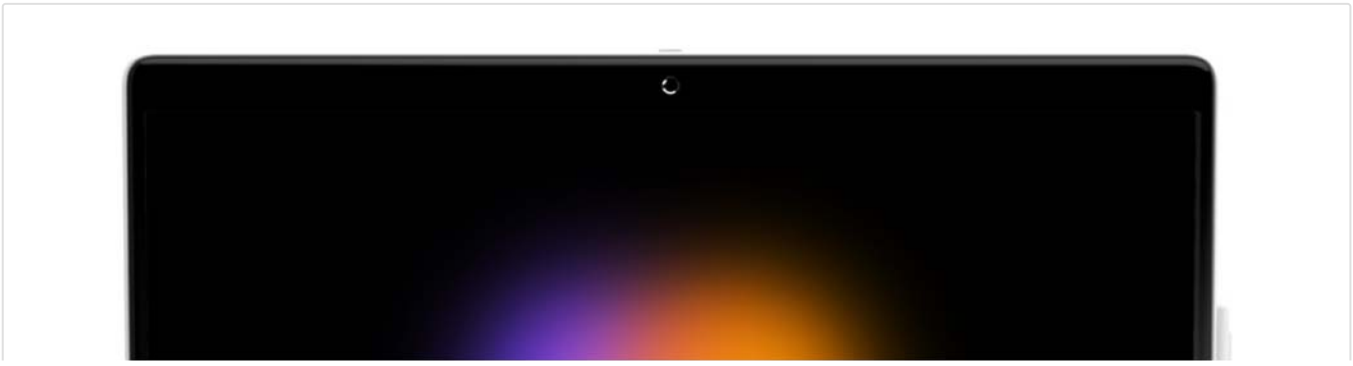


Webex Desk Series

The Cisco Webex Device Series brings high-quality video conferencing to your desktop. Benefits include:

- HD Video and high fidelity audio
- Intuitive touchscreen
- Built for both on-premises and cloud deployment

Webex Desk Device



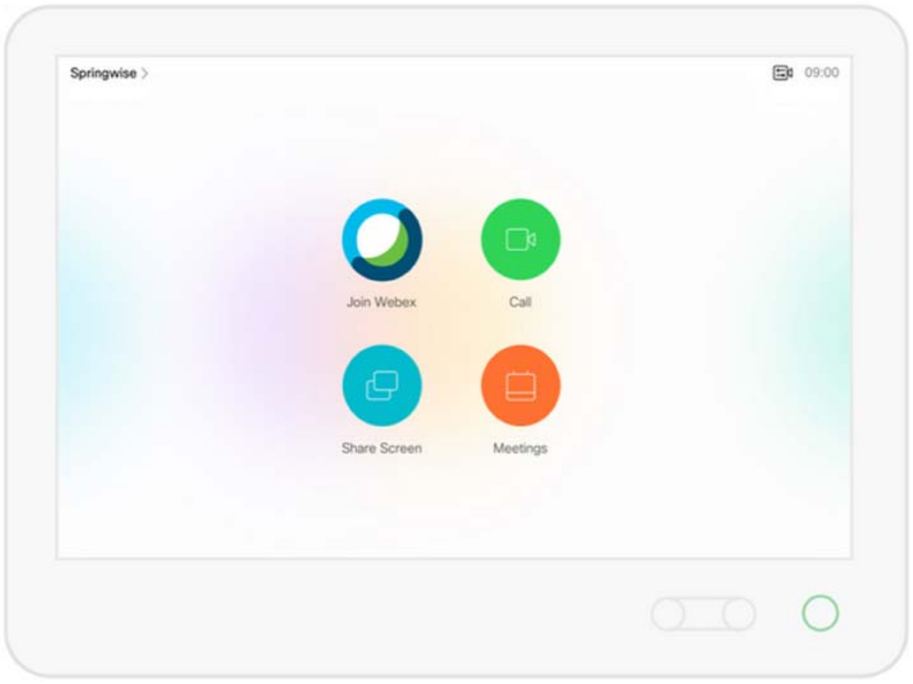
Cisco Touch 10

Cisco Touch 10 is an intuitive touchscreen device for interacting with Cisco conferencing systems.

- Supports the Cisco MX Series, SX Series, IX Series and Webex Room Series
- Power over Ethernet
- Wide language support

Touch 10 customization is enabled with in-room controls, control room devices, and Touch 10 peripherals, and can integrate with Control Systems.

Touch 10 Device



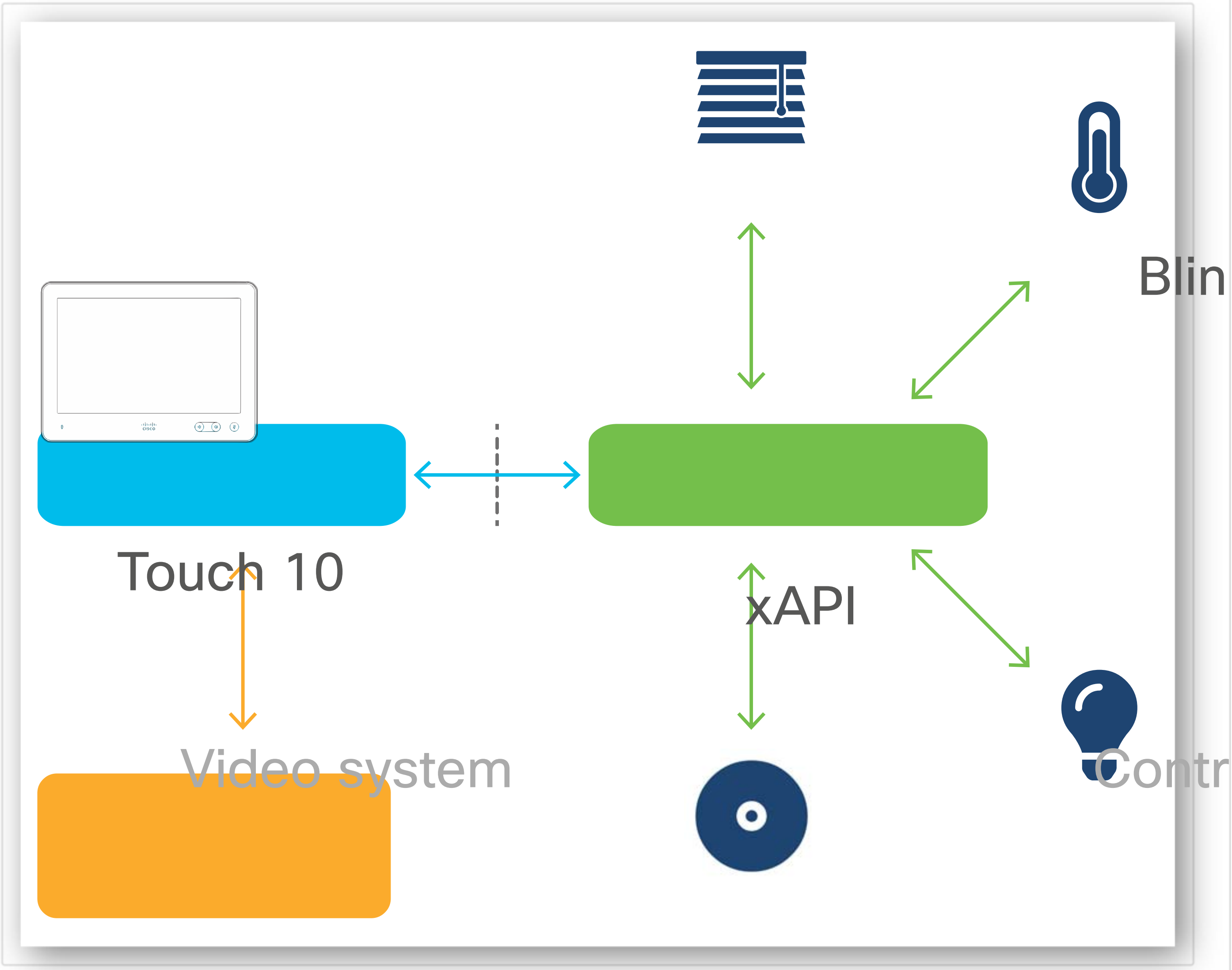
Programmability for Cisco Collaboration Devices

Webex Devices can be customized through the API, known as the xAPI. This enables bi-directional communication with third-party applications and control systems.

There are multiple ways to access xAPI including Telnet/SSH, HTTP, and RS-232 serial connection. Regardless of the method you choose, xAPI has the same general format, behaves similarly, and allows full device control, optimized for integrating with Control Systems.

xAPI supports both XML and JSON and provides direct access via the Command Line. It also supports JavaScript Macros for on-device customization.

Customization Examples



UI Extensions (In-Room Controls) and macros

When speaking about Webex Devices, Cisco is currently transitioning towards use of the generalized term "UI Extensions" to describe custom widgets, buttons, and other virtual controllers that you can create and deploy. These used to be referred to, globally, as "In-Room Controls". Going forward, however, that latter term will be reserved to mean only the subset of UI Extensions that are used to interact in a meeting room (for example, to control lights).

At present, you may still encounter the term "In-room Controls" used in the general sense in the documentation pages. Be sure to read carefully for context!

Likewise, going forward (as of last quarter of 2019), there is only one tool, called the UI Extensions Editor. But you may see earlier versions of this tool referred to as Control Panel Editor, even in study guides and other materials for the last quarter of 2019.

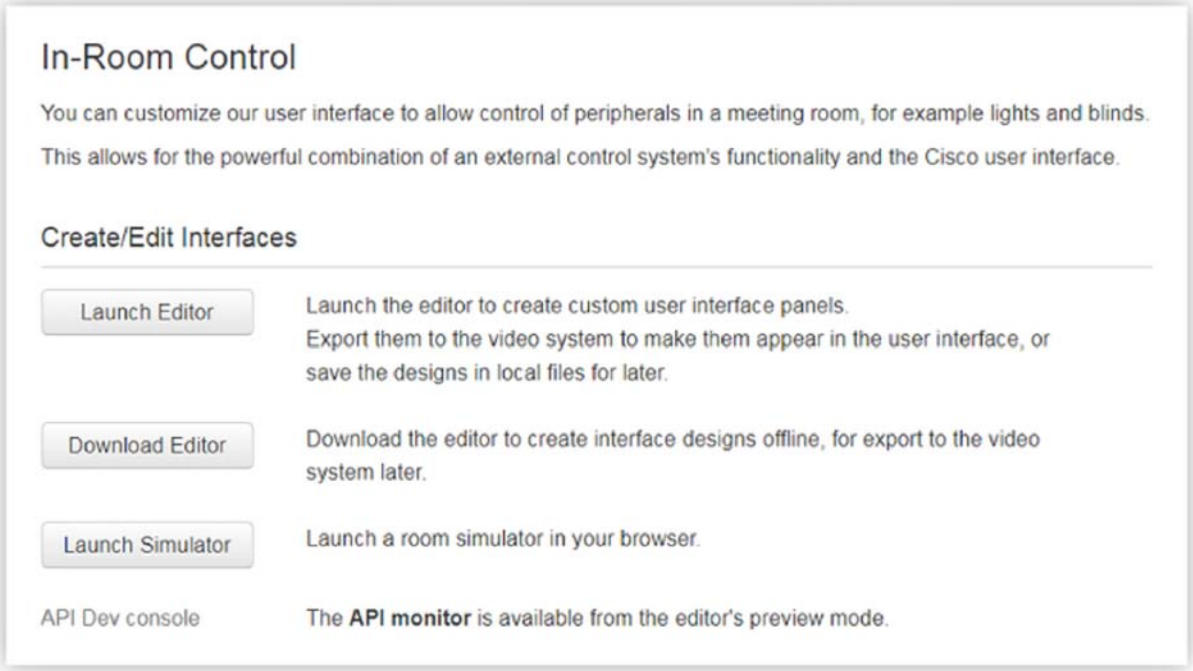
Oth

What UI Extensions do

UI Extensions enable you to add custom user interface elements to the Touch 10 display used to control room devices, as well as the on-screen control interface of the DX Series. These elements can trigger applications to control aspects of the device itself or affect in-room lighting, blinds, video switches, or other peripherals.

UI Extensions can be bi-directionally integrated with additional external control systems (such as AMX/Crestron) with xAPI. To get a feeling for the possibilities offered by custom UI Extensions, Webex collaboration devices come with a built-in control simulator that can be run directly from a web browser.

Customize In-room Controls



In-Room Control

You can customize our user interface to allow control of peripherals in a meeting room, for example lights and blinds.

This allows for the powerful combination of an external control system's functionality and the Cisco user interface.

Create/Edit Interfaces

Launch Editor	Launch the editor to create custom user interface panels. Export them to the video system to make them appear in the user interface, or save the designs in local files for later.
Download Editor	Download the editor to create interface designs offline, for export to the video system later.
Launch Simulator	Launch a room simulator in your browser.
API Dev console	The **API monitor** is available from the editor's preview mode.

Launching the simulator will load a virtual meeting room, equipped with several automated systems controlled from switches on the walls (and your Touch10/DX interface, as you will see later in this step.)

The switch controls let you switch the projector on or off, interact with the projector canvas, close/open the blinds, etc.

Simulator

Try out how the whole meeting experience can be managed using in-room controls.
Load simulator config
Clean up

Personalizing Collaboration Devices

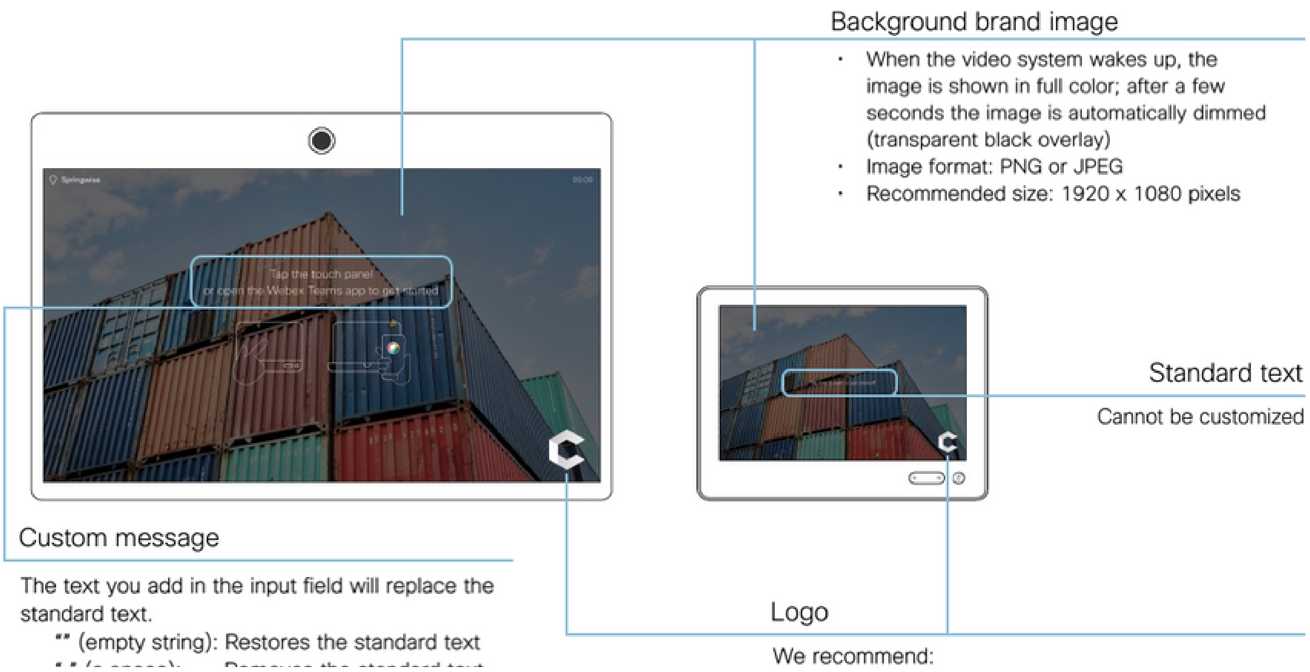
As of version 9.2 of Cisco's Collaboration Endpoint software release, on-screen branding, signage and message customization options let you personalize the appearance of a room device and its Touch10 interface. This capability allows these systems to harmonize with your corporate branding, communicate info to in-room users, update with alerts, and more. While the admin can always statically configure these custom display options via the room device's web interface, you can also explore various ways to use scripts, apps or other automation tools via xAPI to automate such tasks.

Branding and customization of the room device "Halfwake" screen lets you upload your own text and images to customize the appearance of the screen and/or the Touch10 control interface.

In the "Halfwake" state, you can:

- Add a background image (screen/Touch10)
- Add a small logo image to the bottom right corner (screen/Touch10)
- Customize or remove the default on-screen welcome message (screen only)

Touch 10 Personalization in "Halfwake" State


Tap the touch panel or open the Webex Teams app to get started

Background brand image

- When the video system wakes up, the image is shown in full color; after a few seconds the image is automatically dimmed (transparent black overlay)
- Image format: PNG or JPEG
- Recommended size: 1920 x 1080 pixels

Standard text

Cannot be customized

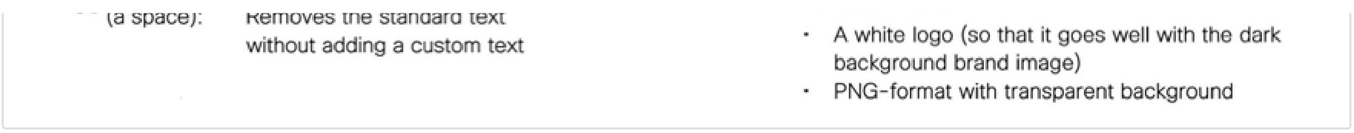
Custom message

The text you add in the input field will replace the standard text.

"" (empty string): Restores the standard text

Logo

We recommend:



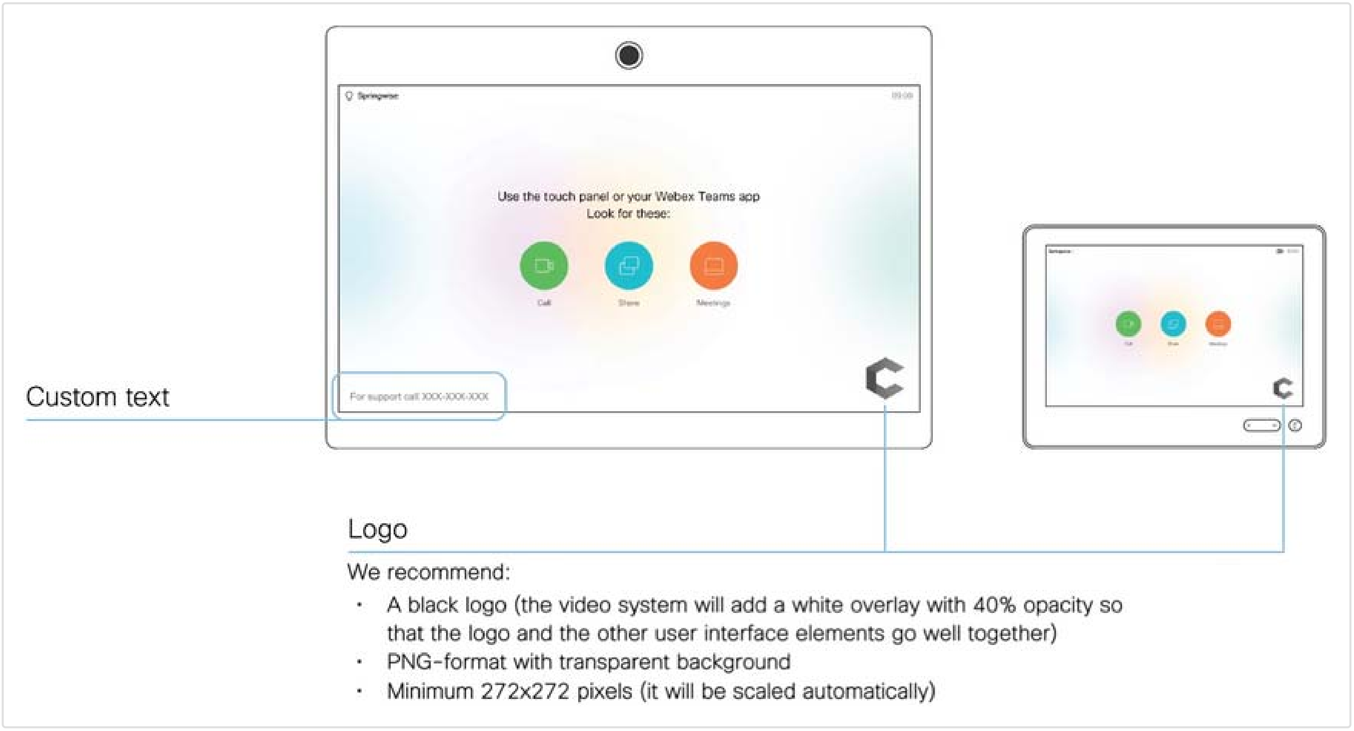
" " (a space): Removes the standard text without adding a custom text

- A white logo (so that it goes well with the dark background brand image)
- PNG-format with transparent background

In the "Awake" state, you can:

- Add a small logo image to the bottom right corner (screen/Touch10)
- Add a label or message to the bottom left corner (screen only)

Touch 10 Personalization in Awake State


Custom text
Use the touch panel or your Webex Teams app
Look for these:
For support call XXX-XXX-XXX
Logo

We recommend:

- A black logo (the video system will add a white overlay with 40% opacity so that the logo and the other user interface elements go well together)
- PNG-format with transparent background
- Minimum 272x272 pixels (it will be scaled automatically)

Running Javascript on devices with macros

CE v9.2.1+ enables you to deploy custom code to the device itself via the macro feature. This feature lets you upload JavaScript code and run it directly on the collaboration device (hosted in a secure 'sandbox' environment). This custom code can interact with the device using the exposed xAPI JavaScript object.

Ideally, code developed on an external app server using `jsxapi` could be uploaded to run directly on a collaboration device without requiring an external server. However, the macro JavaScript environment has some limitations, including the lack of local file storage, the inability to install additional JavaScript packages via NPM, and restrictions on establishing network connections.

8.6.9

Project Activity 5: Network Programmability and Automation

In this activity, you will complete the following tasks:

- Design a new application for network automation.
- Use data modeling with YANG.
- Implement automation with NETCONF.
- Automate the process using Python.

Refer to the **DEVASC Project Rubric** below for this activity to record your process and outcomes.

Scenario

In this scenario, you will work on a new application. Your new IT team consists of software developers, network engineers and cybersecurity professionals, all with DEVASC backgrounds and certifications. Your company is looking to:

- Be more responsive to customer needs and requirements
- Have shorter development cycles (more responsive to changes, updates, customer requirements)
- Increase deployment frequency
- Have more dependable releases
- Align more closely with business objectives
- Use automation wherever possible

The team uses a common philosophy, culture, discipline, and a set of holistic methodologies to deliver and support products and services for both external and internal customers. Your manager wants your team to use the DevOps approach to demonstrate the team's ability to implement the necessary tools within your network.

In your first meeting with your manager, your team has identified many new features and improvements that could simplify network operations. You have an opportunity to also explore new problems, but your first priority is to support Level 1 Support Engineers (L1 SE) who are receiving daily calls from customers. Customers usually require some sort of network change. One common request is a need to change a description on a router’s interface.

The L1 SE team does not have management access to the network infrastructure. They escalate the requests to the L2 network team engineers, who are spending valuable time every day with simple manual configuration changes. Your manager wants to find a way to optimize these processes and has tasked your team with designing a system that would let the L1 SE team trigger some of these changes using a simple chatbot (or web interface, console application, etc.).

Because configuration is one of the requirements, your team has decided to use NETCONF. Your team will create an automated process that demonstrates how L1 Support Engineers can make updates to networking devices.

Operational Objective - Automate the following process:

- Verify the current running-config.
- Make three changes to the configuration.
- Verify the changes.
- Verify the new running-config.
- Send a notification message to the WebEx SE Teams group (L1 and L2 SEs) announcing the update.

Technical Objectives:

- Utilize data modeling with YANG.
- Implement network automation with NETCONF.
- Automate the process using Python.

Deliverable/Rubric:

- Which changes did your team choose to implement in the running-config?
- Why were these features chosen?
- Document your team member roles, knowledge and skills sets, if anything has changed in relation to this new Project Activity.
- Provide a brief description of your team strategy for completing this project, if anything has changed in relation to this new Project Activity.

Final Deliverables

At this point, your project has been completed. Your team will present to your manager:

- Presentation
- Team activities and reflection

Presentation

Deliverable/Rubric: Create a presentation. Your presentation should include:

- Application code
- Describe the YANG Model chosen
- Provide example output of running code showing:
  - running-config before changes
  - running-config after changes
  - Message sent to WebEx Teams

- Reflection points – what issues have you faced while working on this activity, how did you find solutions, what have you learned, etc.

Team Activities and Reflection

Deliverable/Rubric: Your manager is interested in knowing how everyone worked together as a team. Here is a list of questions from your manager:

- What did you enjoy about working as a team? What worked well?
- What team problems did you encounter and how did you resolve them?
- What technical problems did you encounter and how did you resolve them?
- How was each team member held accountable individually and for the team as a whole?
- What was your team's decision-making process?
- Overall, how were the team dynamics and what were any lessons learned?

Project Activity 5 – Network Programmability and A...

8.5
Cisco Compute Management

8.7
Cisco Security Platforms