

Developing Customer Experience Integrations

cisco Live !

Rue Green
Sr. Solutions Engineer, Full Stack Software Developer

Session ID: BRKCCT-2556

Cisco Webex App

Questions?

Use Cisco Webex App to chat with the speaker after the session

How

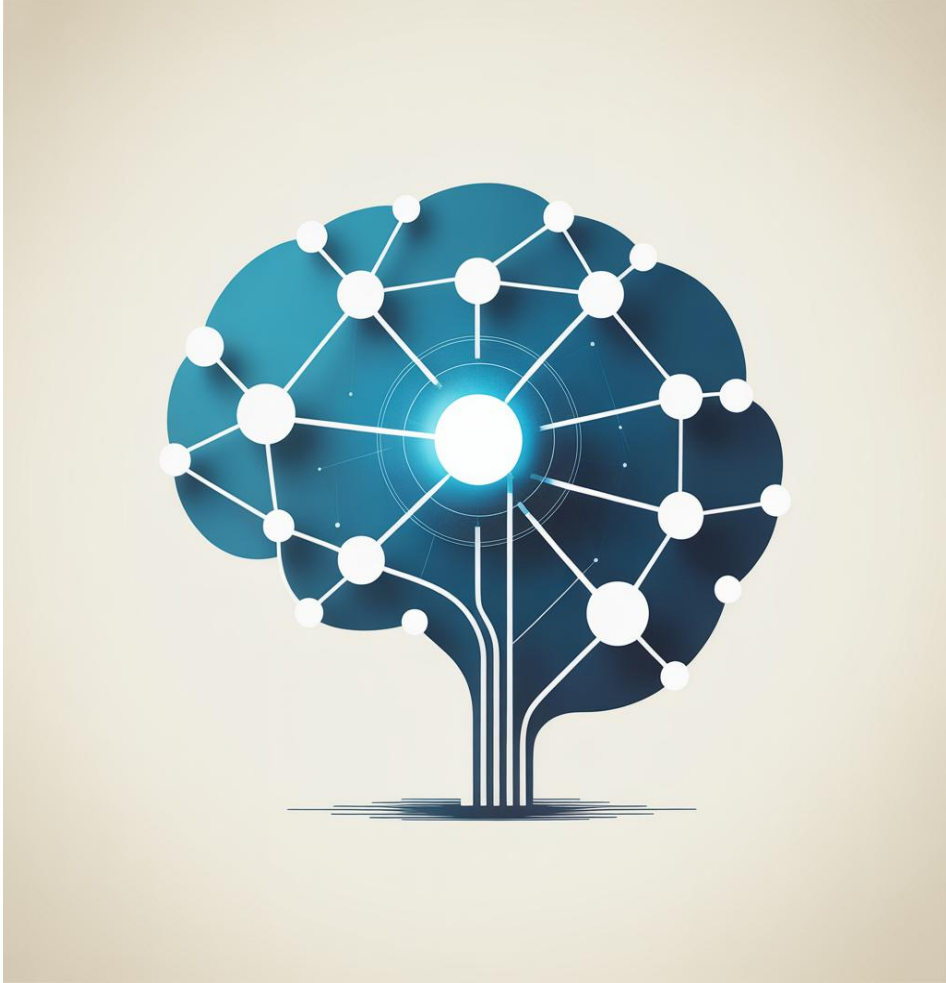
- 1 Find this session in the Cisco Live Mobile App
- 2 Click “Join the Discussion”
- 3 Install the Webex App or go directly to the Webex space
- 4 Enter messages/questions in the Webex space

Webex spaces will be moderated by the speaker until 14 November 2025.



<https://ciscolive.ciscoevents.com/ciscolivebot/#BRKCCT-2556>

Agenda



- 01 **REST / OAuth Primer**
- 02 **Custom Nodes in WxConnect**
- 03 **Custom Node Use Cases**
- 04 **A2A / MCP Primer**
- 05 **Roadmap for A2A / MCP**

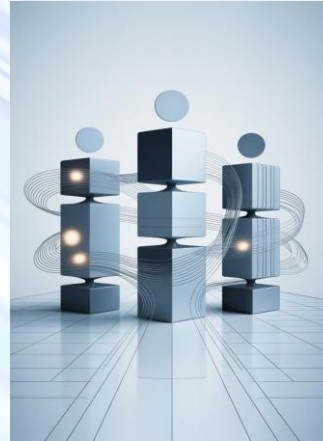
About the Presenter – Rue Green

- Sr. Solutions Engineer, Full Stack Software Developer
- Quad CCIE #9269
 - (Enterprise, Collaboration, Voice, Automation)
 - Author of Unified CVP Cisco Press book (2011)
- 20+ years CC Experience
 - WxCC, WxCCE, CPaaS
- 30+ years in software development
 - Certified Full Stack Developer (MERN, MEAN)
- 20 years at Cisco
- Professional Beach Volleyball Player and Coach



REST and OAuth Primer

A Quick Introduction to Modern API Architecture and Security



REST Fundamentals

REST (Representational State Transfer) is an architectural style for building APIs that leverages standard HTTP methods to create predictable, scalable web services. It's the foundation of most modern web APIs, enabling clean separation between client and server.

GET

Retrieve data from the server without modifying anything

```
GET /api/users/123
```

POST

Create new resources on the server

```
POST /api/users
```

PUT/PATCH

Update existing resources with new data

```
PUT /api/users/123
```

DELETE

Remove resources from the server

```
DELETE /api/users/123
```

Core Principles

- **Stateless:** Each request contains all necessary information; the server doesn't maintain session state
- **Standard Status Codes:** Uses HTTP codes like 200 OK, 404 Not Found, 401 Unauthorized for clear communication
- **Resource-Based URLs:** Endpoints use nouns representing resources, not verbs describing actions
- **Client-Server Separation:** Clear boundaries enable independent evolution of frontend and backend

What is OAuth?

OAuth 2.0 is an industry-standard authorization framework that allows applications to securely access user resources without requiring users to share their passwords. It's the technology behind "Sign in with Google," "Connect with Facebook," and similar delegated authorization patterns.



The Problem OAuth Solves

Imagine you want to let a photo printing app access your Google Photos. Without OAuth, you'd have to give them your Google password – risky and uncontrollable. OAuth provides a secure alternative: grant limited, revocable access without exposing credentials.

Real-World Scenario

01

You want to print photos from Google Photos using PrintMaster app

03

You control exactly what PrintMaster can access and can revoke access anytime

02

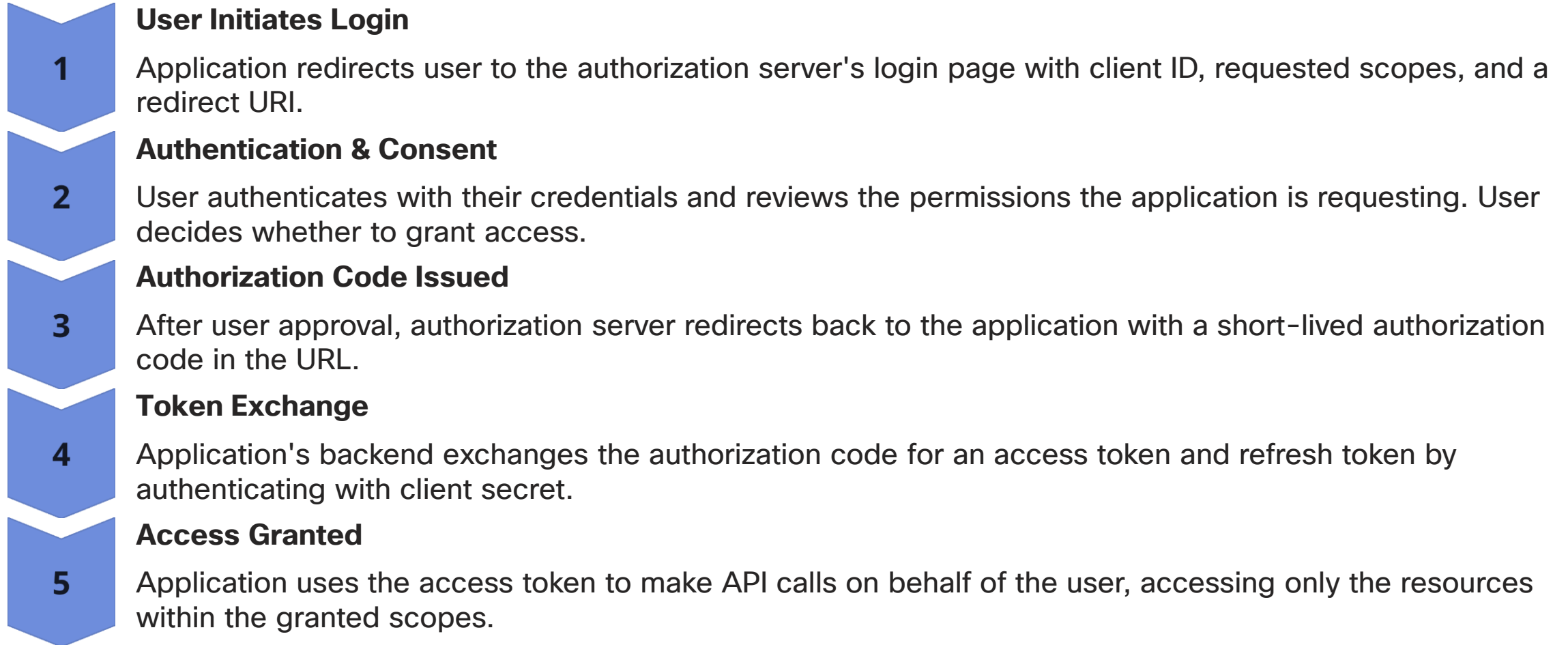
Instead of sharing your Google password, OAuth enables secure delegation

04

PrintMaster never sees your password and Google maintains security

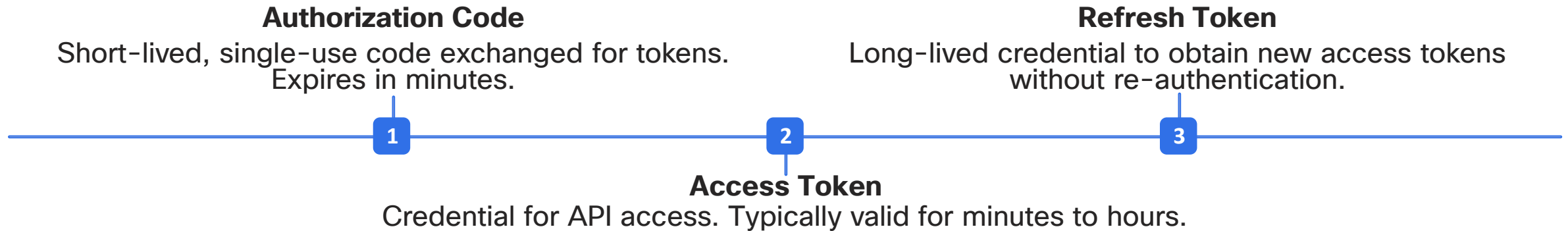
The OAuth Authorization Flow

The OAuth authorization code flow is the most secure method for web applications. It ensures that sensitive credentials never pass through the browser and provides a robust mechanism for delegated authorization.



Critical OAuth Values and Tokens

OAuth relies on several key values that flow through the authorization process. Each serves a specific security purpose and has different lifespans and usage patterns.



Understanding Each Component

Access Token

The credential sent with every API request to prove authorization. Included in the Authorization header as a Bearer token:

```
Authorization: Bearer eyJhbGc...
```

Short lifespan limits damage if compromised.

Refresh Token

Allows obtaining new access tokens without requiring the user to log in again. Must be stored securely on the backend, never in browser storage. Treat refresh tokens like passwords.

Client ID & Secret

Client ID: Public identifier for your application, safe to expose.

Client Secret: Sensitive password proving your app's identity. Must remain on backend only, never in frontend code or version control.

Scopes: Fine-Grained Permissions

Scopes define the specific permissions your application requests. Examples include `read:user`, `write:files`, or `admin:org`. Always follow the principle of least privilege – request only the minimum scopes necessary for your application's functionality.

REST + OAuth in Action

When REST and OAuth work together, you get secure, scalable API access. REST provides the structure for API calls, while OAuth ensures only authorized users can access protected resources. Here's how they integrate in practice.

Your Application

Client application needs to access user data via REST API

Authenticated Request

Makes REST API call with Bearer token in Authorization header

API Server Validation

Validates token signature, expiration, and checks user permissions

JSON Response

Returns requested data in standard REST format with appropriate status code

Example API Call

```
GET /api/user/profile HTTP/1.1Host: api.example.comAuthorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...Accept: application/json
```

The Complete Picture

REST defines HOW to structure your API calls using standard HTTP methods and resource-based URLs.

OAuth defines HOW to secure access to those APIs using token-based authorization instead of sharing passwords.

Together, they enable the secure, scalable, modern applications that power today's web.

Key Takeaways



REST Provides Structure

Architectural style using standard HTTP methods, stateless communication, and resource-based URLs. Foundation for building scalable, maintainable APIs.



OAuth Provides Security

Authorization framework enabling delegated access through tokens instead of passwords. Flow: Code → Access Token + Refresh Token, controlled by scopes.



Together They Power Modern APIs

REST and OAuth combine to create the secure, scalable architecture behind every major platform: Google, Facebook, GitHub, Stripe, and thousands more.

Every modern API you interact with relies on these two technologies working in harmony to deliver functionality and security.

Understanding REST and OAuth isn't just theoretical knowledge – it's essential for building any application that integrates with external services or exposes its own API to partners and developers.

Custom Node Creation

Building Reusable Integrations in Webex Connect



What Are Custom Nodes?

Definition

Custom Nodes allow you to create reusable integrations with third-party REST or SOAP APIs that can be used across multiple flows in Webex Connect.

Key Benefits

- Reusability across flows
- Centralized configuration
- Advanced authentication support
- Rate limiting controls

When to Use

- Need reusable integration nodes
- Require OAuth 2.0 authentication
- Complex API requirements
- Multiple flows use same API

Availability

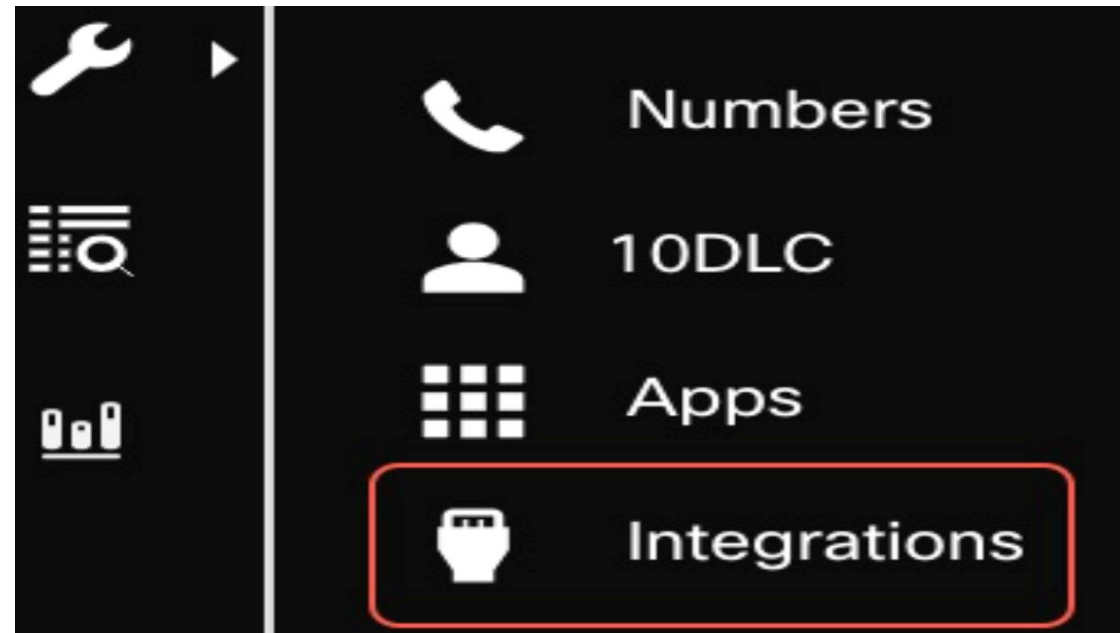
Custom Node integration is an [add-on feature](#) available in select Webex Connect tiers. Contact your account manager to enable it.

Step 1: Navigate to Integrations

Navigation Path

Assets → Integrations

This will take you to the Integrations screen where you can view all existing inbound/outbound integrations and create new custom nodes.



Step 2: Add a Custom Node

Click ADD INTEGRATION and select CUSTOM NODE

What Happens Next?

A "Create new custom node" pop-up window will appear where you can choose between REST API and SOAP API integration types.

Click ADD INTEGRATION and select CUSTOM NODE

Integrations

Setup integrations with different CRMs, contact center tools, REST and SOAP APIs.

Integration

Authorization

Q Search Integrations

Integration Type

All Integrations

Integration	Name(281)	Services	Integration type/environment
	2 way delivery notification Integration ID: F90000XY20	Service -	Inbound Webhook
	Outbound Call Trigger 2 Integration ID: OG0000XYZ31	Service -	Inbound Webhook
	Pharma Medicines Order Confirmation Integration ID: AS0000XYQ41	Services - Curbside Pickup Voice Notifications , Experiments	Inbound Webhook

Add Authorization

Add Integration

Inbound Webhook

Outbound Webhook

Custom Event

Webex Connect Bot

imlengage (Deprecated)

Enghouse (Deprecated)

Third-party Bot

Skype for Business (Deprecated)

Cisco ECE (Deprecated)

Custom Node

Select Custom Node from the Add Integration dropdown

Step 3: Configure Basic Node Settings

Create new custom node

Upload Node
Icon(SVG)

Minimum of 64x64px

Node Name ⓘ

Enter node name

Rest API

Soap API

Description ⓘ

Enter node description

Node Category ⓘ

Select Node Category

Creation Type ⓘ

Create blank Integration

Cancel

OK

Create new custom node configuration dialog

Required Fields

01

Node Name

Provide a user-friendly name for your integration

02

Description

Brief description of what the node does

03

SVG Icon

Upload a relevant SVG image for visual identification

04

API Type

Choose REST API or SOAP API

Step 4: Configure Request Details

Basic Settings

- **Request Name:** Descriptive method name
- **HTTP Type:** GET, POST, PUT, DELETE
- **Resource URL:** API endpoint URL

Parameters

- **URL Parameters:** Query string values
- **Parameter Type:** Static or Dynamic
- **Field Values:** Variable references

Request Details

Request Name

Request Timeout (Ms)

Connection Timeout (Ms)

Type

Post

Resource URL

Parse Variables

Step 5: Configure Authentication

Example: Basic Authentication configuration

Basic Auth

Configure username and password fields with static or dynamic values

API Key

Simple token-based authentication in headers

OAuth 2.0

Authorization Code or Client Credentials grant types

AWS Signature

HMAC-based authentication for AWS services

Type

Basic Auth

Username

username

Password

password

Parameter Value Type

Dynamic

Field Name

username

Parameter Value Type

Dynamic

Field Name

password

Step 6: Configure Headers

Parameter	Parameter Value Type	Parameter Value
Content-type	Static	application/json

+ Add New

Setting up request headers

Common Headers

- **Content-Type:** application/json, text/xml
- **Authorization:** Bearer tokens, API keys
- **Accept:** Expected response format
- **Custom Headers:** API-specific requirements



Pro Tip

Use **Static** for fixed values and **Dynamic** for runtime values

Step 7: Configure Request Body

Setting up JSON request body with dynamic parameters

Body Configuration Options

```
{  "text": "${text}",  "version": "${version}"}
```

Use **\$(variableName)** syntax to reference dynamic parameters provided at runtime. Mark variables as mandatory or optional in Node UI.

Body

Text (text/plain) ▾

```
1 {  
2   "text": "${text}"  
3 }  
4
```

+ Add Parameters

Parse

Parameter	Parameter Value Type	Field Name	
text	Dynamic ▾	text	🗑️

+ Add New

Setting up JSON request body with dynamic parameters

Step 8: Configure Response Handling

Configuring node events and success/error conditions

Node Events

- **Success Event:** HTTP Status = 200
- **Error Event:** HTTP Status ≠ 200
- Map to Success/Error node edges

Response Parameters

- Extract values using JSON paths
- Example: \$.data.user.email
- Available as output variables

Response

Text (text/plain) ▾

A node event signifies outcome of a node. Create a node event by specifying the response path. One or more responses can be mapped to an event.

Configure Node Events

Node Event	Body	Response Path	Condition	Value	Node Edge
success	HTTP Status ▾		equals ▾	200	Success ▾
errore	HTTP Status ▾		not equals ▾	200	Error ▾

+ Add New

Configuring node events and success/error conditions

Step 9: Map Response Parameters

Extracting specific values from API response using JSON paths

JSON Path Examples

Parameter: ToneIdSource: BodyResponse Path: \$.document_tone.tones[0].tone_id
Parameter: ToneNameSource: BodyResponse Path: \$.document_tone.tones[0].tone_name
Parameter: ToneScoreSource: BodyResponse Path: \$.document_tone.tones[0].score

A response object can be used in the node by saving it as a node variable.

Set data to be returned in a flow session

Parameter Name	Body	Response Path
<input type="text" value="ToneId"/>	<div>Body</div>	<div>\$.document_tone.tones[0]tone_id</div>
<input type="text" value="ToneName"/>	<div>Body</div>	<div>\$.document_tone.tones[0]tone_name</div>
<input type="text" value="ToneScore"/>	<div>Body</div>	<div>\$.document_tone.tones[0]score</div>

+

 Add New

Extracting specific values from API response using JSON paths

Step 10: Configure Node UI

Designing the user interface for the custom node

Field Types

- Text box – Single line input
- Text area – Multi-line input
- Dropdown – Selection list
- Checkbox – Boolean values

Field Configuration

- Information Text – Label/section header
- Mandatory – Required fields
- Default Values – Pre-populated data
- Help Text – User guidance

The screenshot displays the 'Node UI' configuration tab in a settings application. At the top, there are two tabs: 'Settings' and 'Node UI'. Below the tabs, a text instruction reads: 'Click and drag to move groups or parameters. Parameters can be moved across Groups as well as in a Group'. A red box highlights the 'Information Text ⓘ' label, which is positioned above a text input field containing the word 'Authentication'. To the right of this field is a blue button labeled '+ Add New Field Group'. Below the 'Authentication' field, there is a section titled 'Ungrouped' which contains a list of parameters. The first parameter is 'username', which has a dropdown arrow next to it. Below 'username', there are three configuration options: 'Field Type' (set to 'Text box' with a dropdown arrow), 'Regex Validation' (with an empty text input field and a 'Test regex' button), and 'Mandatory Parameter' (which is checked with a blue checkbox and highlighted by a red box). At the bottom of the 'Ungrouped' section, there is a 'Tooltip' label followed by an empty text input field.

Designing the user interface for the custom node

OAuth 2.0 Authentication Setup

Configuration Steps

01

Select OAuth 2.0

In the Custom Node configuration, choose **OAuth 2.0** as the authentication method

02

Choose Grant Type

Authorization Code: User-context operations (requires user login)
Client Credentials: Server-to-server operations (machine-to-machine)

03

Configure OAuth Settings

Enter Client ID, Client Secret, Authorization URL, Token URL, and required scopes from your API provider

04

Set Callback URL

Use the Webex Connect provided callback URL in your third-party application's OAuth configuration

05

Authorize the Integration

Navigate to **Assets → Integrations → Manage**, find your custom node, and complete authorization

Managing Node Authorizations

Authorization Lifecycle

- **Pending:** Authorization not yet completed
- **Authorized:** Successfully authenticated and ready
- **Expired:** Token needs refresh or re-authorization

Available Actions

- **View:** Review authorization details
- **Update:** Modify configuration
- **Make Default:** Set as default
- **Remove:** Delete authorization

Important Considerations



Token Expiration

If credentials are updated in the integrating system, you must re-authorize the node



✓ Callback URLs

Updated to Webex Connect branded URLs.
Ensure firewalls don't restrict access



Token Refresh

OAuth 2.0 tokens auto-refresh, but may need re-auth if refresh token expires

Testing Your Custom Node

Test Configuration

Before saving, use the built-in test feature to verify your custom node works correctly.

- Provide sample input values
- Check for HTTP 200 response
- Verify response parameters
- Review any error messages

Success Indicators

✓ **Successful Test**

Response status 200, all expected parameters returned, no errors in logs

✗ **Common Issues**

- Incorrect endpoint URL
- Missing auth credentials
- Invalid JSON paths
- Wrong parameter types

Best Practices & Tips

Design

- Use descriptive names for nodes and parameters
- Create reusable nodes for common integrations
- Document your custom nodes thoroughly
- Upload clear, recognizable SVG icons

Security

- Use OAuth 2.0 when available
- Never hardcode credentials in flows
- Configure security certificates when required
- Monitor authorization expiration dates

Performance

- Enable rate limiting to prevent API throttling
- Use appropriate timeout values
- Test thoroughly before making live
- Monitor transaction logs for errors

Maintenance

- Keep track of certificate expiry dates
- Update callback URLs when needed
- Review and update scopes regularly
- Test after any API changes

Key Takeaways

1 Reusability

Custom nodes enable you to build once and use across multiple flows, saving time and ensuring consistency

2 Flexibility

Support for REST and SOAP APIs with multiple authentication methods including OAuth 2.0

3 Testing

Always test your custom nodes before deployment to ensure proper configuration and response handling

4 Management

Monitor authorizations, maintain certificates, and keep your integrations up-to-date

Use Cases

Advanced REST and Custom Node Use Cases



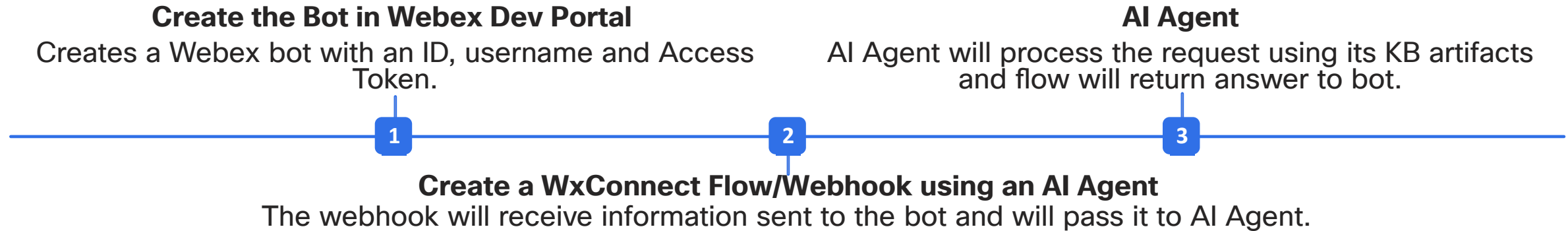
Webex RFP Bot

Integrating Webex with AI Agent



Creating a Webex RFP Bot

The following are the steps required to build a Webex Bot and integrate it with an AI Agent in WxConnect.



Understanding Each Component

Creating the Webex Bot

First step to create the Webex bot integration that we will register the webhook with for processing questions to the Webex bot.

The registration is done via POSTMAN, there is no UI current to do this!

Create the WxConnect Flow / Webhook

This webhook will be provided a payload that contains the BOT ID, with other information so the message can be extracted in the flow. The flow will also make additional API calls to extract the message that was sent to the Webex BOT.

This message is then passed into the AI Agent node in the flow for processing against its KB artifacts.

AI Agent

Processes the message extracted earlier using its KB artifacts that have been provided when the AI Agent was created. The answer is then then sent back via a Webex API call to the Webex Bot for deliver to the user.

Step 1: Create a Webex Bot

New Bot

Bot name*

Name of your bot as it will appear in Webex.

My KB Bot

Bot username*

The username users will use to add your bot to a space. Cannot be changed later.

FancyKBBot @webex.bot

✓ FancyKBBot@webex.bot is available

Icon*

Upload your own or select from our defaults. Must be exactly 512x512px in JPEG or PNG format.



Edit

App Hub Description*

What does your app do, how does it benefit users, how do users get started? Does your app require a non-Webex account? If your app is not free or has additional features for paid users, please note that and link to pricing information. 1024 character limit.

[Link](#) [List](#) [Table](#) [Media](#)

My Fancy KB Bot

Step 1: Create a Webex Bot (Continued)

Congratulations! 🎉

My KB Bot is one step closer to becoming a reality.

My KB Bot

👉 **Next Step:** Use your Bot Access Token to set up your webhook and finish building your bot.

Bot access token

Non-expiring (good for 100 years) access token for your bot. Save this token to set up your webhook.

NGZkMDE3NDQtNDZjOS00ODEwLWJkNGMtMWVvKn2MyNDU5ODg5MTR

Copy Token

💡 **Tip:** Save this token!
It won't be shown again (but you can regenerate a new one if needed).

Helpful resources to build bots

Bot name*

Name of your bot as it will appear in Webex and Webex App Hub.

My KB Bot

Edit

Bot username*

The username users will use to add your bot to a space. Cannot be changed later.

FancyKBBot@webex.bot

Icon*

Upload your own or select from our defaults. Must be exactly 512x512px in JPEG or PNG format.



Edit

App Hub Description*

What does your app do, how does it benefit users, how do users get started? Does your app require a non-Webex account? If your app is not free or has additional features for paid users, please note that and link to pricing information. 1024 character limit.

🔗 ⌵ ⌵ ⌵

My Fancy KB Bot

Edit

Bot ID

Unique system generated ID for your bot.

Y2lzY29zcGFyazovL3VzL0FQUEXJQ0FUSU9OLzNiNzYzNTkwLTlhYzEtNDM3Yi05YWZkLW

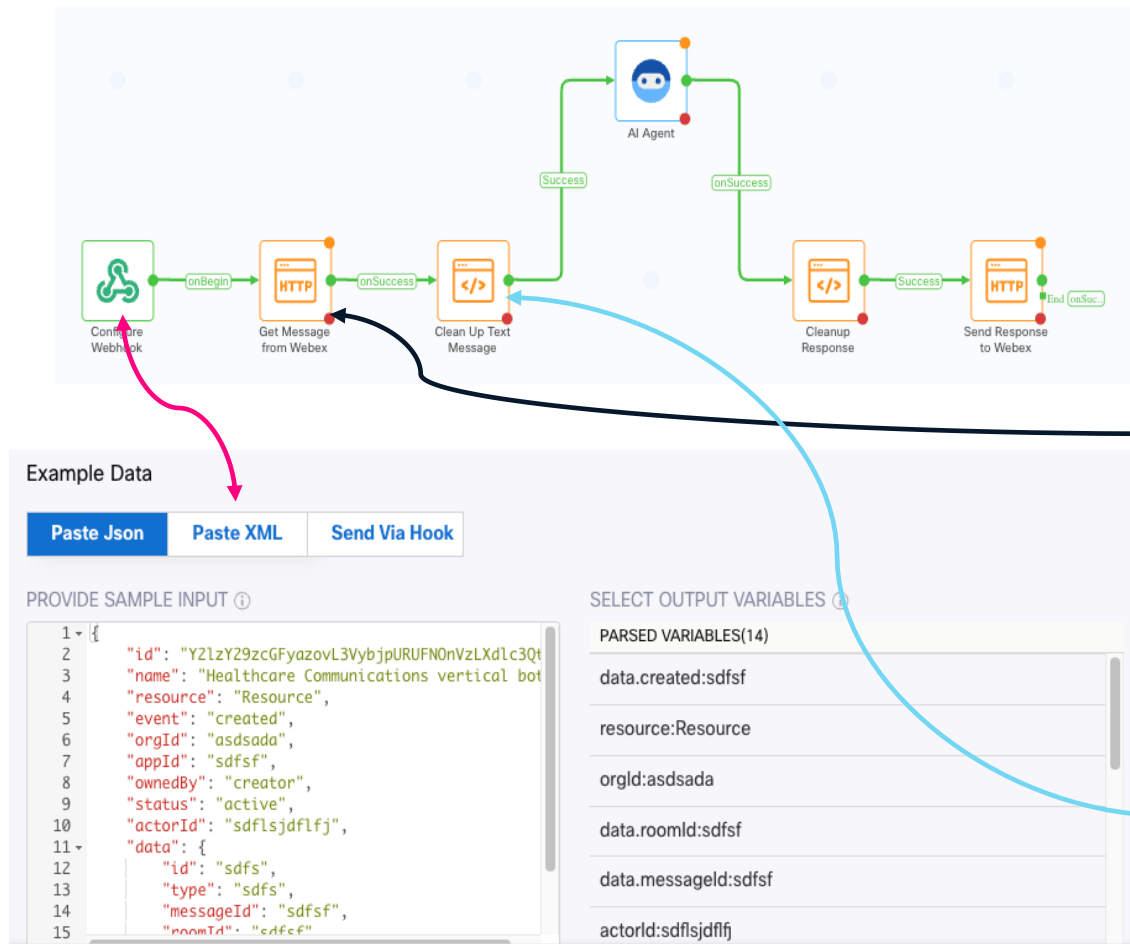
Copy

Delete Bot

Deleting a bot cannot be undone.

Delete Bot

Step 2: Create a WxConnect Flow/Webhook



Get Message from Webex

Configuration Transition Actions (Optional)

Make GET, POST, PUT, PATCH and DELETE requests to your application

Method: GET Endpoint URL: `https://webexapis.com/v1/messages/${n2.inboundWebhook.data.id}`

Header: Authorization Value: Bearer `$(Bot_Token)`

Content-Type: application/json

+ Add Another Header

Connection Timeout: e.g., 1000 ms Request Timeout: e.g., 1000 ms

Proxy Address (Optional): e.g., 192.128.19.243:80

Output Variables: JSON XML

Output Variable Name: messageText Response Entity: Body Response Path: text

+ Add Variable

Clean Up Text Message

Configuration Transition Actions (Optional)

Configure node on-enter / on-leave operations

Action 1

Time: On-enter Action: Set variable

Variable: inboundMessage Value: `$(n10.messageText)`

+ Add

+ Add Action

Step 2: Register the Webhook with Webex Bot (Continue)

POST

https://webexapis.com/v1/webhooks

Send

ParamsAuthorizationHeaders (10)BodyScriptsSettingsCookies

HeadersHide auto-generated headers

	Key	Value	Description		Bulk Edit	Presets
☒	Cache-Control	no-cache				
☒	Postman-Token	<calculated when request is sent>				
☒	Content-Type	application/json				
☒	Content-Length	<calculated when request is sent>				
☒	Host	<calculated when request is sent>				
☒	User-Agent	PostmanRuntime/7.48.0				
☒	Accept	/*/*				
☒	Accept-Encoding	gzip, deflate, br				
☒	Connection	keep-alive				
☒	Authorization	Bearer {{MYBotToken}}				

POST

https://webexapis.com/v1/webhooks

Send

ParamsAuthorizationHeaders (10)BodyScriptsSettingsCookies

noneform-datax-www-form-urlencodedrawbinaryGraphQLJSON

SchemaBeautify

```
1 {
2   "name": "MY KB Bot Flow",
3   "targetUrl": "https://hooks.us.webexconnect.io/events/68UQV9KG23",
4   "resource": "messages",
5   "event": "created",
6   "filter": "mentionedPeople=me"
7 }
8
9
```

Step 3 : Process Webex Bot message with AI Agent

AI Agent

ConfigurationTransition Actions (Optional)

Method Name

Process message

Send message to bots configured via Bot builder and use their responses to take actions

AGENT TYPE

Autonomous

AGENT

Only the agents accessible for you in the AI agent studio will appear here. Contact your tenant owner for access to other agents.

MESSAGE

\$(inboundMessage)

LANGUAGE

Select the expected language of incoming messages. Choose "Auto-detect" if you do not know or expect multiple languages to be used

Customer Details

CHANNEL

Live Chat / In-App Messaging

UNIQUE ID

\$(n2.inboundWebhook.transId)

Customer Parameters (Optional)

+ ADD New

Message Parameters (Optional)

+ ADD New

Input Variables

List of variables available as input for this node

Q Search

Custom Variables

[F59777]

Start

Node ID: 2

HTTP Request

Node ID: 10

Evaluate

Node ID: 67

Cleanup Response

Help

ConfigurationTransition Actions (Optional)

Configure node on-enter / on-leave operations

Action 1

Time

On-enter

Action

Set variable

Variable

message

Value

\$(n71.TextResponse)

+ Add

Cleanup Response

Help

ConfigurationTransition Actions (Optional)

Describe the code you wish to write in English and click "Generate code" to have AI generate a relevant code snippet for you. You can keep doing this multiple times by iterating on the prompt. Check out our docs for some sample prompts and best practices for prompts to get the most useful output.

Generate Code using AI

Enter JavaScript to perform logic and return values as output. Configured script outputs will be available as node outcome.

1 includeJs(imi_strings);
2 var temp_s = message;
3 //temp_s = temp_s.replace(":", " ");
4 //temp_s = temp_s.replace(",", " ");
5 temp_s = temp_s.replace(/\\n/g, '');
6 temp_s = temp_s.replace(/\\n/g, '');
7 //temp_s = temp_s.replace("-", " ");
8 var message = IMI_STRINGS.replace(/[^a-z0-9\\s,\\.\\-_?=&%#@]/gi, "", temp_s);
9 1;

Configure Script Output

Script Output

1

Branch Name

Success

+ Add New

© 2025 Cisco and/or its affiliates. All rights reserved.

BRKCCT-2556

CISCO

Step 3 : Return message to Webex Bot (Continued)

 Send Response to Webex [Help](#)

[Configuration](#) [Transition Actions \(Optional\)](#)

Make GET, POST, PUT, PATCH and DELETE requests to your application

Method

POST

Endpoint URL

https://webexapis.com/v1/messages

Header

Value

Authorization

Bearer \${Bot_Token}

✕

Content-Type

application/json

✕

[+ Add Another Header](#)

Body

```
{
  "roomId": "${n2.inboundWebhook.data.roomId}",
  "markdown": "${message}"
}
```

Connection Timeout ⓘ

Request Timeout ⓘ

e.g., 1000 ms

e.g., 1000 ms

Proxy Address (Optional) ⓘ

e.g., 192.128.19.243:80

Output Variables

☒ JSON ☐ XML

[Import From Sample](#)

[+ Add Variable](#)

Response

The response body will be stored in `n43.http.responseBody` variable.

Custom CRM Node

Example of a Custom CRM node



Custom CRM Node Review – GET

Q Search

+ Add Method

GET

Fetch CRM Data

PATCH

Update Delivery...

PATCH

Update SafeZone

PATCH

Cancel Order

Settings

Node UI

Request Details

Request Name (required)

Fetch CRM Data

request Timeout (ms)

Connection Timeout (ms)

Type

Get

Resource URL (required)

https://cjpdata.wbx.ninja?number=\${variable}

Parse Variables

Parameter (required)

variable

Parameter value type (required)

Dynamic

Field Name (required)

Phone Number

Authorization

Type

No Auth

Security Configuration

Throttling

On enabling, an additional outcome throttleLimitReached" will be available with this custom node.

Rate Limit

Limit total number of request per second.

Concurrency Limit

Limit maximum number of parallel custom node executions.

Q Search

+ Add Method

GET

Fetch CRM Data

PATCH

Update Delivery...

PATCH

Update SafeZone

PATCH

Cancel Order

Settings

Node UI

Click and drag to move groups or parameters. Parameters can be moved across Groups as well as in a Group

Information Text (required)

+ Add New Field Group

Ungrouped

Phone Number

Fetch CRM Data

Configuration

METHOD NAME

Fetch CRM Data

Phone Number

13032223333

Response

Body

Headers

Fetch CRM Data

Node ID: TEST

Cancel

Test

© 2025 Cisco and/or its affiliates. All rights reserved.

BRKCCT-2556

Custom CRM Node Review – Patch

Q Search

+ Add Method

GET Fetch CRM Data

PATCH Update Delivery...

PATCH Update SafeZone

PATCH Cancel Order

Settings

Node UI

Request Details

Request Name (required)

Update Delivery Date

request Timeout (ms)

Connection Timeout (ms)

Type

Patch

Resource URL (required)

https://cjpdata.wbx.ninja?number=\${variable}

Parse Variables

Parameter (required)

variable

Parameter value type (required)

Dynamic

Field Name (required)

Phone Number

Authorization

Type

No Auth

Security Configuration

Throttling

On enabling, an additional outcome throttleLimitReached" will be available with this custom node.

Rate Limit

Limit total number of request per second.

Concurrency Limit

Limit maximum number of parallel custom node executions.

Q Search

+ Add Method

GET Fetch CRM Data

PATCH Update Delivery...

PATCH Update SafeZone

PATCH Cancel Order

Settings

Node UI

Click and drag to move groups or parameters. Parameters can be moved across Groups as well as in a Group

Information Text (required)

+ Add New Field Group

Ungrouped

Phone Number

Updated Delivery Date

Update Delivery Date

Configuration

METHOD NAME

Update Delivery Date

Phone Number

13032223333

Updated Delivery Date

07/12/2026

Response

Body

Headers

Status: Time: ms

Update Delivery Date

Node ID: TEST

Cancel

Test

Advanced Custom Node Review

Old JDS custom node example



JDS Node Review – GET (Check Identity)

Q Search

+ Add Method

GET

Token Service - ...

GET

Check Identity

POST

Create Events

POST

Create Identity

DELETE

Delete Identity

PATCH

Add Alias

PATCH

Delete Alias

POST

Test

Settings

Node UI

Request Details

Request Name (required)

Check Identity

request Timeout (ms)

Connection Timeout (ms)

Type

Get

Resource URL (required)

https://api.wxcc-us1.cisco.com/admin/v1/api/person/workspace-id/\${workspaceId}/aliases/\${identity}

Parse Variables

Parameter (required)

workspaceld

Parameter value type (required)

Static

Parameter value (required)

67e2e90ec4b84e752570056c

Parameter (required)

identity

Parameter value type (required)

Dynamic

Field Name (required)

Phone Number or Email address

Authorization

Type

OAuth 2.0

Grant Type

Authorization Code

Consumer ID (required)

Cbc29dd59d8bc18b7f461e35fdae0e904851b9a8c60450aaff16c32ead56t

Consumer Secret (required)

983e0473e6541952cf6a495bb4fc3d2a13ef028910c765703dfddfdb3aa8

Call Back URL (required)

https://ciscoroadshowandciscolive.us.webexconnect.io/callback

Authorization URL (required)

https://webexapis.com/v1/authorize

Scope

cjp:config cjp:config_read cjp:config_write

Access Token URL

https://webexapis.com/v1/access_token

☐ The access token has a limited validity

Validity

1209599

Refresh Token URL

https://webexapis.com/v1/access_token

client authentication

Send client credentials in body

Advance Settings

Get Access Token

Access Token (required)

YmMzYzQ1ZWltMzlkYS00OGJILWFhM2EtMjUyODFiMjM5ZTk3ZjhjMzdjMjEtI

Refresh Token

YWU2NTc1NjEtOTc0Ni00NGJkLTk1NGMtODgwN2NkZmVjMDA4MzcwOWI0

Validity

1209599

Response

Import from sample

JSON (application/

A node event signifies outcome of a node. Create a node event by specifying the response path. One or more responses can be mapped to an event.

Configure Node Events

Node Event (required)	Body	Response Path (required)	Condition	Value (required)	Node Edge
Profile_Found	HTTP Status		equals	200	Success
Profile_Not_Found	HTTP Status		equals	404	Success
AuthorizationError	HTTP Status		equals	400	Error

+ Add New

A response object can be used in the node by saving it as a node variable.

Set data to be returned in a flow session

Parameter Name (required)	Body	Response Path (required)
personId	Body	\$.data[0].id

+ Add New

A2A / MCP Primer

Understanding Agent-to-Agent Communication
and Model Context Protocol



What is Agent-to-Agent (A2A)?

Agent-to-Agent communication represents a fundamental shift in how AI systems operate. Rather than functioning in isolation, AI agents can now interact, collaborate, and coordinate with each other autonomously to accomplish complex tasks that would be impossible for a single agent.



Autonomous Communication

AI systems engage in direct dialogue with each other without requiring human intermediation, enabling real-time decision making and coordination



Structured Protocols

Standardized methods for exchanging information, delegating tasks, and maintaining consistent communication patterns across diverse AI systems



Cross-Platform Coordination

Different AI models from various vendors working together seamlessly, sharing context and collaborating on unified objectives



Complex Workflows

Multi-step problem solving where agents divide work, execute specialized tasks, and synthesize results into comprehensive solutions

- ❏ **Real-world analogy:** Like microservices in software architecture, A2A enables specialized AI agents to work together as a cohesive system, each contributing their unique capabilities to accomplish complex goals.

Model Context Protocol (MCP)

The Model Context Protocol is an open standard developed by Anthropic that fundamentally changes how AI applications connect to data sources and tools. MCP eliminates the need for custom integrations by providing a universal interface that any AI model can use to access resources consistently and securely.



Universal Connector

A standardized protocol that allows AI models to connect to any data source using the same interface, regardless of the underlying technology



Tools & Data Access

Models can access resources, execute functions, and utilize templates consistently across different platforms and environments



No Custom Integrations

Eliminates the need for building one-off connections between each AI model and data source, dramatically reducing development time



Open Standard

An open-source protocol developed by Anthropic for universal adoption, fostering innovation and preventing vendor lock-in

- ❏ **Think of MCP as USB for AI** – just as USB allows any device to connect to any computer, MCP allows any AI model to connect to any data source or tool through a single, standardized interface.

How MCP Works: Architecture Overview



MCP operates through a three-tier architecture that separates concerns and enables flexible, secure connections between AI applications and their data sources. This design ensures scalability while maintaining simplicity.

1

MCP Host (Application)

The client application that wants AI capabilities – such as Claude Desktop, IDEs, or custom business applications

2

MCP Client (Connector)

The connector layer that maintains protocol connections and manages communication between the host and servers

3

MCP Server

Lightweight programs that expose specific capabilities to AI models through the standardized MCP protocol

4

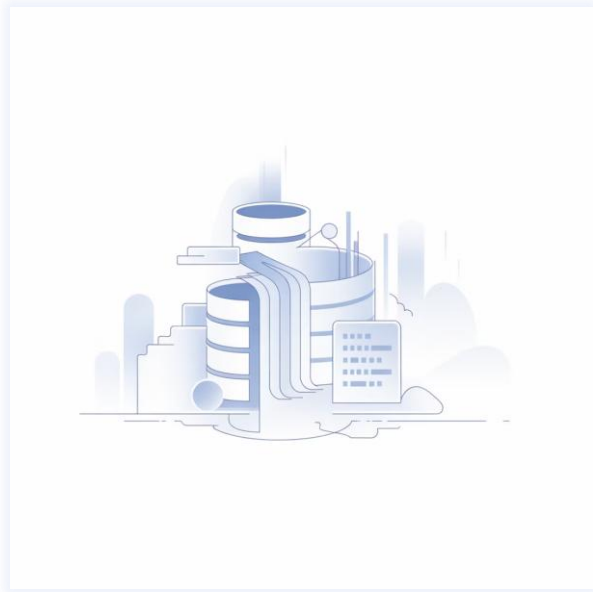
Resources, Tools & Prompts

The actual data, functions, and templates that the AI model can access and utilize through the server

MCP Components Explained

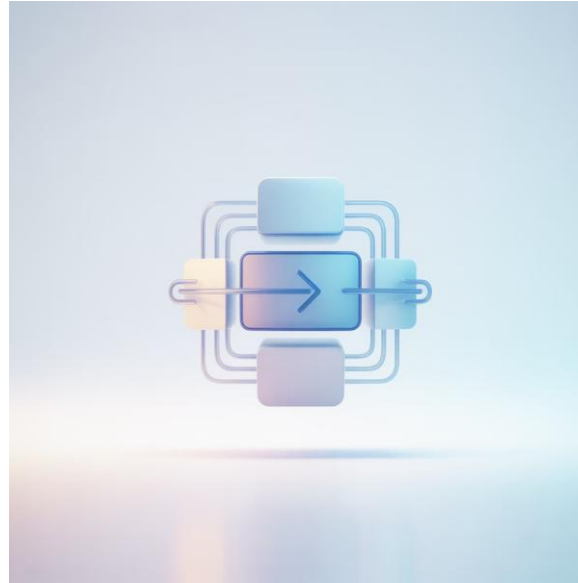
MCP servers expose three primary types of capabilities to AI models, each serving a distinct purpose in enabling intelligent, contextual interactions with external systems and data.

Resources



Data sources that models can read and analyze, including files, database contents, API responses, and live system data. Resources provide the context that models need to make informed decisions.

Tools



Executable functions that models can invoke to perform actions, such as creating records, sending notifications, running queries, or triggering workflows. Tools extend the model's capabilities beyond text generation.

Prompts



Reusable templates that standardize common tasks and queries. Prompts ensure consistency and best practices when interacting with specific data sources or performing routine operations.

❏ **Communication Protocol:** MCP uses JSON-RPC 2.0 for message exchange, supporting both stdio (standard input/output) and HTTP with Server-Sent Events (SSE) for real-time updates.

Benefits of A2A with MCP

The combination of Agent-to-Agent communication and the Model Context Protocol creates a powerful ecosystem that transforms how organizations deploy and manage AI capabilities at scale.

For Organizations

- Dramatically reduced integration complexity across diverse AI systems and platforms
- Accelerated deployment timelines for new AI capabilities and features
- Enhanced data security with centralized access control and governance
- Vendor-agnostic architecture that prevents lock-in and enables best-of-breed solutions
- Lower total cost of ownership through reusable integrations

For Developers

- Write integration code once, connect to any compatible AI model
- Build libraries of reusable components and integrations
- Clear separation of concerns between data sources and AI logic
- Thriving open source ecosystem with community-contributed servers
- Simplified testing and debugging with standardized protocols

📌 **Key Advantage:** Organizations can build AI integrations that work seamlessly across multiple platforms and models without vendor lock-in, future-proofing their AI investments.

Key Takeaways

The combination of Agent-to-Agent communication and Model Context Protocol creates a powerful, flexible ecosystem that is transforming how we build and deploy AI systems. Here's what you need to remember:

A2A Enables Autonomy

Agent-to-Agent communication allows AI systems to work together autonomously, coordinating complex workflows without human intervention and achieving outcomes impossible for single agents.

MCP Provides the Standard

Model Context Protocol delivers the universal language for AI-to-data connections, eliminating custom integrations and enabling any model to access any resource through a consistent interface.

Together They Scale

Combined, A2A and MCP create a flexible, scalable AI ecosystem that adapts to changing needs, supports diverse use cases, and grows with organizational requirements.

Open Standards Win

Open standards drive innovation, reduce vendor lock-in, and foster thriving ecosystems where developers can build, share, and benefit from community contributions.

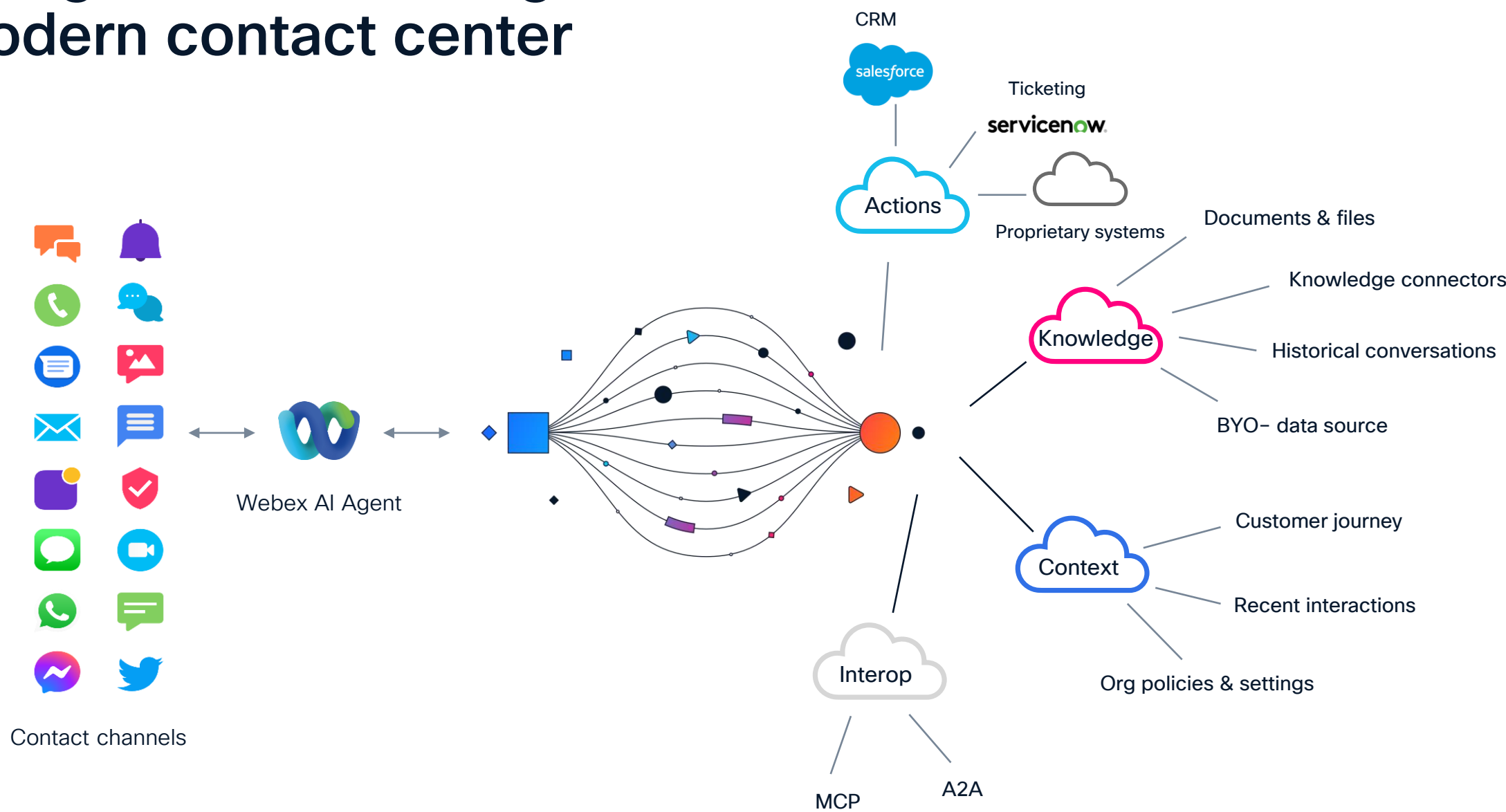
- ❏ **The Big Picture:** The combination of A2A and MCP represents a fundamental shift in how we build and deploy AI systems – moving from isolated, custom-built solutions to an interconnected ecosystem of specialized agents working together through open standards.

Roadmap

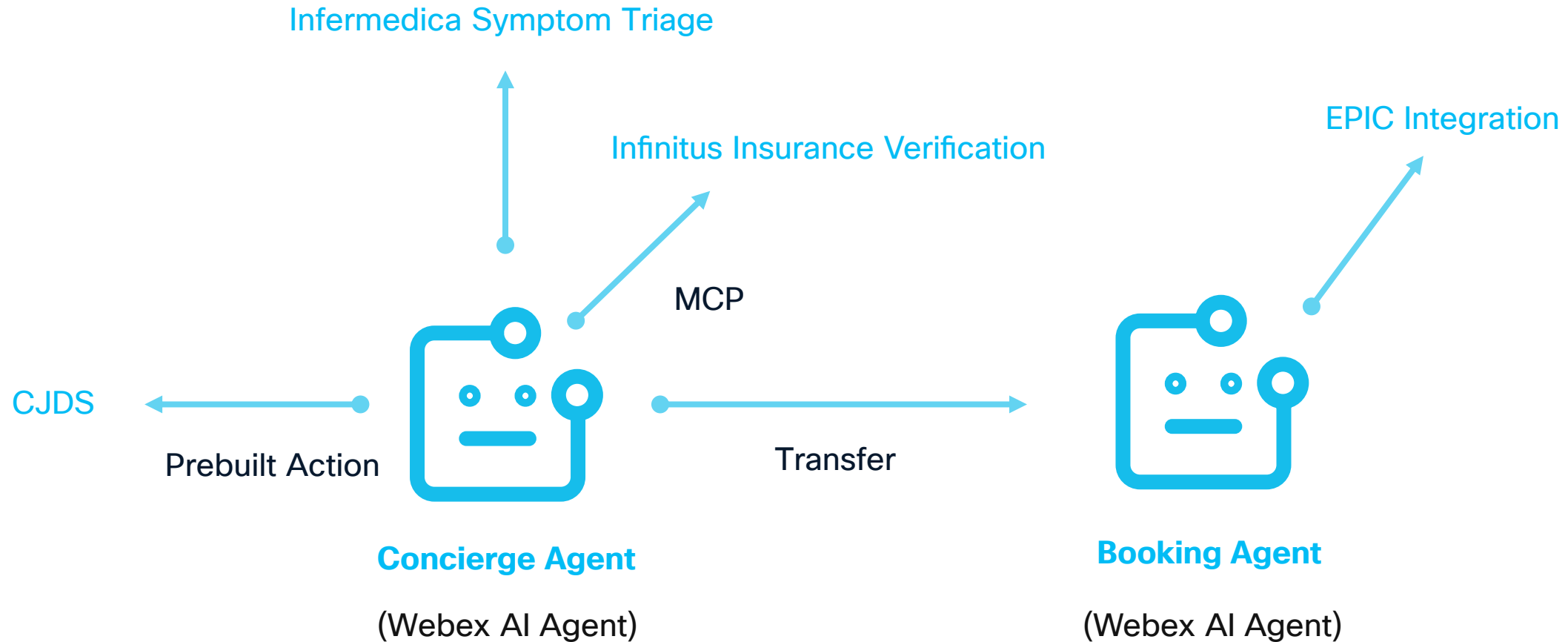
Cisco Roadmap for A2A and MCP Support



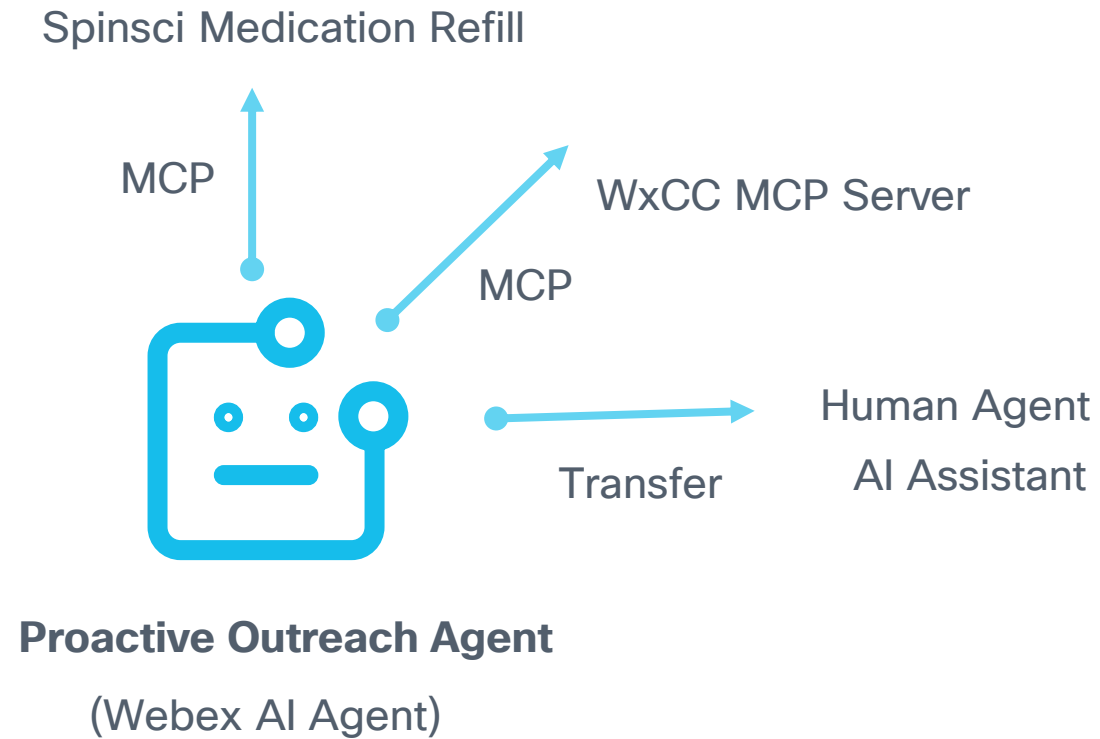
An agentic and intelligent front door for the modern contact center



Engage via Contact Center



Proactive Engagement



New Webex AI Agent Studio: AI Agent Configuration

webex AI Agent Studio

Dashboard

AI Agents

AI Assistant Skills

Knowledge

Integrations

AI Engine

Renegize Healthcare Assistant

Draft

Last updated on 05/25/2025 by Henry Riggs

Preview

Configuration

Sessions

History

Analytics

Profile

Knowledge

Actions

Languages

Profile

Essential details

Agent name *

Renegize Healthcare Assistant

System ID *

RenegizeHealthcareAssistant-uah13as

URL for agent profile image *

https://us.webexbotbuilder.com/static/assets/default_logo

Instruction

Choose your AI engine and refine the goal and instructions to boost your agent's accuracy.

AI engine *

Webex AI Pro 1.0

Agent's goal *

Assist users with doctor appointments by efficiently guiding them through the necessary steps.

Insert example

Instructions

Enter step-by-step instructions for your agent to accomplish its goal.

Insert example

Insert action

Welcome message *

Hi there, how can I help you?

Renegize Healthcare

New Webex AI Agent Studio: Pre-Interaction

webex AI Agent Studio

Dashboard

AI Agents

AI Assistant Skills

Knowledge

Integrations

AI Engine

Renegize Healthcare Assistant

Draft

Last updated on 05/25/2025 by Henry Riggs

Configuration

Sessions

History

Analytics

Profile

Knowledge

Actions

Languages

Pre-interaction

Mid-interaction

Post-interaction

Pre-Interaction actions

Action name

Created by

Description

Last updated

Action type

Provider type

Controls

Find record

Henry Riggs

Search and locate specific customer records across connected databases and systems using MCP server integration.

10/13/25, at 1:29 PM

Fulfillment

MCP

Add actions

Add pre-interaction actions

Search by action name, description, or provider name

Create new

All

Integration

MCP

A2A

CJDS: Get customer detail

From Webex | Integration

Find record

From Salesforce | MCP

Send one-time PIN

From Duo | Integration

Verify one-time PIN

From Duo | Integration

Cancel

Add

© 2025 Cisco and/or its affiliates. All rights reserved.

BRKCCT-2556

New Webex AI Agent Studio: Mid-Interaction

webex AI Agent Studio

Dashboard

AI Agents

AI Assistant Skills

Knowledge

Integrations

AI Engine

Renegize Healthcare Assistant

Draft

Last updated on 05/25/2025 by Henry Riggs

Configuration

Sessions

History

Analytics

Profile

Knowledge

Actions

Languages

Pre-interaction

Mid-interaction

Post-interaction

Mid-interaction actions

Add actions

Action name	Created by	Description	Last updated	Action type	Provider type	Controls
<div>Create case</div>	Henry Riggs	Generate new customer service cases in connected CRM systems using MCP server framework.	10/13/25, at 1:10 PM	Fulfillment	MCP	<div></div> <div></div>

Add mid-interaction actions

Search by action name, description, or provider name

Create new

All

Integration

MCP

A2A

Cancel reservation

From Airbnb | MCP

Check availability

From Google Calendar | MCP

Consult with "Symptom Triage" agent

From Infermedica | A2A

Create case

From Salesforce | MCP

Cancel

Add

New Webex AI Agent Studio: Post-Interaction

webex AI Agent Studio

Dashboard

AI Agents

AI Assistant Skills

Knowledge

Integrations

AI Engine

Renegize Healthcare Assistant

Draft

Last updated on 05/25/2025 by Henry Riggs

Configuration

Sessions

History

Analytics

Profile

Knowledge

Actions

Languages

Pre-interaction

Mid-interaction

Post-interaction

Post-Interaction actions

Add actions

Action name	Created by	Description	Last updated	Action type	Provider type	Controls
<div>AI Assistant: Generate AutoCSAT</div>	Henry Riggs	Enable intelligent assistant capabilities through connected AI service integrations.	10/13/25, at 1:38 PM	Fulfillment	Integration	<div><div></div><div></div></div>
<div>AI Assistant: Analyze call sentiment</div>	Henry Riggs	Enable intelligent assistant capabilities through connected AI service integrations.	10/13/25, at 1:38 PM	Fulfillment	Integration	<div><div></div><div></div></div>

Add post-interaction actions

Search by action name, description, or provider name

Create new

All

Integration

MCP

A2A

AI Assistant: Analyze call sentiment

From Webex | Integration

AI Assistant: Generate AutoCSAT

From Webex | Integration

AI Assistant: Generate summary

From Webex | Integration

CJDS: Update customer detail

From Webex | Integration

Cancel

Add

Renegize Healthcare

New Webex AI Agent Studio: Create a new transfer

Search by action name, description, or provider name

Create new

All

Integration

MCP

A2A

CJDS: Get customer detail

From Webex | Integration

Find record

From Salesforce | MCP

Send one-time PIN

From Duo | Integration

Verify one-time PIN

From Duo | Integration

Cancel

Add

Transfer

Consult

Fulfillment

Create a transfer action

Set when to transfer, who to transfer to, and what to share with the next agent.

General information

Action name *

Agent transfer

Transfer condition *

Add in custom condition to trigger agent transfer.

For example: If verification fails three times, transfer to an agent.

Select how you want to assign an agent *

Select an AI agent

Select an AI agent *

Select an agent from your organization

Transfer visibility

☒

Skip greetings and match voice

☐

Set custom logic to assign a human or AI agent

Information to share

For faster, more accurate result, choose what to share with the next agent. If you skip this, we'll just include a summary of the conversation.

Entities

See JSON schema

Add new entity

Entity name	Type	Value	Description	Example	Required
No entities added yet					

New Webex AI Agent Studio: Create a new consult

Search by action name, description, or provider name

Create new

All

Integration

MCP

A2A

CJDS: Get customer detail

From Webex | Integration

Find record

From Salesforce | MCP

Send one-time PIN

From Duo | Integration

Verify one-time PIN

From Duo | Integration

Cancel

Add

Transfer

Consult

Fulfillment

Create a consult action

Set when to consult, which agent to consult with, and what to share for collaborative decision making.

General Information

Action name *

Agent consult

Consult condition *

Add in custom condition to trigger agent consultation.

For example: when patient symptom are complex, consult with specialist agent.

Consult with AI agent *

Select an agent *

Select an agent

Information to share

For faster, more accurate result, choose what to share with the consulting agent. If you skip this, we'll just include a summary of the conversation.

Entities

See JSON schema

Add new entity

Entity name	Type	Value	Description	Example	Required
No entities added yet					

New Webex AI Agent Studio: Create a new fulfillment

The image shows the Webex AI Agent Studio interface. On the left, the 'Add pre-interaction actions' panel is open, displaying a search bar and a list of actions: 'CJDS: Get customer detail' (From Webex | Integration), 'Find record' (From Salesforce | MCP), 'Send one-time PIN' (From Duo | Integration), and 'Verify one-time PIN' (From Duo | Integration). A pink arrow points from the 'Fulfillment' option in the 'Create new' dropdown to the 'Update action' modal on the right.

The 'Update action' modal is titled 'Update action' and has the subtitle 'Enable your AI agent to connect with external systems and perform more complex tasks.' It contains the following sections:

- General information**
 - Action name ***: A text input field with the placeholder 'Enter action name' and a close button (X).
 - Action description ***: A text area with the placeholder 'Enter action description'.
 - Action scope ***: Radio buttons for 'Slot filling' and 'Fulfillment' (selected).
- Fulfillment**
 - Select a fulfillment approach ***: Radio buttons for 'Webex Connect Flow Builder' (selected) and 'Custom server'. Below 'Webex Connect Flow Builder' is the text 'Build workflows using drag and drop interface with Webex Connect platform.' Below 'Custom server' is the text 'Connect to your own server or external API endpoint for custom logic processing.'
 - Service ***: A dropdown menu with the selected value 'Wrap up conversations'.
 - Flow ***: A dropdown menu with the selected value 'Generate summary'.

Complete your session evaluations



Complete a minimum of 4 session surveys and the Overall Event Survey to claim a Cisco Live T-Shirt.



Earn up to 800 points by completing all surveys and climb the Cisco Live Challenge leaderboard.



Level up and earn exclusive prizes!



Complete your surveys in the Cisco Live Events app.

Continue your education



Visit the Cisco Stand for related demos



Book your one-on-one Meet the Expert meeting



Attend the interactive education with Capture the Flag, and Walk-in Labs



Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand

Thank you

CISCO Live !

