Cisco Webex WFO Data Import/Export Reference Guide

First Published: July 10, 2020
Last Updated: August 18, 2020

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0882

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS. THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

Contents

Introduction

You can import and integrate historical and real-time data from any automatic contact distributor (ACD) to the Webex WFO database using one of the following tools.

- **GIS API**. The GIS API is used to import both historical and real-time service queue and agent productivity data to the Webex WFO database at specified intervals. Webex WFO uses this agent data to generate distributions, forecasts, reports, and statistics.
- **Workforce Management Historical Import Tool (WHIT)**. WHIT is used to convert existing historical data from a non-Webex WFO system into data files that can be loaded into Webex WFO service queues. You might want to use WHIT if you are installing a new Webex WFO system.

GIS API

The GIS API is part of Webex WFO and requires no separate installation or executable to function.

> **NOTE** File paths shown in this document are the default file path. Webex WFO can be installed to a custom location, so your file path might differ.

Historical Data Capture

The following tasks are required to import historical data from an ACD to the Webex WFO database:

1. Write scripts that retrieve the historical data from the ACD.
2. Create a batch process to run the scripts at specified intervals to retrieve the historical data and convert it to the required comma-delimited text file format.
3. Place the comma-delimited text files in a predefined folder on the Webex WFO Data Server using FTP or some other tool.

The GIS API processes the comma-delimited text files and writes the historical data to the Webex WFO database.

The following graphic is a high-level depiction of the historical data integration process.

Requirements

The following are conditions that must be met in order to assure successful call and non-call data capture.

- Ensure that the format used in the ACD agent IDs that are synced from the ACD to Webex WFO is used whenever sending historical and real-time data to Webex WFO. If there is a case mismatch (for example, the agent ID in the ACD is **JSMITH** but in the data import file it is **Jsmith**), the data import will fail.
- Service historical data is calculated from the summation of agent service productivity data. As a result, both the AgentProductivity.AGENT and ServiceHistorical.SERVICE files must be available for a given period for capture to take place. If both files are found, but data is missing from one, certain values might be taken to be zero (the sum of zero items equals zero). It is also important

that the agent productivity and service historical data are consistent in terms of which agents and services have data.

- Only one file of each type should be written per historical period per ACD. A second file of the same type for the same period will cause data from the first file to be overwritten.
- The same ACD number should be used in the file name and file headers for all files for that ACD.

File Usage

Use the three capture files as follows.

File	Use
.SERVICE	This file is required and used for service queue statistics.
.AGENT	This file is used to report agent productivity data for each service queue the agents works with. The file is required in order to process the files for the given intervals. It is not necessary for the file to contain any data rows, only the header row. However, if you want to track individual agent productivity, then it should include data rows.
.EVENT	This file is optional. It can be used as one method to display the agent real-time state and capture real-time events to calculate agent conformance and adherence. The GIS API processes this file as soon as it drops into the Reports folder. The frequency of new files dropping into the Reports folder determines how close to real time the data is. **NOTE** The preferred method of capturing real-time events is through the use of the Real-Time Data Capture API. See Real-Time Data Capture API for more information.

Text File Locations

The comma-delimited text files must be placed in the location configured on the Data Server Configuration page. If a location is not configured on the Data Server Configuration page, then the text files must be placed in the following default location on the Data Server:

```
C:\Program Files\Common Files\Webex WFO\Data Server\gis\reports\<tenant>
```

After the text files are processed they are archived in the Cloud and the Data Server for troubleshooting purposes.

ACD Numbering

ACDs from which data is imported are assigned numbers to identify them as the data source. The ACD numbers are assigned by Webex WFO on the ACD Configuration page (see the section, “Connection Settings for Generic ACDs”).

Even if there is only one ACD, that ACD must be assigned a number. In the examples in this document, the ACD is assigned the number 1.

Additionally, the agent identifiers (ACD ID in Webex WFO) and service queue identifiers (Service Queue ID in Webex WFO) in GIS text files must be unique

Text File Details

This section provides the details of the three comma-delimited text files used by the GIS API. These files are the following:

- AgentProductivity.AGENT
- ServiceHistorical.SERVICE
- AgentState.EVENT

The data in the files is by 30-minute intervals. This interval is not configurable.

NOTE The date format in the text files (YYYY-MM-DD) might be changed by a spreadsheet application such as Microsoft Excel to the format that is set in the application as the default. Be aware of this possibility and make sure that the date columns are configured so that the date format is correct.

Data Type Definitions

The data types noted in this section have specific definitions. These definitions are detailed in the following table.

Data Type	Definition
date	A date expressed in YYYY-MM-DD format.
double	A floating point value expressed in a manner that can be parsed as a signed double-precision floating point value. Any value allowed as a

Data Type	Definition
	long is also allowed as a double. The + sign should be omitted.
integer	An integer expressed as a signed decimal integer value, without leading zeros, whose value is at least –0x8000 and at most 0x7fff. The + sign should be omitted.
long	An integer expressed as a signed decimal integer value, without leading zeros, whose value is at least –0x80000000 and at most 0x7fffffff. The + sign should be omitted.
nonnegative double	A floating point value expressed in a manner that can be parsed as a signed double-precision floating point value, whose value is at least 0x00000000. Any value allowed as a non-negative long is also allowed as a non-negative double. The + sign should be omitted.
nonnegative long	An integer expressed as a signed decimal integer value, without leading zeros, whose value is at least 0x00000000 and at most 0x7fffffff. The + sign should be omitted.
null	The literal value NULL (case insensitive).
positive double	A floating point value expressed in a manner that can be parsed as a positive signed double-precision floating point value. Any value allowed as a positive long is also allowed as a positive double. The + sign should be omitted.
positive long	An integer expressed as a signed decimal integer value without leading zeros, whose value is at least 0x00000001 and at most 0x7fffffff. The + sign should be omitted.
time	A time expressed in any of the following formats: ■ hh:mm ■ hh:mm:ss ■ hh:mm:ss.sss
timestamp	A time stamp consisting of the following data types:

Data Type	Definition
	`<date> <time> <tzoffset>`
tzoffset	A time zone offset expressed in +HHMM, -HHMM, HHMM format or the literal values GMT or UTC.

AgentProductivity.AGENT File

The AgentProductivity.AGENT file contains agent productivity data by 30-minute intervals from 00:00 to 23:59.

File Name Format

`<date/time><tzoffset>_<ACD#>_AgentProductivity.AGENT`

Element	Description
<date/time>	The date and time the file was generated, in YYYYMMDDHHMM format. Example: 201301151430 (14:30 on January 15, 2013). Time is in 24-hour format.
<tzoffset>	The time zone (where the ACD is located) date/time offset from GMT in AHHMM format, where: ■ A is either P (positive) or N (negative), characterizing the offset from GMT ■ HH is the number of offset hours ■ MM is the number of offset minutes **EXAMPLE** If the ACD is in Greenwich Mean Time (GMT), then the <tzoffset> is P0000. If the ACD is in Eastern Standard Time (EST) (GMT–0500), then the <tzoffset> is N0500.
<ACD#>	The ACD number (see ACD Numbering).

File Header

`AGENT DATE: <date> INTERVAL: <time> TZOFFSET: <tzoffset> ACD: <ACD#>`

Element	Description
<date>	The date the file was generated, in YYYY-MM-DD format.
<time>	The beginning of the 30-minute interval covered by the data, in HH:MM 24-hour format.
<tzoffset>	The time zone (where the ACD is located) offset from GMT, in +HHMM, -HHMM, or HHMM format. If plus or minus is not specified, the offset is assumed to be plus (positive).
<ACD#>	The number of the ACD that is the source of the data.

The file header must be the first non-blank line in the file.

> **NOTE**
> The AGENT DATE, INTERVAL, and TZOFFSET values must represent a period start time that matches those in the file name as well as a historical period supported by Webex WFO. It is not required that the individual values match, but only to mean the same point in time. For example, if the file is named:
>
> `201301231800N0500_1_AgentProductivity.AGENT`
>
> and the header is:
>
> `AGENT DATE: 2013-01-23 INTERVAL: 23:00 TZOFFSET: 0000 ACD: 1`
>
> there is no error, because each is expressing the same point in time in different time zones. The time 18:00 in the time zone N0500 (a 5-hour negative offset from GMT) is the same as the time 23:00 in the GMT time zone.

Column Header

`<columnname1>,<columnname2>,<columnname3>. . .,<columnnameN>`

Where <columnname1> through <columnnameN> are the names of columns represented in the file.

Column names are not restricted to required and optional columns. You can add additional unrecognized columns and associated data (for example, agent names or service names) to make the GIS files more human-readable. The GIS API ignores these columns and their associated data.

Column order is not specified. The only requirement for column order is that the column names are in the same order as the data in each line.

Column names cannot contain commas, are case sensitive, and cannot start or end with a space (such spaces are automatically trimmed when the file is parsed). Column names must be unique. Duplicate column names (after space trimming) result in an error.

The column header must be the second non-blank line in the file.

Data Lines

`<columnvalue1>,<columnvalue2>,<columnvalue3>. . .,<columnvalueN>`

Where<columnvalue1> through <columnvalueN> are the values of <columnname1> through <columnnameN> for one row of data.

Each line of data corresponds to one data item within the file, with one value for each column, in the same order as the column header. The values are separated by commas. No value can contain a comma. Data values are trimmed of leading and trailing white space when parsed.

The third and subsequent non-blank lines in the file must be data lines. It is possible for a file to contain no data lines. This means there were no data items for that period.

Required Columns

The following table describes the columns that are required to be in the AgentProductivity.AGENT file.

> **NOTE**
> If the AgentProductivity.AGENT file describes a non-interactive service queue, some column names will not make sense. See the Description column for clarification.
>
> For non-interactive service queues, if columns do not apply, then set the values in these columns to 0 (zero).

Column	Data Type	Description
acdAgentId	string	The ID of the agent in the ACD.
acdServiceId	positive long	The ID of the service queue in the ACD. Can be alphanumeric.
contactsHandled	non-negative double	The number of contacts for the service queue handled by the agent during the period. For chats, this includes chats that have been dropped and chats that have been resolved.

Column	Data Type	Description
totalTalkSeconds	non-negative double	The total talk time on contacts for the service queue handled by the agent during the period. A contact's talk time can start in the previous period; the entire talk time is counted.
totalHoldSeconds	non-negative double	The total hold time on contacts for the service queue handled by the agent during the period. A contact's hold time can start in the previous period; the entire hold time is counted. This column is not applicable to chats. Enter zero for this column.
totalAfterContactWorkSeconds	non-negative double	The total after-contact work time on contacts for the service queue handled by the agent during the period. A contact's after-contact time can start in the previous period; the entire after-contact time is counted.
totalPeriodHandleTimeSeconds	non-negative double	The total time the agent spent handling contacts (talk, hold, work) for the service queue that occurred within the boundaries of the period. This metric (A) is differentiated from the sum of totalTalkSeconds + totalHoldSeconds + totalAfterContactWorkSeconds (B) in that it does not include time that exceeds the period boundary. **EXAMPLE** If a call is answered by the agent at 09:29 and the call ends

Column	Data Type	Description
		at 9:33 with no after-contact work, and if the ACD considers the call to be handled during the 09:30–10:00 period, then the call contributes 4 minutes to B (09:29–09:33) but only 3 minutes to A (09:30–09:33).
totalUnproratedReadyWaitingSeconds	non-negative double	The total Ready/Waiting time for the agent during the period, across all service queues in this ACD. A non-interactive service queue, like chat, can be Ready while the agent is answering chats as long as the agent is not handling the maximum number of chats.
totalUnproratedNotReady BusySeconds	non-negative double	The total Not Ready/Busy time for the agent during the period, across all service queues (not counting time while the agent is handling a contact or reported as Ready/Waiting on another service queue) in this ACD.
totalUnproratedInSession Seconds	non-negative double	The total in session time (logged in time) for the agent during the period, across all service queues in this ACD.

Optional Columns

The following table describes the columns that are optional in the AgentProductivity.AGENT file. If not included, the default value is used for the metric.

Column	Data Type	Description
periodStart	timestamp	A timestamp representing the start of the interval period. If included, this field must refer to the same point in

Column	Data Type	Description
		time (although not necessarily expressed in the same time zone) as the period start time in the file name and in the file header. If it does not match, an error occurs.
contactsTransferredOut	non-negative double	The number of contacts transferred out by the agent during the period. Default value = 0.
contactsTransferredIn	non-negative double	The number of contacts transferred in to the agent during the period. This field is reserved for future use. Currently the value is ignored.
contactsExternalIn	non-negative double	The number of external inbound contacts to that agent during the period. Default value = 0. This column is not applicable to chats. Enter zero for this column.
contactsExternalOut	non-negative double	The number of external outbound contacts from the agent during the period. Default value = 0. This column is not applicable to chats. Enter zero for this column.
totalReservedSeconds	non-negative double	The total reserved time for contacts for the service queue for the agent during the period. If non-zero, the reserved time is not counted as Ready/Waiting time. Default = 0. This column is not applicable to chats. Enter zero for this column.
totalProratedReadyWaitingsSeconds	non-negative double	The total prorated Ready/Waiting time for the agent and service queue. When totaled across all service queues in this ACD, this must be equal to the required field

Column	Data Type	Description
		totalUnproratedReadyWaitingSeconds. If the prorated values are omitted or set to zero, the Data Server divides the unprorated values equally among the service queues in this ACD for which the agent has agent-service queue productivity data during the period. If the prorated values are specified, and add up to the unprorated values, then the prorated values are used for the agent-service queue combination. If the prorated values are specified and do not add up to the unprorated values, then for each agent-service queue entry, the Data Server recalculates the prorated value as the unprorated value weighted by the specified prorated value divided by the sum of the specified prorated values for all of the service queues in this ACD for which the agent has agent-service queue data in the period. This is most useful for ACDs that assign Ready/Waiting time to an agent for one service queue and Not Ready/Busy time for other service queues. It is important that unprorated and prorated Not Ready/Busy times do not include times when the agent is accruing Ready/Waiting or other time for other service queues. For ACDs that report Not Ready/Busy time for service queues when the agent is handling calls or Ready/Waiting time in another service queue, it is important to remove all of the duplicate time and report Not Ready/Busy time only when an agent is Not Ready/Busy in all service queues simultaneously. For this type of ACD, the total unprorated

Column	Data Type	Description
		Ready/Waiting time is the sum of each service queue's Ready/Waiting time and the total unprorated Not Ready/Busy time is computed from the total in-session (logged-in) time less the total in-service (Talk, Hold, After Work, Ready/Waiting, and Reserved) time. A non-interactive service queue, like chat, can be Ready while the agent is answering chats as long as the agent is not handling the maximum number of chats.
totalProratedNotReadyBusySeconds	non-negative double	The total prorated Not Ready/Busy time for the agent and service queue. When totaled across all service queues in this ACD, this is equal to the required field totalUnproratedNotReadyBusySeconds. For a detailed explanation of this value, see the description for the field totalProratedReadyWaitingSeconds.
contactsAnswered	non-negative double	The number of contacts for the service queue answered by the agent during the period. The difference between this field and the required field contactsHandled is up to the implementer. The contactsHandled field is used for things such as computing average handle times, while contactsAnswered is used for computing average speed of answer. Some ACDs make a distinction between the two fields. For example, in some ACDs a handled contact is a contact that ends during the reporting interval, while an answered contact is a contact that is answered during the reporting interval.

Column	Data Type	Description
		This field is reserved for future use. Currently the value is ignored.
totalAnswerDelaySeconds	non-negative double	The total delay in answering contacts for the service queue for contacts the agent answered during the period. This field is reserved for future use. Currently the value is ignored.

Examples

The following are examples of AgentProductivity.AGENT files.

Sequential Contacts

In this example, an agent talks to two customers sequentially. This is typical of voice contacts.

	0:00	0:05	0:10	0:15	0:20	0:25	0:30
SQ1	Cust1	A for C1	Cust1	A for C1			
SQ2					Cust2	A for C2	

Cust1	Customer 1 talking
Cust2	Customer 2 talking
A for C1	Agent working with Customer 1
A for C2	Agent working with Customer 2

The GIS file resulting from these two contacts looks like this:

```
File name: 201701150000N0600_1_AgentProductivity.AGENT
Agent Date: 2017-01-15 INTERVAL: 00:00 TZOFFSET: -0600 ACD: 1
acdAgentId,acdServiceId,contactsHandled,contactsAnswered,
   contactsTransferredOut,contactsExternalIn,contactsExternalOut,
   totalTalkSeconds,totalHoldSeconds,totalAfterContactWorkSeconds,
   totalPeriodHandleTimeSeconds,totalUnproratedReadyWaitingSeconds,
   totalUnproratedNotReadyBusySeconds,totalUnproratedInSessionSeconds
5009,1,1,1,0,0,0,1200.000,0.000,0.000,1200.000,0.000,0.000,1800.000
5009,2,1,1,0,0,0,600.000,0.000,0.000,600.000,0.000,0.000,0.000
```

If the two contacts came in to the same service queue, the GIS file would look like this:

```
File name: 201701150000N0600_1_AgentProductivity.AGENT
```

```
Agent Date: 2017-01-15 INTERVAL: 00:00 TZOFFSET: -0600 ACD: 1
acdAgentId,acdServiceId,contactsHandled,contactsAnswered,
   contactsTransferredOut,contactsExternalIn,contactsExternalOut,
   totalTalkSeconds,totalHoldSeconds,totalAfterContact
   WorkSeconds,totalPeriodHandleTimeSeconds,totalUnprorated
   ReadyWaitingSeconds,totalUnproratedNotReadyBusySeconds,
   totalUnproratedInSessionSeconds
5009,1,2,2,0,0,0,3000.000,0.000,0.000,1800.000,0.000,0.000,1800.000
```

Simultaneous Contacts (Both Within Interval)

In this example, an agent handles two contacts at the same time within an interval. This is typical when the contacts are chat sessions.

	0:00	0:05	0:10	0:15	0:20	0:25	0:30
SQ1	Cust1	A for C1	Cust1		A for C1		
SQ2		Cust2		A for C2	Cust2	A for C2	

Cust1	Customer 1 talking
Cust2	Customer 2 talking
A for C1	Agent working with Customer 1
A for C2	Agent working with Customer 2

The GIS file resulting from these two contacts looks like this:

```
File name: 201701150000N0600_1_AgentProductivity.AGENT
Agent Date: 2017-01-15 INTERVAL: 00:00 TZOFFSET: -0600 ACD: 1
acdAgentId,acdServiceId,contactsHandled,contactsAnswered,
   contactsTransferredOut,contactsExternalIn,contactsExternalOut,
   totalTalkSeconds,totalHoldSeconds,totalAfterContactWorkSeconds,
   totalPeriodHandleTimeSeconds,totalUnproratedReadyWaitingSeconds,
   totalUnproratedNotReadyBusySeconds,totalUnproratedInSessionSeconds
5009,1,1,1,0,0,0,1500.000,0.000,0.000,900.000,0.000,0.000,1800.000
5009,2,1,1,0,0,0,1500.000,0.000,0.000,900.000,0.000,0.000,0.000
```

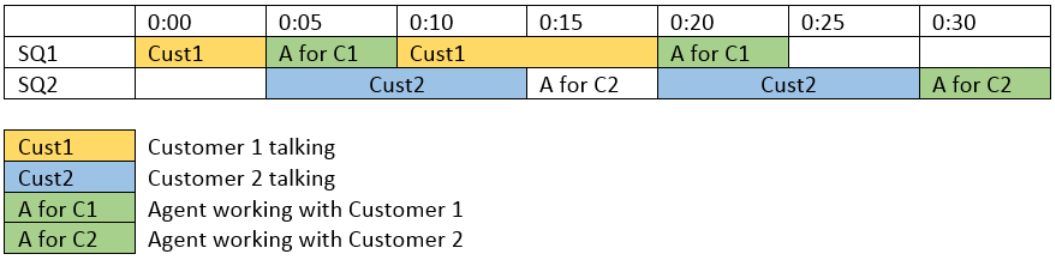
If the two contacts came in to the same service queue, the GIS file would look like this:

```
File name: 201701150000N0600_1_AgentProductivity.AGENT
Agent Date: 2017-01-15 INTERVAL: 00:00 TZOFFSET: -0600 ACD: 1
```

```
acdAgentId,acdServiceId,contactsHandled,contactsAnswered,
   contactsTransferredOut,contactsExternalIn,contactsExternalOut,
   totalTalkSeconds,totalHoldSeconds,totalAfterContactWorkSeconds,
   totalPeriodHandleTimeSeconds,totalUnproratedReadyWaitingSeconds,
   totalUnproratedNotReadyBusySeconds,totalUnproratedInSessionSeconds
5009,1,2,2,0,0,0,3000.000,0.000,0.000,1800.000,0.000,0.000,1800.000
```

Simultaneous Contacts (One Crosses Interval Boundary)

In this example, an agent handles two contacts at the same time, and the second contact extends past the interval boundary to end at 0:35.



	0:00	0:05	0:10	0:15	0:20	0:25	0:30
SQ1	Cust1	A for C1	Cust1		A for C1		
SQ2		Cust2		A for C2	Cust2		A for C2

Cust1	Customer 1 talking
Cust2	Customer 2 talking
A for C1	Agent working with Customer 1
A for C2	Agent working with Customer 2

Customer 1 begins chatting from 00:00 to 00:05. The agent responds to customer 1 from 00:05 to 00:10. Customer 2 begins chatting from 00:05 to 00:15. Customer 1 responds to chatting from 00:10 to 00:20. The agent responds to chatting with customer 2 from 00:15 to 00:20. The agent responds to chatting with customer 1 from 00:20 to 00:25, when their conversation is completed. Customer 2 responds to chatting from 00:20 to 00:30. The agent responds chatting to customer 2 from 00:30 to 00:35, when their conversation is completed.

The two GIS files resulting from these two contacts would like this:

```
File name: 201701150000N0600_1_AgentProductivity.AGENT
Agent Date: 2017-01-15 INTERVAL: 00:00 TZOFFSET: -0600 ACD: 1
acdAgentId,acdServiceId,contactsHandled,contactsAnswered,
   contactsTransferredOut,contactsExternalIn,contactsExternalOut,
   totalTalkSeconds,totalHoldSeconds,totalAfterContactWorkSeconds,
   totalPeriodHandleTimeSeconds,totalUnproratedReadyWaitingSeconds,
   totalUnproratedNotReadyBusySeconds,totalUnproratedInSessionSeconds
5009,1,1,1,0,0,0,1500.000,0.000,0.000,900.000,0.000,0.000,1800.000
5009,2,0,1,0,0,0,0.000,0.000,0.000,900.000,0.000,0.000,0.000
File name: 201701150030N0600_1_AgentProductivity.AGENT
```

```
Agent Date: 2017-01-15 INTERVAL: 00:30 TZOFFSET: -0600 ACD: 1
acdAgentId,acdServiceId,contactsHandled,contactsAnswered,
   contactsTransferredOut,contactsExternalIn,contactsExternalOut,
   totalTalkSeconds,totalHoldSeconds,totalAfterContactWorkSeconds,
   totalPeriodHandleTimeSeconds,totalUnproratedReadyWaitingSeconds,
   totalUnproratedNotReadyBusySeconds,totalUnproratedInSessionSeconds
5009,1,0,0,0,0,0,0.000,0.000,0.000,0.000,0.000,0.000,0.000
5009,2,1,0,0,0,0,1800.000,0.000,0.000,300.000,1500.000,0.000,1800.000
```

ServiceHistorical.SERVICE File

The ServiceHistorical.SERVICE file contains service queue data by 30-minute intervals from 00:00 to 23:59 for both interactive and non-interactive service queue types.

- Interactive service queue types consist of contacts in which agents and customers have real-time communication, such as voice (call) or chat.
- Non-interactive service queue types consist of contacts in which agents and customers do not have real-time communication, such as email, fax, and social media, as well as contact activities like stuffing envelopes.

File Name Format

```
<date/time><tzoffset>_<ACD#>_ServiceHistorical.SERVICE
```

Element	Description
<date/time>	The date and time the file was generated, in YYYYMMDDHHMM format. **EXAMPLE** 201701151430 (14:30 on January 15, 2017). Time is in 24-hour format.
<tzoffset>	The time zone (where the ACD is located) date/time offset from GMT in AHHMM format, where: ■ A is either P (positive) or N (negative), characterizing the offset from GMT ■ HH is the number of offset hours ■ MM is the number of offset minutes

Element	Description
	EXAMPLE If the ACD is in Greenwich Mean Time (GMT), then the <tzoffset> is P0000. If the ACD is in Eastern Standard Time (EST) (GMT–0500), then the <tzoffset> is N0500.
<ACD#>	The ACD number (see ACD Numbering).

File Header

```
SERVICE DATE: <date> INTERVAL: <time> TZOFFSET: <tzoffset> ACD: <ACD#>
```

Element	Description
<date>	The date the file was generated, in YYYY-MM-DD format.
<time>	The beginning of the 30-minute interval covered by the data, in HH:MM 24-hour format.
<tzoffset>	The time zone (where the ACD is located) offset from GMT, in +HHMM, -HHMM, or HHMM format. If plus or minus is not specified, the offset is assumed to be plus (positive).
<ACD#>	The number of the ACD that is the source of the data.

The file header must be the first non-blank line in the file.

NOTE

The SERVICE DATE, INTERVAL, and TZOFFSET values must represent a period start time that matches those in the file name as well as a historical period supported by WFM. It is not required that the individual values match, but only to mean the same point in time. For example, if the file is named:

```
201301231800N0500_1_ServiceHistorical.SERVICE
```

and the header is:

```
SERVICE DATE: 2013-01-23 INTERVAL: 23:00 TZOFFSET: 0000 ACD: 1
```

there is no error, because each is expressing the same point in time in different time zones. The time 18:00 in the time zone N0500 (a 5-hour negative offset from GMT) is the same as the time 23:00 in the GMT time zone.

Column Header

```
<columnname1>,<columnname2>,<columnname3>. . .,<columnnameN>
```

Where <columnname1> through <columnnameN> are the names of columns represented in the file.

Column names are not restricted to required and optional columns. You can add additional unrecognized columns and associated data (for example, agent names or service names) to make the GIS files more human-readable. The GIS API ignores these columns and their associated data.

Column order is not specified. The only requirement for column order is that the column names are in the same order as the data in each line.

Column names cannot contain commas, are case sensitive, and cannot start or end with a space (such spaces are automatically trimmed when the file is parsed). Column names must be unique. Duplicate column names (after space trimming) result in an error.

The column header must be the second non-blank line in the file.

Data Lines

```
<columnvalue1>,<columnvalue2>,<columnvalue3>. . .,<columnvalueN>
```

Where<columnvalue1> through <columnvalueN> are the values of <columnname1> through <columnnameN> for one row of data.

Each line of data corresponds to one data item within the file, with one value for each column, in the same order as the column header. The values are separated by commas. No value can contain a comma. Data values are trimmed of leading and trailing white space when parsed.

The third and subsequent non-blank lines in the file must be data lines. It is possible for a file to contain no data lines. This means there were no data items for that period.

Required Columns

The following table describes the columns that are required to be in the ServiceHistorical.SERVICE file for both interactive and non-interactive service queue types.

> **NOTE**
>
> If the ServiceHistorical.SERVICE file describes a non-interactive service queue, some column names will not make sense. See the Description column for clarification.
>
> For non-interactive service queues, if columns do not apply, then set the values in these columns to 0 (zero).

Column	Data Type	Description
acdServiceId	positive long	The ID of the service queue in the ACD. Can be alphanumeric.
contactsOffered	non-negative double	The number of contacts for the service queue offered to agents during the period.
contactsHandled	non-negative double	The number of contacts for the service queue handled by agents during the period.
contactsAnswered	non-negative double	The number of contacts for the service queue answered by agents during the period. The difference between this field and the required field contactsHandled is up to the implementer. The contactsHandled field is used for things such as forecasting numbers of contacts and computing average handle times, while contactsAnswered is used for computing average speed of answer. Some ACDs make a distinction between the two fields; for example, in some ACDs a handled contact is a contact that ends during the reporting interval, while an answered contact is a contact that is answered during the reporting interval.
contactsAbandoned	non-negative double	The number of contacts for the service queue abandoned during the period.
totalTalkSeconds	non-negative double	The total talk time on contacts for the service queue handled by agents during the period.
totalHoldSeconds	non-negative double	The total hold time on contacts for the service queue handled by agents during the period. This column is not applicable to chats. Enter zero for this column.
totalAfterContactWorkSeconds	non-negative double	The total after contact work time on contacts for the service queue handled by agents during the

Column	Data Type	Description
		period.
totalAnswerDelaySeconds	non-negative double	The total delay in answering contacts for the service for contacts agents answered during the period. This column is not applicable to chats. Enter zero for this column.
serviceLevelPercent	non-negative double ≤ 100	The total delay in answering contacts for the service for contacts agents answered during the period.

Optional Columns

The following table describes the columns that are optional in ServiceHistorical.SERVICE file. If not included, the default value is used for the metric.

Columns	Data Type	Description
periodStart	timestamp	A timestamp representing the start of the interval period. If included, this field must refer to the same point in time (although not necessarily expressed in the same time zone) as the period start time in the file name and in the file header. If it does not match, an error occurs.
contactsBlocked	non-negative double	The number of contacts blocked for the service queue during the period. Default value = 0.
serviceLevelSeconds	positive double	The service level threshold in seconds for the service queue during the period. This field is reserved for future use. Currently the value is ignored.
contactsInQueue	non-negative double	The maximum number of contacts in queue for the interval. Used only with non-interactive queues. For interactive queues, this value will always be zero no matter what value is entered in this column.

Missing Columns

Some columns, such as totalInServiceSeconds and totalInSessionSeconds, might appear to be missing. In fact, these are values that need to be prorated; they are computed by combining other historical service queue data with agent productivity data. This is done so that per-service queue values from historical service queue data add up to the same totals as the per-agent-per-service queue values from agent productivity data.

For example, in most ACDs, time in a Ready/Waiting state is not associated with a service queue. Rather, the agent is just in that state. In the AgentProductivity.AGENT file, this total is provided in each agent-service queue entry for the agent as totalUnproratedReadyWaitingTime. From that, we compute a proratedReadyWaitingTime for each service queue (the total unprorated value divided by the number of service queues for which the agent has agent-service queue productivity data during the period).

A similar approach is taken for totalUnproratedNotReadyBusySeconds, which is converted to propratedNotReadyBusySeconds. We use the prorated values to compute totals for a service queue. As a result, we don't count the same time twice (or more for multiple service queues) and the metrics are zero-sum.

If the totalProratedReadyWaitingTime and/or totalProratedNotReadyBusyTime values are specified, we use these as a guide to prorating the unprorated times. If they add up correctly, the specified prorated values are used. If not, we use them as a weight (using the specified prorated value divided by the sum of the specified prorated values across all the service queues for the agent during the interval) for computing the prorated values.

Implementers who want to have more control over how time is prorated, other than dividing by the number of service queues, can add the columns totalProratedReadyWaitingSeconds and totalProratedNotReadyBusySeconds to the AgentProductivity.AGENT file.

It is strongly recommended that you ensure that the sum of the totalProratedReadyWaitingSeconds values for each agent-service queue line for a given agent during a period adds up to the totalUnproratedReadyWaitingSeconds value in each agent-service queue line for that agent, and likewise for Not Ready/Busy time. It is also strongly recommended that you ensure that Not Ready/Busy time is never counted for a service queue if that time is concurrent with time in another state for other service queues.

For example, for ACDs that report Not Ready/Busy time for one service queue while an agent is handling calls in another service queue, the Not Ready/Busy time concurrent with call handling should not be included in the specified unprorated or prorated values. Likewise, if an ACD counts Ready time for one service queue while also counting Not Ready/Busy time for another service queue, that Not Ready/Busy time should not be included in the specified unprorated or prorated values. Only the total unprorated

values totalUnproratedReadyWaitingTime and totalUnproratedNotReadyBusyTime should be double counted, and only as time an agent is Ready/Waiting for some service queue, and time an agent is Not Ready/Busy for all service queues, respectively.

EXAMPLE

```
File name: 201301151330N0600_1_ServiceHistorical.SERVICE

SERVICE DATE: 2013-01-15 INTERVAL: 13:30 TZOFFSET: -0600 ACD: 1

acdServiceId,contactsOffered,contactsHandled,contacts
   Answered,contactsAbandoned,totalTalkSeconds,totalHold
   Seconds,totalAfterContactWorkSeconds,totalAnswerDelay
   Seconds,serviceLevelPercent

5236,7,8,9,0,1314.324,0.000,238.228,80929.903,0.000

5240,4,4,4,0,365.039,0.000,104.644,35600.210,0.000
```

AgentState.EVENT File

The AgentState.EVENT file contains agent state data used to calculate adherence and conformance. The file must contain all agent state data for the entire day indicated in file name and file header. The day is interpreted as midnight to midnight in the tenant's time zone. The file replaces any existing agent state data for that day.

File Name Format

`<date>_<ACD#>_AgentState.EVENT`

Element	Description
<date>	The date the file was generated for, in YYYYMMDD format.
<ACD#>	The ACD number (see ACD Numbering).

File Header

`EVENT DATE: <date> ACD: <ACD#>`

Element	Description
<date>	The date the file was generated for, in YYYY-MM-DD format.
<ACD#>	The number of the ACD that is the source of the data.

The file header must be the first non-blank line in the file.

NOTE

The EVENT DATE values must represent the server date that matches both that in the file name and the date of all agent events in the file. For example, if a file is named:

20160123_1_AgentState.EVENT

then the file must contain only events for January 23, 2016 and no other date in the tenant time zone. This does not mean that the individual events must be expressed in the tenant time zone, only that after the events are converted to the tenant time zone by the Data Server, the date of the converted event timestamp must match the date in the file name and the file header.

Column Header

`<columnname1>,<columnname2>,<columnname3>. . .,<columnnameN>`

Where <columnname1> through <columnnameN> are the names of columns represented in the file.

Column names are not restricted to required and optional columns. You can add additional unrecognized columns and associated data (for example, agent names or service names) to make the GIS files more human-readable. The GIS API ignores these columns and their associated data.

Column order is not specified. The only requirement for column order is that the column names are in the same order as the data in each line.

Column names cannot contain commas, are case sensitive, and cannot start or end with a space (such spaces are automatically trimmed when the file is parsed). Column names must be unique. Duplicate column names (after space trimming) result in an error.

The column header must be the second non-blank line in the file.

Data Lines

`<columnvalue1>,<columnvalue2>,<columnvalue3>. . .,<columnvalueN>`

Where <columnvalue1> through <columnvalueN> are the values of <columnname1> through <columnnameN> for one row of data.

Each line of data corresponds to one data item within the file, with one value for each column, in the same order as the column header. The values are separated by commas. No value can contain a comma. Data values are trimmed of leading and trailing white space when parsed.

The third and subsequent non-blank lines in the file must be data lines. It is possible for a file to contain no data lines. This means there were no data items for that period.

Required Columns

The following table describes the columns that are required to be in the AgentState.EVENT file.

Column	Data Type	Description
acdAgentId	string	The ID of the agent in the ACD.
eventDateTime	timestamp	A timestamp representing the point in time at which the agent state event took place, consisting of the following data types: <date> <time> <tzoffset> If the tzoffset is omitted, the date and time are assumed to be in GMT time.
agentState	string, long, or null	A code representing the agent state. See Agent State Values for more information.

Optional Columns

The following table describes the columns that are optional in the AgentState.EVENT file. If not included, the default value is used for the metric.

Column	Data Type	Description
reasonCode	string, long, or null	The reason code associated with the agent state change. Default value = NULL.

Agent State Values

The agentState field accepts only certain values. If an event does not match one of these agent states, no line for that event should be written to the file. Each state has an associated integer or string code that can be used as a value. The following table describes the agent states and their associated codes.

Agent State	Code	Code	Description
Logging Out/ Out of Service	1	LO	The agent logs out of the ACD
Ready/Waiting	2	RE	The agent is ready to handle a contact

Agent State	Code	Code	Description
Talking/In Contact	3	TK	The agent is talking to a contact
Work After Contact	4	WK	The agent is performing after-contact work
On Hold	5	OH	The agent is on hold
Not Ready/Busy	6	NR	The agent is not ready to receive contacts

Only one event should be written per agent per timestamp. If multiple events are written, it is likely (but not guaranteed) that only the last event in the file for the agent for the same timestamp will be captured.

For ACDs that report an agent as Ready/Waiting or Not Ready/Busy for specific service queues, and use one of these values in one service queue while the agent is in another state on another service queue, the events must be converted to be non-service queue specific. For example, if an ACD has an agent as Talking in one service queue and Not Ready/Busy in other service queues, a single TK or 3 agentState should be written.

In general, when an agent is in multiple states for different service queues, and only one event per agent per timestamp is written, priorities are established as follows:

- Every other event takes priority over Logged Out
- Every event except Logged Out takes priority over Not Ready/Busy
- Every event except Logged Out and Not Ready/Busy takes priority over Ready/Waiting

EXAMPLE

```
File name: 20160115_1_AgentState.EVENT
EVENT DATE: 2016-01-15 ACD: 1
acdAgentId,eventDateTime,agentState,reasonCode
[lines omitted for brevity]
5009,2016-01-15 09:00:08.977,6,3
5073,2016-01-15 09:00:08.977,3,NULL
5073,2016-01-15 09:00:25.983,4,NULL
5073,2016-01-15 09:01:13.367,2,NULL
5073,2016-01-15 09:01:14.367,3,NULL
```

```
[lines omitted for brevity]
5009,2016-01-15 16:51:24.247,3,NULL
5009,2016-01-15 16:55:43.720,5,NULL
5009,2016-01-15 16:56:04.553,3,NULL
5009,2016-01-15 17:01:51.483,4,NULL
5009,2016-01-15 17:02:02.313,1,NULL
5073,2016-01-15 17:04:55.210,4,NULL
5073,2016-01-15 17:05:20.987,1,NULL
```

Integrating Your HRMS with Webex WFO

If HRMS (Human Resources Management System) integration is enabled (Application Management > WFM Configuration > HRMS Configuration), Webex WFO will import vacation data from your HRMS and export data on vacation hours used to a file that your HRMS can use.

Vacation data is exported once a day from Webex WFO to your HRMS at the time configured on the HRMS Configuration page. Vacation data is imported from your HRMS to Webex WFO throughout the day, whenever a GIS file is made available to Webex WFO.

IMPORTANT

Webex WFO uses either an agent's user name or an agent's employee ID as configured on the Users page to identify vacation hours data. Whichever identifier you use, it must be identical in Webex WFO and the HRMS for the data to be written to the correct record in each.

By default, the Data Server monitors the following folder for the text file sent from the HRMS to Webex WFO:

```
C:\Program Files\Common Files\Webex WFO\Data
Server\gis\vacationreports\<tenant>
```

NOTE

If the location of GIS files is changed from the default when configuring the Data Server, HRMS files will be saved to that path:

```
. .\gis\vacationreports\<tenant>
```

See the topic "Data Server Configuration" in the *Webex WFO User Guide* for more information.

As soon as the file appears, data from the file is imported into Webex WFO using the GIS API. The imported data overwrites any existing vacation data in the Webex WFO database.

BEST PRACTICE Data is exported from Webex WFO daily. It is up to the customer to determine the frequency at which data will be imported from the HRMS to Webex WFO. Best practice is to import the data at the interval at which earned vacation hours are calculated in the HRMS.

At the time of day configured on the HRMS Configuration page, Webex WFO outputs a report file that contains the number of hours used by agents for each day for the last seven days. This file also contains pending and approved hours as of the current date. Hours are considered used for schedules ending the previous day. Any schedules in progress are not counted as used.

When the text files are processed by Webex WFO, they are archived on the Data Server and in the Cloud for troubleshooting purposes.

Vacation export files are not automatically deleted. It is up to the customer to determine when and if the export files are to be deleted.

Import File Details

The following table displays the details of the import file sent from the HRMS to WFM.

IMPORTANT Every column header is required to be present in the import file, even if data is not required to be present for that column. If a column header is missing, then the import will fail.

Element	Description
File Location	C:\Program Files\Common Files\Webex WFO\Data Server\gis\vacationreports\<tenant>
File Name	VacationHours_From_HRMS
File Header	VACATION FROM HRMS DATE: <date>
Column Headers	login Data optional. This is the email address that users enter when they log in to Webex WFO. Although login and employeeId are individually optional, you are required to include data in at least one of these two columns. Otherwise, the data import file will not be processed.
	vacationTypeLabel Data required. The data value entered for vacationTypeLabel in all rows must match exactly the vacation type name as configured in Webex

Element	Description
	WFO(Application Management > Vacation Planning > Vacation Types). This value cannot contain commas.
	availableHours Data required. This is the number of hours of vacation available to the agent.
	totalEarnedHours Data optional. This is the total number of vacation hours earned by the agent for the year. If the optional totalEarnedHours field is not included in the file, Webex WFO defaults its values to zero and overwrites the existing values. This can result in a negative value when calculating the hours used for an agent. If the field is included in the file, the value entered cannot exceed 9999.
	employeeId Data optional. Although login and employeeId are individually optional, you are required to include data in at least one of these two columns. Otherwise, the data import file will not be processed.

The following is an example of an import file.

```
VACATION FROM HRMS DATE: 2018-04-19
   login,vacationTypeLabel,availableHours,totalEarnedHours,employeeId
   smithj@example.com,Floating Holiday,16,24,john.smith
   smithj@example.com,Vacation,120,160,john.smith
   jonest@generic.com,Vacation,70,80,teri.jones
   adamsb@generic.com,Vacation,120,120,betty.adams
```

NOTE Data elements are limited to 2 digits to the right of the decimal point. If there are more than 2 digits, the value will be rounded to 2 digits before import. Errors in records are logged in the log file.

Commas that are part of the HRMS import data must be escaped with a backslash (for example, “3\,000”) in order to be treated as literals and not file delimiters.

Export File Details

The following table displays the details of the export file sent from WFM to the HRMS.

Element	Description
File Location	C:\Program Files\Common Files\Webex WFO\Data Server\gis\vacationreports\<tenant name>
File Name	vacation_<date>_WFM
File Header	VACATION FROM WFM DATE: <date>
Column Headers	login date vacationTypeLabel usedHours requestedHours approvedHours employeeId acdId

- All agents that have a vacation plan via the import file (see Import File Details) are included in the export file.
- Only dates with usedHours in the past 7 days are included in the export for each agent.
- If a date has usedHours for an agent for at least one vacation type, then all vacation types are shown in the file, even if no hours were used on that date. In that case, the usedHours field displays a dash (—).
- The requestedHours and approvedHours fields appear only on the last date that is reported for each agent. These values are not tied to a specific date since they are totals for all future dates.
- If an agent does not have any used hours for any vacation type in the 7 day period, then a date row is inserted for yesterday for each vacation type and requestedHours and approvedHours are reported there.

The output export file contains the following for each vacation type:

- Used Hours: The hours used for the date on the row. This value is based on the duration of vacation type exceptions in the agent's schedule for the date. Vacation type exceptions are configured on the Webex WFO Vacation Types page (Application Management > Vacation

Planning > Vacation Types).

- Requested Hours: The total hours that are pending and not yet approved at the current time.
- Approved Hours: The total approved hours. For partial day requests, the duration of the request is used. For all-day requests, the duration is calculated as Minimum Hours Per Week ÷ 5 as defined in the agent's Full Time Equivalents Profile, configured on the Webex WFO Full Time Equivalents Profiles page (Application Management > Vacation Planning > Full Time Equivalents Profiles).

The following is an example of an export file.

```
VACATION FROM WFM DATE: 2018-04-20
   login,date,vacationTypeLabel,usedHours,requestedHours,approvedHours,
   employeeId,acdId
   smithj@example.com,2018-04-13,Floating Holiday,8.0,-,-,john.smith,2
   smithj@example.com,2018-04-13,Vacation,-,-,-,john.smith,2
   smithj@example.com,2018-04-15,Floating Holiday,-,-,-,john.smith,2
   smithj@example.com,2018-04-15,Vacation,8.0,16.0,8.0,john.smith,2
   jonest@example.com,2018-04-14,Vacation,4.0,-,-,teri.jones,1
   jonest@example.com,2018-04-15,Vacation,8.0,-,-,teri.jones,1
   adamsb@example.com,2018-04-19,Vacation,-,-,-,teri.jones,1
```

In this example, vacation hours are reported as follows.

Employee	Vacation Hour Usage
smithj	■ Used 8 hours of Floating Holiday on April 13. ■ Used 8 hours of Vacation on April 15. ■ Has 16 hours of requests for Vacation that are pending approval on or after April 20. ■ Has 8 hours of requests for Vacation that are approved on or after April 20. ■ Has no approved or pending requests for a Floating Holiday on or after April 20.
jonest	■ Used 4 hours of Vacation on April 14. ■ Used 8 hours of Vacation on April 15. ■ Has no approved or pending requests for a Floating Holiday on or

Employee	Vacation Hour Usage
	after April 20.
adamsb	■ Has no used hours in the 7-day period starting April 13. ■ Has no approved or pending requests for a Floating Holiday on or after April 20.

Real-Time Data Capture API

The real-time data capture API can be used to notify Webex WFO of real-time agent state information for any ACD. This real-time data is then used by Webex WFO to calculate agent adherence and conformance.

The real-time API is an HTTPS REST API.

API Definition

Applications using this API must do the following:

1. On application startup, establish an HTTPS session with the Application server.
2. For the life of the application:
    a. Monitor agent state changes using an ACD-specific integration method.
    b. Submit agent state changes to the Application server.
3. On application shutdown, destroy the HTTPS session.

Establish Session

Before providing agent state data to Webex WFO, you must first establish an HTTPS session (log in).

To log in, issue an HTTPS POST request to:

```
https://<Calabrio Cloud host>/api/rest/authorize
```

The Calabrio cloud host is set during installation. See the *Service Provider Installation Guide* for more information.

The body of the request must be in the following format:

```
{
```

```
        “userid”:“<username>”,
        “password”:“<password>”,
        “locale”:“en”
    }
```

NOTE The user whose credentials are in the <username> and <password> strings above must have Administer WFM permissions. If not, the API will not work.

NOTE Contact Support if the credentials used are valid for multiple tenants.

A response of **200 OK** indicates success.

Submit Agent State Information

You can submit each agent state change as a separate API request, or bundle several agent state changes into a single request. In general, sending multiple agent state changes in a single request results in higher performance.

To submit agent state information, issue an HTTPS POST request to:

```
https://<Webex WFO Cloud host>/api/rest/scheduling/
   gis/realTimeAgentState?acdServerId=<value>
```

The body of the request must be in the following format:

```
[
    {
        “acdAgentId”:string,
        “gisStateIdentifier”:string,
        “timestamp”:number,
        “reasonCode”:string
    }, ...
]
```

The fields in this request are defined in the following table.

Field	Descriptions
acdAgentId	The ACD identifier for the agent. This is the agent’s ACD ID in Webex

Field	Descriptions
	WFO (Application Management > Users > Agents). ■ Ensure that the format used in the ACD agent IDs that are synced from the ACD to Webex WFO is used whenever sending historical and real-time data to Webex WFO. If there is a case mismatch (for example, the agent ID in the ACD is **JSMITH** but in the data import file it is **Jsmith**), the data import will fail.
acdServerId	Identifier of the ACD. This is the number of the ACD shown on the ACD Configuration page.
gisStateIdentifier	The identifier for the agent state. ■ 1—Logout. The agent has logged out of the ACD. In many ACDs, a reason code might accompany this state to indicate the reason for the state change. ■ 2—Not Ready. The agent is not accepting contacts from the ACD. In many ACDs, a reason code might accompany this state to indicate the reason for the state change. ■ 3—Ready. The agent is ready to accept contacts from the ACD. ■ 4—Talking. The agent is on a contact. ■ 5—Work Not Ready. The agent is engaged in after contact work and will enter a Not Ready state when finished. ■ 6—Work Ready. The agent is engaged in after contact work and will enter a Ready state when finished. ■ 10—Hold. The agent has placed the contact on hold. **NOTE** If the ACD has an after contact work state, but makes no distinction between Work Not Ready and Work Ready, then either state can be used.
timestamp	The time that the agent state change took place, expressed in milliseconds since epoch. Timestamps must be sent in UTC expressed in epoch time in milliseconds.

Field	Descriptions
reasonCode	An optional reason code. This can be null or blank. Valid values are any alphabetical characters and symbols except the comma, and numbers 1–65535..

EXAMPLE

The following is an example of an agent state change sent as a single API request.

- Agent 5009 goes into the hold state at Wed, 27 Jun 2012 18:03:27 GMT.

```
[
    {
        “acdAgentId”:“5009”,
        “gisStateIdentifier”:“10”,
        “timestamp”:1340820207000,
        “reasonCode”:null
    }
]
```

The following is an example of multiple agent state changes bundled together into a single request.

- Agent 5009 goes Not Ready with reason code 9 at Wed, 27 Jun 2012 18:03:39 GMT.
- Agent 5009 goes Ready at Wed, 27 Jun 2012 18:03:40 GMT.
- Agent 5002 goes Not Ready with reason code 2 at Wed, 27 Jun 2012 18:03:39 GMT

```
[
    {
        “acdAgentId”:“5009”,
        “gisStateIdentifier”:“2”,
        “timestamp”:1340820219000,
        “reasonCode”:9
    },
    {
```

```
        "acdAgentId":"5009",
        "gisStateIdentifier":"3",
        "timestamp":1340820220000,
        "reasonCode":null
    },
    {
        "acdAgentId":"5002",
        "gisStateIdentifier":"2",
        "timestamp":1340820219000,
        "reasonCode":2
    }
```

Destroy Session

On application shutdown, you should destroy the HTTPS session (log out).

To log out, issue an HTTPS DELETE request to:

```
https://<Webex WFO Cloud host>/
   api/rest/authorize
```

A response of **200 OK** indicates success.

Bulk Contact Import APIs

Third parties use the Bulk Contact Import APIs to merge and insert metadata and recordings in a multi-part HTTPS request.

There are two Bulk Import APIs:

- Bulk Contact Import API—Used by third parties; allows insertion of both metadata and recordings in a multi-part HTTPS request

  NOTE You can also use the Bulk Contact Import API to import contacts in bulk through the Data Server. See Using the Data Server for Bulk Contact Import.

- Real-time Contact API—Used by Webex WFO Smart Desktop recording client; inserts metadata first and then uploads recordings separately later, based on the response from the metadata insertion. This allows the client to delay uploading recordings and allows contacts to be inserted

while the call is still in progress. However, this forces the client to track IDs to upload the recordings for the correct contact.

Protocol and URI

Bulk Contact Import API

URI	`/api/upload/contacts`
Method	POST
Permissions	Administer Tenant
Content Type	multipart/form-data

Real-time Bulk Contact Import API (Metadata)

URI	`/api/rest/wfo/contact/import`
Method	POST
Permissions	Record Voice/Record Screen
Content Type	application/JSON

Real-time Bulk Contact Import API (Recording Upload)

URI	`/api/rest/v2/recording/media/upload/audio` `/api/rest/v2/recording/media/upload/video`
Method	POST
Permissions	Record Voice/Record Screen
Content Type	Media Mime Type

Supported Formats

The following formats can be included in a multipart request.

Type	Description
CSV	A comma-separated file used to assign metadata.
JSON	The same metadata format as the Real-time API, but can be used for bulk import.
WAV	An audio recording format.
WEBM	A combined audio and video recording format.
WEBMA	An audio-only (WebM container) format.
WEBMV	A video-only (WebM container) format.
SPX	An audio format.
WMV	A combined audio and video recording format (or video only if paired with audio in the same contact).

Request and Response Fields

The CSV and JSON files include fields defined in the following table. Not all fields are used in both types of files. The file the field applies to is indicated in the description.

Name	Req?	Description
AgentId	Y	Used in CSV and JSON. The Agent ID in one of three formats. Processing figures out which format is used based on parsing the contents. ■ Person ID. A unique identifier from WfoPerson.id. This number is also used in the User Export spreadsheet (Application Management > Global > Users > Import and Export > Export > User ID column). It is not the same as the ID in the Webex WFO user profile. ■ AD Login. A domain\username (requires "\"). ■ Email address. An email address (requires "@"). When using a CSV to upload contacts, the agent ID is

Name	Req?	Description
		required. If you are using JSON to make the request, the agent ID is optional. In the latter case, the agent ID is set to the ID of the authenticated user initiating the upload. Max characters = 254 Default = none
AssocCallId	N	Used in CSV and JSON. An ID that ties contacts together. For example, a transferred call from one agent to another each have the same ID. Max characters = 52 Default = NULL
Audio.Location	N	Used in CSV and JSON. In the Audio sub-object. The key (file) name of the recording in the multipart request. This can be any supported recording format (audio/screen/combined). Only a single audio file per contact is allowed. The key name must have a valid extension that matches the media type of the recording. The extension identifies the file as an audio or screen recording, or both. Max characters = 128 Default = None
Audio.StartTimeMs	N	Used in CSV and JSON. In the Audio sub-object. The start time in milliseconds GMT since 1970-01-01 (UNIX time) of the audio recording. This is used to determine the audio offset from when the contact starts. Max characters = long Default = ContactStartTimeMs
CalledAddress	N	Used in CSV.

Name	Req?	Description
		The called phone number. Max characters = 64 Default = Empty string
Called	N	Used in JSON. The called phone number. Max characters = 64 Default = NULL
CallId	N	Used in CSV and JSON. An ID that identifies a contact. Max characters = 128 Default = NULL
Calling	N	Used in JSON. The calling phone number. Max characters = 64 Default = NULL
CallingAddress	N	Used in CSV. The calling phone number. Max characters = 64 Default = Empty string
ClientTimeZone	N	Used in CSV and JSON. The time zone in UTC format. Windows Time is also supported. The Desktop Recording client sends Windows Time, which is mapped to Olson time. Max characters = 255 Default = Customer's time zone as defined in Webex WFO

Name	Req?	Description
		EXAMPLE –06:00
ContactStartTimeMs	N	Used in CSV and JSON. The start time in milliseconds GMT since 1970-01-01 (UNIX time). A value in this field is required, so if the value is missing, the API uses the current upload time. Note that this likely results in a poor user experience, with many contacts that have the same timestamp. **IMPORTANT** If you are importing contacts with Excel, you must format the Start Time column to display milliseconds (consult the Excel user documentation for more information). Otherwise, Excel truncates milliseconds, resulting in a false time and preventing recordings from importing correctly. Max characters = long Default = current upload time **EXAMPLE** 1447100000000 - 11/09/2015 20:13:20 GMT
Direction	N	Used in CSV and JSON. The direction of the call, inbound or outbound. 0 = outbound 1 = inbound Max characters = 1 Default = NULL
Line	N	Used in CSV and JSON. The agent's line/extension. Max characters = 64 Default = NULL

Name	Req?	Description
metadata.<custom metadata field name>	N	Used in CSV. The custom metadata fields to populate. The field will be created if it does not exist. Any column beginning with "metadata" will be treated as a custom metadata field. **EXAMPLE** To set "accountNumber", create a column named "metadata.accountNumber". Max characters field name = 39 Max characters of custom metadata value = 2056
CustomMetadata	N	Used in JSON. The custom metadata fields to populate. The field will be created if it does not exist. The object contains data in the form of name/value pairs. **EXAMPLE** "accountNumber":"123456" Max characters field name = 39 Max characters of custom metadata value = 2056
Recording1	Y	Used in CSV. The key (file) name of the recording in the multipart request. This can be any supported recording format (audio/screen/combined). Only a single audio file per contact is allowed. The key name must have a valid extension that matches the media type of the recording. The extension identifies the file as an audio or screen recording, or both. Max characters = 128 Default = None
Recording2	N	Used in CSV. The key (file) name of the recording in the multipart request. This can be any supported recording format

Name	Req?	Description
		(audio/screen/combined). Only a single audio file per contact is allowed. The key name must have a valid extension that matches the media type of the recording. The extension identifies the file as an audio or screen recording, or both. Max characters = 128 Default = None
Recording3	N	Used in CSV. The key (file) name of the recording in the multipart request. This can be any supported recording format (audio/screen/combined). Only a single audio file per contact is allowed. The key name must have a valid extension that matches the media type of the recording. The extension identifies the file as an audio or screen recording, or both. **NOTE** There must be an audio file or the import fails. Max characters = 128 Default = None
Recording2Offset	N	Used in CSV. The offset of Recording2 from Recording1. **EXAMPLE** An audio file (WAV) that starts 5 seconds after the screen file (WEBM) has an offset of 5000 (5000 = 5 seconds). Max characters = Long Default = 0
Screen.Location	N	Used in CSV and JSON. In the Screen sub-object. The key (file) name of the recording in the multipart request. This can be any supported recording format (audio/screen/combined). Only a single audio file per

Name	Req?	Description
		contact is allowed. The key name must have a valid extension that matches the media type of the recording. The extension identifies the file as an audio or screen recording, or both. Max characters = 128 Default = None
Screen.StartTimeMs	N	Used in CSV and JSON. In the Screen sub-object. The start time in milliseconds GMT since 1970-01-01 (UNIX time) of the screen recording. This is used to figure out the screen offset from when the contact starts. Max characters = long Default = ContactStartTimeMs

CSV File Examples

CSV can be uploaded as part of a multipart upload request. Some rules regarding the format are as follows.

- The number of columns is variable. For example, if you always want to use the customer's time zone, you do not have to include the TimeZone column in the CSV.
- The columns included in the CSV can be in any order.
- The number of columns in each row must match the number of header columns.
- If a value has a comma, it must be surrounded by quotes.
- If a value is not known for a specific contact, but the header exists, use a empty string for that column.

> **NOTE** A successful response is formatted in JSON, because the response adds some status for each contact and recording.

Full Example

This example shows a file that uses every field possible for a CSV file.

```
AgentId,ContactStartTimeMs,TimeZone,AssocCallId,CallId,CalledAddress,Line,CallingAddress,Direction,Recording1,Recording2,Recording2Offset,metadata.accountNumber
calabrio/bunkowm,1447100000000,America/Chicago,103585664793210000,30611848,1801,1800,1800,1,call1.webmv,call1.wav,5000,1234567890
mark.bunkowske@calabrio.com,1447110000000,America/Chicago,103585664793220000,30611848,1801,1800,1800,1,call2.wav,,,987654321
```

Short Example

This example shows only the fields required for a CSV file.

```
AgentId,ContactStartTimeMs,Recording1
2,1447100000000,call1.wav
2,1447110000000,call2.wav
```

JSON File Examples

Contact information can be imported into Webex WFO in JSON format as an alternative to CSV format.

Full Example

The following is an example of a formatted JSON file.

```
{
        “AgentId”:“john.smith@acme.com”,
        “AssocCallId”:“103585664793254280”,
        “CallId”:“30611848”,
        “CalledAddress”:“1801”,
        “CallingAddress”:“1800”,
        “ClientTimeZone”:“Central Standard Time”,
        “ContactStartTimeMs”:1447075073000,
        “Direction”:1,
        “Audio”:[
              {
                    “Location”:“25.wav”,
                    “StartTimeMs”:1447075080000
              }
],
```

```
        "Screen":[
                "Location":"25.webm",
                "StartTimeMs":1447075075000
        ],
        "CustomMetadata":{
                "accountNumber":"123456",
                "department":"sales"
        }
}
```

Short Example

This example shows only the fields required for a JSON file.

```
{
        "AgentId":"acme\smithj",
        "ContactStartTimeMs":1447075073000,
        "Audio":[
              {
                    "Location":"25.wav"
              }
        ]
}
```

ZIP Format

The ZIP format is handled differently than CSV or JSON, in that it is a collection of files that are processed as if they were individual files within the multipart request.

- The name of the file is the key that needs to be referenced in the CSV/JSON.
- Any folder structure in the ZIP file is flattened and ignored.

For example, a multipart request looks like the following.

```
batch.zip
        batch.csv (contains 2 rows, for call1 and call2)
        call1.wav
        call2.wav
```

This multipart request is processed as if the files were all in the ZIP or all individually in the multipart request.

Notes

- The order of files does not matter.
- An upload for a contact that contains a recording file name but does not include that recording will fail to be inserted.
- An upload that contains a recording the is not referenced in a CSV or JSON will ignore that recording.

IMPORTANT You must have Tenant Administrator access, access to the data server to use for the Bulk Contact Import, and the Bulk Import permission checked for your role.

Using the Data Server for Bulk Contact Import

You can use the Bulk Contact Import API to upload contacts in bulk through the Data Server.

Using the Bulk Contact Import API requires the following:

- Webex WFO Administrator role with the Bulk Import permission enabled.
- Read/write access to the Data Server.

To upload contacts in bulk through the Data Server:

PREREQUISITE For a bulk import to successfully upload files using a data server, the data server must be configured within Webex WFO. This can be a Webex WFO Data Server that is already being used for any other purpose or a new Webex WFO Data Server. If you are configuring a new Webex WFO Data Server for bulk import, see the topic, “Data Server Configuration” in the *Calabrio ONE User Guide*.

1. To use a data server that is already configured in Webex WFO navigate to the Data Server Configuration page (Application Management > System Configuration > Data Server Configuration) and select that data server.
2. Ensure the following features are enabled:
    - In the Regional Data Server ACD Sync Settings section, select Enable Sync and assign the Generic (Default) ACD.

- In the Regional Data Server ACD Sync Settings section, select Enable Capture and assign the Generic (Default) ACD.
- In the Regional Data Server ACD Sync Settings section, select the Enable Media Import check box and assign the Generic (Default) ACD.

3. Click **Save**.
4. Create a CSV (not JSON) file that contains all required fields, plus any optional ones that you want to add.
5. Prefix the file with the word CONTACT. The word is case-sensitive, and you must type it in upper-case.

   EXAMPLE
   Your CSV file is named ExampleContacts.csv. You must rename it with the CONTACT prefix as follows:

   CONTACT.ExampleContacts.csv

6. Place the CSV file and all associated media files in the GIS <tenant> folder on the Data Server. This folder is in the location defined by the Regional Data Server GIS File Location field on the Data Server Configuration page.

   EXAMPLE C:\Program Files\Common Files\Webex WFO\Data Server\gis\<tenant>

Importing Post-Call Survey IVR Data

Data that is collected from post-call surveys via an IVR can be imported into Webex WFO using a generic IVR integration that uses CSV files saved to a specific folder on the Data Server. See “Add Post-Call Surveys to Contacts” in the *Webex WFO User Guide* for more information about configuration.

NOTE This folder location is configured in Webex WFO on the Data Server Configuration page (Application Management > Global > System Configuration > Data Server Configuration) in the Regional Data Server GIS File Location field.

The File Observer service triggers the import of these data files into the database and attaches the data to contact recordings using the Contact ID, Associated Call ID, or the ICM Call ID.

Two CSV files are required:

- The **Form** CSV file contains the survey questions and must be processed first.
- The actual survey results are imported through the **Results** CSV file.

Form CSV File

The Form file name must follow the format **Form_<Form ID>.csv**, where <Form ID> is a number.

The Form CSV file is formatted to contain the following information:

```
<form name>,<form status>,<form date>,<total score>
1,DIGITS,"Contact Identifier",0
<question number>,<question type>,<question text>,<question response and weight>
```

The first row in the Form CSV file contains the following information:

Field	Description
form name	The name of the survey form.
form status	The form's status can be **editable** or **active**. Editable forms can be modified with another import. With editable forms existing result data is deleted before updating the form details. Active forms cannot be changed.
form date	The form's date, in yyyy-mm-dd format.
total score	The total score possible in the survey.

The second row in the Form CSV file contains the following information which indicates the question number 1 that is used in the results to provide the contact identifier:

```
1,DIGITS,"Contact Identifier",0
```

The third and all subsequent rows in the file contain the survey questions:

Field	Description
question number	The number assigned to the survey question. **IMPORTANT** Question number cannot be 1.
question type	The type of question. **NOTE** Only **OPTION** type questions (an answer on a scale, for example from 1 to 5) can have results saved and a survey must have at least one **OPTION** type question to be associated

Field	Description
	with a contact.
question text	The survey question.
question responses and weights	The question response and weight is a comma-separated array in the format <option id> - <text for result> - <value/weight>. Where ■ <option id> is an ID for the option that is unique in the scope of the question. ■ <text for result> is the text used to identify this option in a survey result line. ■ <value/weight> is the value of the question option.

Results CSV File

Each Results file is an output snapshot from the generic IVR system. For example, the IVR can be configured to export one file every 30 minutes and include all surveys taken within the last 30-minute interval. If a form has two questions, then each survey response file will have three lines per survey:

- Line 1 identifies the contact ID to associate with the survey answers.
- Lines 2–3 contain the survey answers for each question.

The Results file name must follow the format **Results_<yyyyMMdd>_<HHMM>_<unique ID>.csv** where <unique ID> is a value that makes sure that the file name is unique. It can be based on timestamp, agent ID, or a generic sequential increment.

The following is the format of the Results CSV file:

```
<unique identifier>,<form ID>,<survey total earned score>,1,<contact ID or
   associated contact ID>,0
<unique identifier>,<form ID>,<survey total earned score>,<question number>,<answer
   text>,<answer score/weight>
```

- Where in the first row <1> and <0> are used for the contact identifier from the Form CSV file and do not change.

Each question result row in the file contains the following information:

Field	Description
unique identifier	An identifier matching the unique identifier in the Results file name.
form ID	The form ID used in the Form file name.
survey total earned score	The total score for the survey.
contact ID or associated contact ID	The identifier of the contact this survey applies to. **NOTE** The identifier that is used in the Results file is determined by the survey identifier selected on the Post Call Survey page in Webex WFO (Application Management > QM > QM Contact Flows > Post Call Survey).
question number	The question number matching the question number from the Form file.
answer text	The answer text matching one of the <text for result> option values for the question in the Form file.
answer score/weight	The score earned by the answer matching the <value/weight> for one of the option values for the question in the Form file.

A question is only included in the imported results if all of the following are true:

- The form ID matches the form ID of an imported form.
- The question number matches the question number on that imported form.
- The answer text and the answer score/weight match the text for result and value/weight for an answer option on that question.

Example

The following are examples of Form and Results CSV files for a scenario where a post-call survey consists of five questions. A customer answers the survey after a contact identified with the contact ID **987654321**.

The customer enters the following answers to the survey:

Question 301	3
Question 302	2

Question 303	3
Question 304	4
Question 305	4

The Form and Results file for this survey are as follows.

Form File Name: Form_3.csv

```
Customer_Satisfaction_Survey,editable,2019-10-17,200
1,DIGITS,"Contact Identifier",0
301, OPTION,Were you happy with wait time,1 - strongly disagree - 00,2 - disagree - 10,3 - neither - 20,4 - agree - 30,5 - strongly agree - 40
302,OPTION,How was service,1 - strongly disagree - 00,2 - disagree - 20,3 - neither - 10,4 - agree - 30,5 - strongly agree - 40
303,OPTION,Did we resolve issue,1 - strongly disagree - 00,2 - disagree - 20,3 - neither - 10,4 - agree - 30,5 - strongly agree - 40
304,OPTION, Was agent knowledgeable - strongly disagree - 00,2 - disagree - 10,3 - neither - 20,4 - agree - 30,5 - strongly agree - 40
305,OPTION,How satisfied with general services,1 - strongly disagree - 00,2 - disagree - 10,3 - neither - 20,4 - agree - 30,5 - strongly agree - 40
```

Results File Name: Results_20191017_1309_1571310547.csv

```
1571310547,3,110,1,987654321,0
1571310547,3,110,301,neither,20
1571310547,3,110,302,disagree,20
1571310547,3,110,303,neither,10
1571310547,3,110,304,agree,30
1571310547,3,110,305,agree,30
```

File-based Sync Functionality

You can import and synchronize user, team, and service queue data using GIS functionality to add and update this data.

NOTE You cannot delete data using files. That function remains a manual process.

The files are placed in the location configured in Webex WFO Application Management on the Data Server Configuration page in the Regional Data Server GIS File Location section. The Data Server will import files from this location. Once the files are processed by the sync process, the files are archived both on the Data Server and in the cloud.

Good files are archived under the ~/gis/archives<date> folder on the Data Server. They are kept for 1 week. Bad files are not uploaded. They are moved to the ~/gis/penaltyBox/<date> folder on the Data Server, and no further attempts are made to upload them.

Users File

User information is contained in a file called Users.csv. When the file is imported:

- Users are created if they do not exist in the Data Server, and roles are assigned.
- If users already exist, the user names and teams are updated, roles are assigned if not already assigned.

  NOTE Do not reactivate users that have been deactivated. This is to allow you to manually deactivate a team without deactivating users in the ACD.

- Users are assigned to the default team if the team colum is missing or if no team is specified for the user.
- Users must have at least one valid role assigned to them. If a role specified in the file does not exist in Webex WFO, then it is skipped without error.

The details of this file are as follows. Fields in the CSV file can be in any order from left to right.

Field	Required?	Type	Description
acdId	Yes	String	The user's identifier in the ACD.
acdServerId	Yes	Number	Identifier of the ACD. This is the number of the ACD shown on the ACD Configuration page.
displayTimeZone	No	String	The time zone the user's schedules are to be displayed in, in Olson Timezone format. If none is provided, the tenant's timezone is used.
employeeId	No	String	The user's employee ID.
enableScheduling	Yes	Boolean	True or False. Enables the user to be scheduled.
firstName	Yes	String	The user's first name.

Field	Required?	Type	Description
lastName	Yes	String	The user's last name.
roles	Yes	String	The roles assigned to the user. Multiple roles are delimited by semicolons. The roles listed must exist.
teamAcdId	No	String	The ACD ID of the team associated with the user.
username	No	String	The user's Webex WFO user name.
windowsLogin	No	String	The user's Windows login, if Active Directory is used.

File Example

```
acdServerId,acdId,employeeId,firstName,lastName,roles,teamAcdId,username,windowsLogi
    n,displayTimeZone,enableScheduling
    1,1001,123,Larry,Jones,Agent;Supervisor,9001,larry.jones@t.com,larry.jones,Americ
    a/Chicago,true
    1,1002,456,Bob,Henderson,Agent,9001,bob.henderson@t.com,bob.henderson,America/Chi
    cago,true
    1,1003,789,Sara,Williams,Agent,9002,sara.williams@t.com,sara.williams,America/Chi
    cago,true
```

Teams File

Team information is contained in a file called Teams.csv. When the file is imported:

- Teams are created if they do not exist in the Data Server.
- If teams already exist, the team names are updated.
- Teams that were synchronized before but do not exist in the current upload file are deactivated.
- Do not reactivate a team that has been deactivated. This is to allow you to manually deactivate the team without deactivating it in the ACD.

The details of this file are as follows. Fields in the CSV file can be in any order from left to right.

Field	Required?	Type	Description
acdId	Yes	String	The team's identifier in the ACD.
acdServerId	Yes	Number	Identifier of the ACD. This is the number of the ACD shown on the ACD Configuration page.
name	Yes	String	The team's name.

File Example

```
acdServerId,acdId,name
   1,9001,Sales
   1,9002,Support
   1,9003,Customer Relations
```

Service Queues File

Service queue information is contained in a file called ServiceQueues.csv. When the file is imported:

- Service queues are created if they do not exist in the Data Server.
- Skill mappings are created for the service queues.
- If service queues already exist, the service queue names are updated.

The details of this file are as follows. Fields in the CSV file can be in any order from left to right.

Field	Required?	Type	Description
acdId	Yes	String	The service queue's identifier in the ACD.
acdServerId	Yes	Number	Identifier of the ACD. This is the number of the ACD shown on the ACD Configuration page.
name	Yes	String	The team's name.

File Example

```
acdServerId,acdId,name
   1,8001,skill1
   1,8002,skill2
```

```
1,8003,skill3
```

WFM Historical Import Tool (WHIT)

The WFM Historical Import Tool (WHIT) is not included with your Webex WFO installation. You must request the components from Support and install them according to the instructions in this section.

NOTE File paths shown here are the default file paths.

Software Requirements

The software requirements for running the WHIT tool are as follows:

- The Data Server must be installed and running.
- Java 8 or later must be installed on the Data Server.

WHIT Components

WHIT consists of the following components:

- WFM Historical Import Template spreadsheet
- WHIT JAR file (wfm-historical-import-tool.jar)
- WHIT BAT file (WHIT.bat)

WFM Historical Import Template

The WFM Historical Import Template is an Excel spreadsheet named WFM_Historical_Import_ Template.xls. Your historical data is entered in this spreadsheet and then saved in CSV (comma separated value) format.

WHIT JAR and BAT Files

The WHIT BAT file starts the Historical Import Tool, which then is used to convert your historical data from the CSV file you create from the Historical Import Template to a set of files in the format expected by the Generic Interface Services (GIS) API, a feature that comes with WFM. The GIS API, in turn, imports the files into the Webex WFO database.

Importing Data with WHIT

The process of importing data consists of the following tasks:

1. Install WHIT.
2. Insert your data into the WHIT spreadsheet and save it in CSV format.
3. Run WHIT.bat.

Installing WHIT

To install WHIT on your Data Server:

1. Log in to the Data Server.
2. Copy `WHIT.bat`, `wfm-historical-import-tool.jar`, and `WFM_Historical_Import_Template.xls` to any folder on the Data Server.

Preparing the WHIT Spreadsheet

The fields that you complete in the WFM Historical Import Template spreadsheet depend on the type of service queue for which you are importing data.

There are two service queue types:

- **Interactive**: Interactive service queue types consist of contacts in which agents and customers have real-time communication, such as voice (call) or chat.
- **Non-Interactive**: Non-interactive service queue types consist of contacts in which agents and customers do not have real-time communication, such as email, fax, and social media, as well as contact activities like stuffing envelopes.

The field names can be interpreted differently for each service queue type. For example, ReceivedCalls can be thought of as the number of chat contacts received or the number of email contacts received. Complete the spreadsheet in a manner that works best for your contact center.

To prepare the WHIT spreadsheet:

1. Open the `WFM_Historical_Import_Template.xls` spreadsheet.
2. Insert your historical data into the spreadsheet. A description of what goes in each field is included in the spreadsheet.

   To import historical data for interactive service queues, complete the required fields in red. You can also complete optional fields in green if desired.

To import historical data for non-interactive service queues, complete the required fields in red. It is recommended that you also complete the **ServiceLevel**, **QtyOfAgents**, and **OccupancyRatio** fields.

> **NOTE** Be aware that sometimes spreadsheets convert date formats to something other than what you enter. Dates must be in the format YYYY-MM-DD to be imported correctly.

Set the values in required fields to **0** (zero) for data you do not want to specify.

> **NOTE** If those fields are optional, you can remove those columns.

3. When you have added all your historical data, save the spreadsheet in comma separated value (CSV) format.
4. Copy the CSV file to the same location on the Data Server where you copied the WHIT JAR and BAT files

Running WHIT

Perform this task on the Data Server.

To run WHIT:

1. Double-click `WHIT.bat` to start the import tool.
2. Follow the instructions in the tool:
    a. Click **Next** and complete the check list on the second screen. You cannot proceed until you have checked off each item.
    b. Click **Next** and complete the fields on the third screen.
3. Click **Import**. WHIT generates GIS files from your data and places them in the following location:

    ```
    C:\Program Files\Common Files\Webex WFO\Data Server\gis\<tenant name>
    ```

    > **NOTE** This is the default location.

4. The GIS API then processes the GIS files and imports the data into the Webex WFO database. After the GIS files are processed, the GIS API removes the files from the gis folder and archives them under the ~/gis/archives<date> folder on the Data Server, and WHIT displays the following message:

    ```
    All files are captured. Historical import is complete.
    ```

The imported data can be viewed in Webex WFO on the Application Management > View and Edit Historical Data page.