Mitel®

Powering connections

Mitel MiVoice Connect Contact Center Applications Interface User Guide

September 30, 2019

Trademarks

Patents

Mitel products are covered by patents as listed at http://www.shoretel.com/about/patents.html.

Version Information

Applications Interface Guide
Connect Contact Center
Date: January 4, 2019

Company Information

Mitel Networks Corporation
350 Legget Dr.
Kanata, Ontario K2K 2W7
Canada
+1.844.937.6483

Table of Contents

Before You Start

The following sections provide an overview of the book.

About This Book

ShoreTel is now part of Mitel. Together, we look forward to helping you power connections that are brilliantly simple.

This guide describes how to integrate Connect Contact Center with external applications, including Customer Relationship Management (CRM) products. It details the available interfaces and provides necessary reference information, including examples.

Note

The options and modules required for application integration are only available with Connect Contact Center.

This guide provides information for professional services engineers and application programmers who must plan the integration of Connect Contact Center to selected external applications before development begins.

Organization

The document is divided into the following chapters:

- Chapter 1, Integrating with a Customer Database
- Chapter 2, Using Dial Lists
- Appendix A, Supported SQL Syntax

Conventions

The following typographical marking conventions are used in this document.

Marking	Meaning
Bold	Names of interface objects, such as buttons and menus.
`Courier`	Code examples.
`Courier Italic`	Variables in code examples.
Blue	Cross references with hyperlinks. Click the blue text to go to the indicated section. All chapters have a list of section links on the first page. **Note:** Table of Contents entries are also links, but they are not shown in blue.

CHAPTER

1

Integrating with a Customer Database

Connect Contact Center can be closely integrated with databases containing customer information to improve customer service. For example, you can use your Customer Relationship Management (CRM) application's database to specify agent skills, call priority, and call routing, and to let callers bypass agents and interact directly with the CRM database in a self-service application.

Integrating Connect Contact Center with a customer information database requires the interaction of three key components:

- Call profiles
- Call control scripts
- CRM database

Refer to the following sections for more information about integrating with a customer database:

Note

Connect Contact Center does not support file-based databases, such as Microsoft Access *.mdb files, over the network. If the database is small, store it on the Connect Contact Center server. If the database is large, consider using a business-class, server-based database application, such as MySQL. Because of possible performance issues, it is not recommended to store tens of thousands of records in a file-based database and have concurrent access to it from various applications.

Using Call Profiles

A call profile consists of dynamic information that is attached to each call in the system. This information is used to route the call and record its history. The call profile is updated as the call moves through the system and by caller interaction or by a CRM database that uses call control scripts.

Call Profile Fields

Each field in the call profile is either mandatory or optional. Mandatory fields are predefined and exist in any Contact Center Solution installation. Mandatory fields cannot be removed or changed by the user. Optional fields that meet your business needs—also called user fields—can be defined by using Connect Contact Center Director > **System Parameters** > **Routing Preferences** > **Call Profiles**. Refer to the *Mitel Connect Contact Center Director Administration Guide* for specifics.

Connect Contact Center uses Structured Query Language (SQL) for database interactions. Use the Call Profile Name, surrounded with a pair of percent signs (%), in a SQL statement. For example

```
SELECT *
FROM table_1
WHERE phone_number = %ANI%
```

The following is a list of mandatory fields in the call profile. The field names in the call profile are case-sensitive.

Table 1: Call Profile Mandatory Fields

TAPI Call Properties	Call Profile Name	Description
_STCC_ANI	ANI	Automatic Number Identification (ANI) transmits the customer's telephone number and delivers it to your call center's telephone system. **Note**: The format of the ANI call profile field includes a plus sign (+) before the ANI number. This facilitates external incoming calls (i.e. from outside your site).
_STCC_Agent Queue	Agent Queue	Indicates that the call is being handled by a personal agent queue.
_STCC_Group	Group	The internal group id related to the call
_STCC_Trunk	Trunk	PBX and call-type depended
_STCC_Call ID	Call ID	The ID of the call
_STCC_Caller Name	Caller Name	Indicates the name of the caller in the PBX.
_STCC_DNIS	DNIS	The DNIS of the call

Table 1: Call Profile Mandatory Fields

TAPI Call Properties	Call Profile Name	Description
_STCC_DNIS Name	DNIS Name	Indicates the name of the DNIS in the PBX.
_STCC_Type	Call Type	Call type (Voice, Chat, Email, Callback, Abandoned, Web Callback, Dial List)
_STCC_Media	Media	Call media (WEB, VOICE, EMAIL)
_STCC_Priority	Priority	Priority of the call
_STCC_Service	Service	Service that handles the call
_STCC_Customer Number	Customer Number	Identifier number of the customer (if internal, the Customers Table is used)
_STCC_Customer Name	Customer Name	Customer name (if internal the Customers Table is used)
_STCC_ACD Enter Date	ACD Enter Date	Date the call entered Contact Center
_STCC_ACD Enter Time	ACD Enter Time	Time the call entered Contact Center
_STCC_Queue Position	Queue Position	Call position in the queue
_STCC_Average Queue Time	Average Queue Time	Expected average wait time in the queue.
_STCC_Call Back Time	Call Back Time	Callback time set by caller (if empty, Abandoned Call)
_STCC_Call Back Date	Call Back Date	Indicates the date when the call back was made.
_STCC_Call Back Destination	Call Back Destination	Callback destination
_STCC_Language	Language	The language associated with the call (0=English, 1=Hebrew, 2=Russsian, 3=Arabic, 4=Mexican Spanish)
_STCC_Agent Extension	Agent Extension	The agent's telephone extension number
_STCC_Agent Number	Agent Number	Number used to identify the agent
_STCC_Dial List ID	Dial List ID	The ID of the dial list
_STCC_Start Queue Time	Start Queue Time	The time when a call enters the queue
_STCC_Execute Req	Execute Req	For any request to be executed
_STCC_Trunk Number	Trunk Number	The number of the trunk
_STCC_Email To	Email To	The text in the To field of the email
_STCC_Email CC	Email CC	The text in the CC field of the email
_STCC_Email Subject	Email Subject	The text in the Subject field of the email
_STCC_Email From	Email From	The text in the From field of the email
_STCC_Email Sent Date	Email Sent Date	The text in the Date Sent field of the email

Table 1: Call Profile Mandatory Fields

TAPI Call Properties	Call Profile Name	Description
_STCC_Email Sent Time	`Email Sent Time`	The text in the Time Sent field of the email
_STCC_Email Enter OMS Date	`Email Enter OMS Date`	The text in the OMS Date field of the email
_STCC_Email Enter OMS Time	`Email Enter OMS Time`	The text in the OMS Time field of the email
_STCC_Primary Call Back Destination	`Primary Call Back Destination`	The primary destination for the callback
_STCC_Alternative Call Back Destination 1	`Alternative Call Back Destination 1`	The first alternative destination for the callback
_STCC_Alternative Call Back Destination 2	`Alternative Call Back Destination 2`	The second alternative destination for the callback.
_STCC_Last Time To Initiate Callback	`Last Time To Initiate Callback`	The time when the last callback can be made
_STCC_Alternative Call ID	`Alternative Call ID`	The PBX TAPI call ID
_STCC_Outbound Caller ID	`Outbound Caller ID`	The caller ID of the outbound call
_STCC_SW Call GUID	`SW Call GUID`	The PBX call GUID

Extracting and Storing Call Profile Information in TAPI

You can extract and store call profile information in TAPI by using the following process:

1. On the Connect Contact Center Server, in the directory in which Contact Center is installed, open the **Bin** folder.
2. In the Bin folder, use a text editor to create a file named **shoretelcfg.ini**.
3. In the shoretelcfg.ini file, add a section named **call_profile**.
4. In the call_profile section, specify a key named **user_fields** with call profile field names as values. Separate the values with a comma, as shown below:

```
[call_profile]
user_fields=AccNo, Balance, DueDate
```

Tip

The call profile field names are prefaced with _STCC_ when they are saved as TAPI call properties. For example, in the code shown above, the call properties would be _STCC_AccNo, _STCC_Balance, and _STCC_DueDate.

5. Restart the Connect Contact Center Server.

The specified call profile field values display in the TAPI call properties.

Creating Call Control Scripts

Call control scripts define the way a system processes automatic call distribution (ACD) calls. Each script contains a set of actions that are performed on a call. Call control scripts can be used to perform the following functions:

- Create announcements
- Collect caller information
- Make routing decisions
- Implement self-service
- Provide music-on-hold
- Read and write information to a database
- Update the call profile fields that control call handling

Connect Contact Center provides Graphical Call Control Scripts (GCSS) Administration, a graphical call control scripting tool available from Connect Contact Center Director. Refer to the *Connect Contact Center Administrator Guide* for information on using this tool to create call control scripts.

Complete the following steps to use call control scripts:

1. Plan the actions you want the call control script to perform.
2. Identify and connect to an external database and create open database connectivity (ODBC) queries.
3. Create any new call profile fields needed by the call control script.
4. If you are creating an announcement, record the announcement.
5. Create the call control scripts.
6. Specify the IVR application parameters.
7. Identify the call control script as an IRN or service destination.

Database Interactions

To integrate your call control scripts with customer information, you must first identify the database and the columns containing the necessary information. When integrating with an external database, Connect Contact Center acts as the client and uses an ODBC interface on the Connect Contact Center Server to connect to the database.

Connect Contact Center uses Structured Query Language (SQL) for database interactions. External databases must be in SQL format to be accessible by Connect Contact Center through ODBC.

Configuring and using database interactions with Connect Contact Center includes the following procedures. Refer to the associated sections for detailed information about each procedure.

- Plan Data to Capture — Plan the data items you need to gather, define the conditions, and decide where to put data results. If prompts are part of the solution you are building, ensure you place the appropriate files in the IVR folder on the Connect Contact Center server.
- Install ODBC Drivers — Connect Contact Center is a 32-bit application running on a 64-bit operating system, so you must install a 32-bit ODBC driver.
- Create a Data Source on the Connect Contact Center Server
- Set Up a Database Connection in Connect Contact Center Director
- Build Routing — Build out routing as needed. For example, you may need to build out the following items to route interactions appropriately:
  - IRNs and DNIS
  - Services
- Create Script — Create a GCCS script to interact with the data connection you have established.

  Ensure your script contains the following actions:

  - SQLConnect connects to a database to make database queries.
  - SQLExecute sends a query (written as a SQL statement) to the database to obtain specific information, such as the priority of a caller.
  - SQLDisconnect disconnects from the database.

  Refer to Appendix A, Supported SQL Syntax Supported SQL Syntax for information about the supported SQL syntax.

Plan Data to Capture

Before proceeding with the steps to establish a data connection between Connect Contact Center and a database, plan out the data you want to capture and how you want to use it. For example, if you want to capture CRM data to determine call flow, you might plan to capture mandatory call profile data such as the ANI and CUSTOMER_NUMBER as well as user-defined call profile information such as the customer account number and account balance.

Install ODBC Drivers

Using the web, locate, download, and install a 32-bit version of the driver or connector required for the type of database you are using. For example, if you are using a MySQL database, download and install the 32-bit MSI installer version of the MySQL ODBC connector.

Tip

If you get any errors while installing the database connector, refer to the documentation for the ODBC driver you are installing for troubleshooting information.

Create a Data Source on the Connect Contact Center Server

1. Navigate to the `SysWOW64` directory on your Windows server and run the `odbcad32.exe`.
2. In the source administrator screen, click the **System DSN** tab, and then click **Add**.
3. Select the Unicode Microsoft Access driver, and click **Finish**.
4. Enter configuration information such as the IP address of the database server and login credentials, if applicable. Configuration specifics will differ depending on the driver you selected in step 3.

Set Up a Database Connection in Connect Contact Center Director

1. Navigate to `http://<contact center server IP address>:3000/ccd` to open Connect Contact Center Director.
2. Click **System Parameters** > **External Interfaces** > **Database Connections**.
3. Click **New**.
4. Specify the following information:
   - **Name** — Describes the connection you are configuring. For example, if this connection is for gathering account information, you might specify Acct_Data here.
   - **ODBC DSN** — Name of the DSN connection you set up in step of the Create a Data Source on the Connect Contact Center Server section.
   - **Database Type** — Type of database you are connecting Connect Contact Center to.
   - **Timeout** — Length of time that can elapse with no activity between Connect Contact Center and the database before the connection is terminated.
   - **Username** and **Password** — Credentials for database access. This information is required only if you defined credentials for the DSN connection in step of the Create a Data Source on the Connect Contact Center Server section.
   - **Simultaneous Connections** — Number of simultaneous connections that can occur between Connect Contact Center Director and the database.
5. Click **Create**.

Build Routing

Building routing to use the data you have pulled from the database connection you defined may include any or all of the following items. Refer to the sections mentioned for information about each item. Refer to the *Connect Contact Center Administrator Guide* for information about routing and the related entities and applications.

- IRNs and DNIS — Intelligent Routing Numbers (IRNs) are dial numbers used as entry points into the Connect Contact Center routing system. Each number can be used to define rules for routing incoming calls to various destinations, such as services, call control scripts, and devices. The DNIS is the same as the IRN Dial Number. For the purpose of gathering statistics on an IRN, the IRN should also be defined as a DNIS.
- Services — Define how incoming calls to Connect Contact Center are processed.
- IVR Application — Used by the Mitel Connect system to perform interactive tasks with the caller.

Create Script

Using the Connect Contact Center graphical call control scripting (GCCS) application, you can route calls based on the data you have collected from the database connection you established. Refer to the *Connect Contact Center Administrator Guide* and to the GCCS help for information about creating call control scripts.

Example: Routing a Call Based on CRM Information

You can use the data in your customer database to route calls to agents who can better serve a particular customer. The following example routes an incoming call based on a customer type field in a CRM database.

In this scenario, a call arrives at a financial institution and is routed by customer type to either a group of agents handling stock trades, or to a group that specializes in bonds. A field for customer type has been added to the call profile.

The incoming call follows these basic stages:

1. The call arrives at the IRN and is sent to a call control script.
2. The call control script uses the Get Digits action to collect the Customer ID number and record it to the call profile.
3. The call control script assesses the database using the following interaction:
   a. Connects to the CRM database using the SQLConnect action.
   b. Uses the Query Database action to identify the customer's data in the database.
   c. Uses the SQLExecute action to retrieve the customer type from the CRM database, and writes it to the call profile.
4. The call control script uses the Decision action to check the Customer Type field and branch to the correct Change Profile action that sets the required service in the call profile.

Example: Providing Self-Service Options Using CRM Information

You can use the data in your customer database to provide customers with self-serve options. An IVR application can play back selected WAV files based on the customer status. An IVR application also can convert text digits, currency, or dates into speech. The example below provides an account balance to a self-service customer.

In this scenario, a customer calls a financial institution and subsequently hears options for checking an account balance. A field for customer balance has been added to the call profile.

The incoming call follows these basic stages:

1. The call arrives at the IRN and is sent to a call control script.
2. The call control script uses the Get Digits action to collect the Customer ID number and record it to the call profile.
3. The call control script accesses the database using the following interaction:
   a. Connects to the CRM database using the SQLConnect action.
   b. Uses the Query Database action to identify the customer's data in the database.
   c. Uses the SQLExecute action to retrieve the customer's account balance from the CRM database and write it to the call profile.
4. The call control script uses the Announce action to report the account balance to the caller.

CHAPTER

2

Using Dial Lists

Connect Contact Center can automatically make outbound calls for agents based on dial lists, which are lists of phone numbers to call. To use dial lists for outbound dialing campaigns, you must have the dial data stored in an ODBC/SQL database.

It is recommended that your dial list database be case-insensitive, since the format of a dial list query is automatically converted to lower-case characters when run.

Dial lists should be in the form of a table in a SQL database. For more information about dial lists, refer to the *Connect Contact Center Administration Guide*.

Accessing Dial List Data

To use dial lists for outbound dialing campaigns, you must have the dial data stored in an external ODBC/SQL database.

Complete the following steps to access dial list data from an external database:

1. Create the physical connection to the external database. Make sure you can ping between the computers and that they can communicate via the network.
2. Install the needed infrastructure software, such as OBDC drivers and client side processes, and create the OBDC entries on the Connect Contact Center Server.
3. Plan which data items you would like to gather, define the conditions, and decide where and how the data will be used in call control scripts.
4. Define the Data Source Name (DSN) of the database to which you want to connect.

5. Restart Connect Contact Center.
6. Create the SQLConnect and SQLExecute scripts.

- SQLConnect connects to a database for queries.
- SQLExecute sends a query, which is written as a SQL statement, to the database to obtain specific information, such as the priority of a caller.

Refer to Appendix A, Supported SQL Syntax for information on the supported SQL syntax.

Storing Dial List Results

After an agent dials an outbound call, the agent can identify the call status using Connect Contact Center. This includes a complete call wherein a person was reached, an incomplete call that should be tried again later, and a call placed using an incorrect telephone number. You must define a location in your database for storing these call results.

APPENDIX

A

Supported SQL Syntax

This section describes the SQL syntax statements supported by the Connect Contact Center Solution.

Refer to the following sections for more information about supported SQL syntax:

All Simple Statements

The following is an example of a simple SQL statement.

```
SELECT field1, field2, field3
FROM table1
WHERE field1 = value
UPDATE table1
SET field2 = value
WHERE field1 = value
INSERT into table1 values ( field1, field2, field3 )
DELETE
FROM table1
WHERE field1 = value
```

Statements with Computed Fields

The following is an example of a SQL statement with computed fields.

```
SELECT field1, fieldx*2
AS field2, field3
FROM table1
WHERE field1 = value
```

JOIN Statements

The following is an example of JOIN SQL statements.

```
SELECT field1, field2
FROM table1, table2 table1.id1 = table2.id2
AND field1 > 0
SELECT field1, field2
FROM table1 left
JOIN table2 on table1.id1 = table2.id2
WHERE field1 > 0
SELECT field1, field2
FROM table1 inner
JOIN table2 on table1.id1 = table2.id2
WHERE field1 > 0
```

Nested Statements

The following is an example of nested SQL statements.

```
SELECT field1
FROM table1
WHERE id1 in
    ( select id2
    FROM table2 )
UPDATE table1
SET field1 = value1
WHERE id1 in
    ( select id2
    FROM table2 )
DELETE
FROM table1
WHERE id1 in
    ( select id2
    FROM table2 )
```

Scripts

The following is an example of a SQL statement with scripts.

```
DECLARE @value1 int
DECLARE @value2 int
SET @value1 = 5
SELECT @value2 = field2
FROM table1
WHERE field1 = @value1
```

SQL Syntax Format Limitations

The following SQL syntax format restrictions apply to Connect Contact Center.

- Connect Contact Center supports only certain data types for write operations to an external database. For example, when the system writes the ANI to a database for routing purposes, the column must use `VARCHAR` — it cannot use `NVARCHAR`.
- In a field that accepts only `SELECT` statements, any statement that does not start with `SELECT` does not work.
- The outcome of complex scripts that include more than one basic statement usually depends on the ODBC driver of the database.

- For database integration to succeed, music on hold (MOH) must be configured. During the execution of an SQL operation, the system plays the default hold music until it receives a result from the database. If the MOH is not specified, the system disconnects the caller. Specify MOH in **System > Misc**.

Mitel recommends customers work with the Advanced Applications team or implementation services for scripting a database or dial list to ensure that the scripts meet the requirements of the customer.