Mitel MiContact Center Enterprise

CHAT CONFIGURATION – OPERATING INSTRUCTIONS

Release 9.3

NOTICE

The information contained in this document is believed to be accurate in all respects but is not warranted by Mitel Networks™ Corporation (MITEL®). The information is subject to change without notice and should not be construed in any way as a commitment by Mitel or any of its affiliates or subsidiaries. Mitel and its affiliates and subsidiaries assume no responsibility for any errors or omissions in this document. Revisions of this document or new editions of it may be issued to incorporate such changes.

TRADEMARKS

Chat Configuration – Operating Instructions
Release 9.3 – February 2018

INTRODUCTION

MiCC Enterprise may be integrated with chat functionality allowing customers to request chat sessions which will be routed to agents. This document will describe integrating the built in customer chat client into your external Web site. Knowledge of basic Internet Web design and HTML is assumed.

CUSTOMER CHAT CLIENT

Chat functionality may be integrated into your Web site using the stock chat client or you may design your own.

STOCK CHAT CLIENT

During installation of MiCC Enterprise a CustomerChat Web application is added to the default Web site of Internet Information Server. The CustomerChat Web application was built using ASP.NET v4.0 and uses SignalR technology for communication between the client and server.

A typical scenario for invoking a chat session would be having a button on a page that requests information from the customer such as their name, e-mail address and which department they wish to chat with. Then, the chat page would be opened in a popup window with the necessary arguments.

The URL for the stock chat client:

http://<WebServer[:port]>/customerchat/chat.aspx

The external Web server must have access to this URL. It is the responsibility of the Web Admin for opening external access to the page and handling all security configuration.

Several URL arguments are required:

ARGUMENT	REQUIRED	PURPOSE
sessionid	Yes	Unique ID for this chat session. This can be any unique character string such as a guid.
customername	No	Name of customer.
customerid	No	ID of customer. This may be used to tag the chat session during archive operations.
email	No	E-mail address of the customer.
tenantid	Yes	Tenant ID of the MiCC Enterprise service group where the chat session will be routed. This will be -1 for non-tenanted MiCC Enterprise systems.
servicegroupid	Yes	ID of the service group where the chat session will be routed. Either the servicegroupid or servicegroupname argument must be specified.

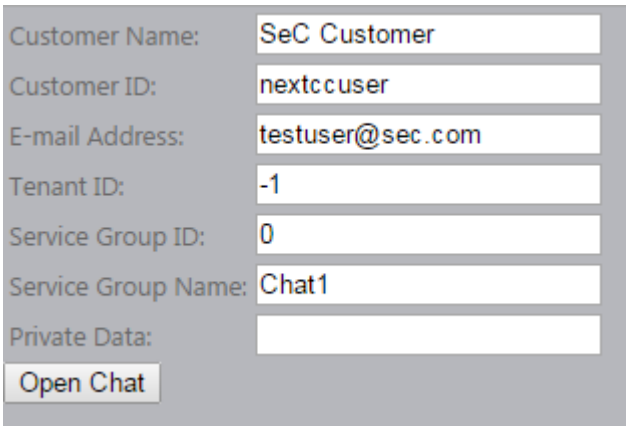
servicegroupname	Yes	Name of the service group where the chat session will be routed. Either the servicegroupid or servicegroupname argument must be specified.
privatedata	No	Any character string that you wish to pass along to the agent.

Example Javascript opening the chat page in a popup window:

```
window.open('http://MICC
ENTERPRISEWEBSERVER/CustomerChat/Chat.aspx?sessionid={2C310076-91E7-4B8B-870D-
275B0BF23042}&customername=John&customerid=100&email=john@somecompany.com&tenantid
=-1&servicegroupname=Sales');
```

The CustomerChat Web application also contains a default test page for invoking the Chat page. Test arguments may be entered and then the Chat page invoked.

http://<WebServer[:port]>/customerchat/default.aspx



LOCALIZATION

The stock chat client may be localized into additional languages by creating language resource files. To create an additional language, copy the file:

<InstallDir>\Services\Web\CustomerChat\App_LocalResources\Chat.aspx.resx

to a new file in the same folder using the following format:

Chat.aspx.<language>.resx

<language> should be the ISO 639 two-letter culture code and may contain an ISO 3166 two-letter uppercase subculture. For example, to create a general language for French, create the file:

Chat.aspx.fr.resx

To create a language for French with a subculture for Belgium, create the file:

Chat.aspx.fr-BE.resx

Chat.aspx.resx is an XML based language resource file. Be careful not to change the structure of the file. The following text strings may be modified:

NAME	PURPOSE
btnSendMessage.Text	Text of the Send button.
Page.Title	Title of the page.
JS.AgentJoined	Message displayed when an agent joins the conversation.
JS.AgentLeft	Message displayed when an agent leaves the conversation.
JS.AgentTyping	Message displayed in the status area when the agent is typing.
JS.CallCenterClosed	Message displayed when the call center is closed.
JS.ErrorRequestingChat	Message displayed when an error occurs requesting the chat session.
JS.ErrorSendingMessage	Message displayed when an error occurs sending the message to the agent.
Js.Queued	Message displayed when waiting for an agent.
JS.SessionTimeout	Message displayed when the chat session has timed out.
JS.Terminated	Message displayed when the chat session has been terminated.
JS.EWT	Status message showing estimated waiting time.

Note: Any placeholders such as {0} or {1} in the text strings must be maintained. The location in the text string may be moved, but it must exist.

BROWSER SUPPORT

- Internet Explorer v9.0 or Later
- Current Version of Chrome
- Current Version of Firefox

Additional versions may be supported, however, only the current released version of Chrome and Firefox at the time of MiCC Enterprise release have been tested.

IDLE TIMEOUTS

Settings are in place to control session idle timeouts. There is one setting which controls the timeout on the chat client side and one setting at the server. The idle timeout will be the smaller of the two values.

Chat Client

This is controlled by the IdleTimeout setting in the <InstallDir>\Services\Web\CustomerChat\Web.config file. The value is specified in seconds. The default is 300 seconds.

Server

This is controlled by the following registry entry:

Key: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CCChat\Parameters
Value Name (REG_DWORD): CustomerTimeout
Value: Timeout in seconds. Default = 600 seconds.

ESTIMATED WAIT TIME

Estimated wait time can be displayed in the status area if enabled. This is controlled by the ShowEWT setting in the <InstallDir>\Services\Web\CustomerChat\Web.config file. Value values are true and false. The time is based on the estimated wait time at the time of queueing counting down toward 0. The time is shown in minutes with a minimum of 1 minute if available. The estimated wait time may not available based on traffic conditions of the Call Center. Refer to the section on Real-Time Windows in the document **9_1551-LXA119154 – Calculation Methods for Reports** for the estimated wait time calculation.

RUNNING STANDALONE ON A PUBLIC FACING WEB SERVER (DMZ)

The CustomeChat Web application may be used on public facing Web Server (DMZ). The following steps should be performed.

1. Install MiCC Enterprise on the DMZ.

    a. Select Custom installation.

    b. On the Select Features page, select only the Services/Web Services feature.

    c. Set the Broker Location to the main MiCC Enterprise server.

    d. Continue and complete the installation.

2. Remove unnecessary Web applications. The only necessary Web application is CustomerChat. The additional Web applications may be removed, however, they will never be used. If you wish to remove the Web applications, open IIS Manager and remove the following applications under the Default Web Site:

    MiCCEInstallation
    RTI
    scheduler
    seceventws
    seclogonws
    secreportws
    secsapicidriver
    SolidusACD
    WebAgent
    WebApps
    WebCallback

3. Run the MiCC Enterprise Setup Utility.

A. On the Web Server Location page, set the location to where the main MiCC Enterprise Web Services are installed. This is typically the same as the main MiCC Enterprise server. Set the appropriate options for connecting to the main MiCC Enterprise server and Web Server.

4. Open necessary ports. The CustomerChat Web application connects to the Broker Service and Chat Service on the main MiCC Enterprise server. The configured ports for these services must be open and accessible by the DMZ.

5. Ensure machine name resolution. The CustomerChat Web application retrieves the location of the Chat Service from the Broker Service. The location received is depending on how the Chat Service registers itself with the Broker. Typically this is the simple machine name, but it may also be a full qualified domain name or an IP address. The DMZ must be able to access the Chat Service using that received location.

CUSTOM CHAT CLIENT

If the stock MiCC Enterprise Chat client does not suit your needs, you may create your own client. A REST service has been exposed from the MiCC Enterprise Chat service running on the MiCC Enterprise server. You may communicate with the service directly from client side script; however, this would require opening access to the MiCC Enterprise Chat service which is typically behind a firewall. A more typical scenario would be to communicate with the MiCC Enterprise Chat service from a back-end Web application such as from the code-behind in an ASP.NET application.

All requests to the Chat service use JSON data format and are made through the following URL:

http://<MiCCEServer[:port]>/chatservice/<Request>

For example:

http://MICC ENTERPRISESERVER:12616/ChatService/RequestChat

The default port for the Chat service is 12616, but this can be changed on the MiCC Enterprise server through the MiCC Enterprise Setup Utility.

The following example shows calling the RequestChat method from Javascript using JQuery:

```
function RequestChatFromService()
{
	var requestParam = new Object();

	requestParam.SessionID = '{50B5D2FF-57BD-4948-A864-B044A6E9FEB7}';
	requestParam.CustomerID = '100';
	requestParam.CustomerName = 'John';
	requestParam.EmailAddress = 'john@somecompany.com';
	requestParam.TenantID = -1;
	requestParam.ServiceGroupID = 0;
	requestParam.ServiceGroupName = 'Sales';
	requestParam.PrivateData = 'SomePrivateData';

	var DTO = {'request' : requestParam};

	$.ajax({
```

```
            type: “POST”,
            contentType: “application/json; charset=utf-8”,
            url: “http://MICC ENTERPRISESERVER:12616/ChatService/RequestChat”,
            data: JSON.stringify(DTO),
            datatype: “json”,
            success:
            function(data)
            {
                  return data.ChatID;
            }
      });
}
```

IDLE TIMEOUTS

An idle timeout scenario should be handled properly by the custom chat client to terminate the session. There is still a server side timeout as a fail-safe procedure. See the IDLE TIMEOUTS section for the stock chat client.

CHAT SERVICE API

Methods

All methods will throw fault exceptions on failure.

RequestChatResponse RequestChat(RequestChatRequest request)

Requests a chat session.

void LeaveChat(LeaveChatRequest request)

Leaves a chat session

void SendMessage(SendChatMessageRequest request)

Sends a message from the customer to the agent.

void SendTyping(SendTypingRequest request)

Sends a notification to the agent that the customer is typing.

GetEventsResponse GetEvents(GetEventsRequest request)

Gets pending events such as messages received from the agent.

GetQueueInfoResponse GetQueueInfo(GetQueueInfoRequest request)

Gets state information for a service group.

GetChatInfoResponse GetChatInfo(GetChatInfoRequest request)

Gets state information for a chat session.

Types

RequestChatRequest

NAME	TYPE	PURPOSE
SessionID	string	Unique ID of session. This may be used to tag the chat session during archive operations.
CustomerID	string	ID of customer. This may be used to tag the chat session during archive operations.
CustomerName	string	Name of customer.
EmailAddress	string	E-mail address of the customer.
TenantID	int	Tenant ID of the MiCC Enterprise service group where the chat session will be routed. This will be -1 for non-tenanted MiCC Enterprise systems.
ServiceGroupID	int	ID of the service group where the chat session will be routed. Either ServiceGroupID or ServiceGroupName must be specified.
ServiceGroupName	string	Name of the service group where the chat session will be routed. Either ServiceGroupID or ServiceGroupName must be specified.
PrivateData	String	Any character string that you wish to pass along to the agent. This value will not be displayed in the Agent's display, but it is provided to the Agent Integration interface.

RequestChatResponse

NAME	TYPE	PURPOSE
ChatID	string	Contains the unique ID associated to the new chat session.
QueuePosition	int	Position in the queue.
EstimatedWaitTime	int	Estimated wait time in seconds or -1 if an estimate could not be determined.

LeaveChatRequest

NAME	TYPE	PURPOSE

ChatID	string	Unique ID returned from the RequestChat method.

SendChatMessageRequest

NAME	TYPE	PURPOSE
ChatID	string	Unique ID returned from the RequestChat method.
Message	string	Message to send.

SendTypingRequest

NAME	TYPE	PURPOSE
ChatID	string	Unique ID returned from the RequestChat method.

GetEventsRequest

NAME	TYPE	PURPOSE
ChatID	string	Unique ID returned from the RequestChat method.
Timeout	int	Timeout in milliseconds to wait for events.

GetEventsResponse

NAME	TYPE	PURPOSE
Events	List<ChatEvent>	List of receieved events

GetChatInfoRequest

NAME	TYPE	PURPOSE
ChatID	string	Unique ID returned from the RequestChat method.

GetChatInfoResponse

NAME	TYPE	PURPOSE
State	ChatState	Current state.
QueuePosition	int	Position in the queue. Only applicable if State = Queued.

GetQueueInfoRequest

NAME	TYPE	PURPOSE
TenantID	int	Tenant ID of the MiCC Enterprise service. This will be -1 for non-tenanted MiCC Enterprise systems.
ServiceGroupID	int	ID of the service group. Either ServiceGroupID or ServiceGroupName must be specified.
ServiceGroupName	string	Name of the service group. Either ServiceGroupID or ServiceGroupName must be specified.

GetQueueInfoResponse

NAME	TYPE	PURPOSE
EstimatedWaitTime	int	Estimated wait time in seconds or -1 if an estimate could not be determined.

ChatEvent

NAME	TYPE	PURPOSE
EventType	ChatEventType	Type of the event. The type indicates what type of object will be contained in the Data member.
Data	object	Event data. See ChatEventType.

ChatEventType (enum int)

NAME	VALUE	DATA OBJECT TYPE	PURPOSE
AgentMessageReceived	0	AgentMessageReceivedEvent	Message was received from agent.
AgentJoined	1	AgentJoinedEvent	Agent has joined the conversation.
AgentLeft	2	AgentLeftEvent	Agent has left the conversation.
AgentTyping	3	AgentTypingEvent	Agent is typing.
ChatState	4	ChatStateEvent	Chat state has changed.

AgentMessageReceivedEvent

NAME	TYPE	PURPOSE
ChatID	string	Unique ID of the chat session.
Name	string	Sender of the message.

Message	string	Message received.
TimeStamp	DateTime	Time the message was received.

AgentJoinedEvent

NAME	TYPE	PURPOSE
ChatID	string	Unique ID of the chat session.
Name	string	Name of the agent that joined the conversation.

AgentLeftEvent

NAME	TYPE	PURPOSE
ChatID	string	Unique ID of the chat session.
Name	string	Name of the agent that left the conversation.

AgentTypingEvent

NAME	TYPE	PURPOSE
ChatID	string	Unique ID of the chat session.
Name	string	Name of the agent that is typing.

ChatStateEvent

NAME	TYPE	PURPOSE
ChatID	string	Unique ID of the chat session.
State	ChatState	Current state.
QueuePosition	int	Position in the queue. Only applicable if State = Queued.
EstimatedWaitTime	int	Estimated wait time in seconds or -1 if an estimate could not be determined. Only applicable if State = Queued.

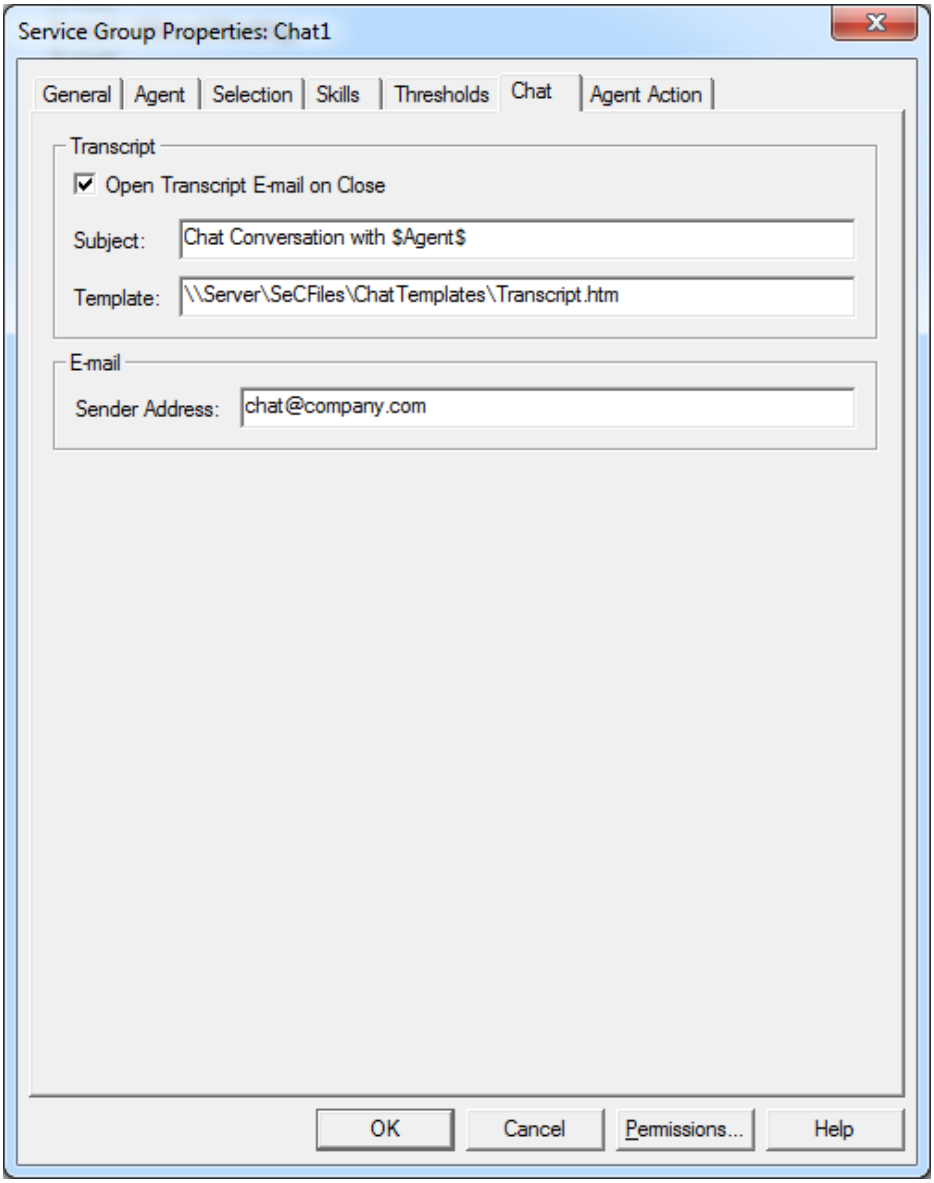
ChatState (enum int)

NAME	VALUE	PURPOSE
Queued	0	Chat has been queued to a service group and is waiting for an agent to be allocated.

Handling	1	Chat is being handled by an agent.
Terminated	2	Chat session has been terminated.

E-MAIL TRANSCRIPT

At the end of a chat session, the customer may be e-mailed a transcript of the conversation. This can be invoked manually from the Agent Chat form or opened automatically when the form is closed. Transcript properties are set on the Chat tab of the Service Group Properties in Configuration Manager.



The transcript is only available to the agent if the Sender E-mail Address is specified for the Service Group. The agent must also have the privilege to send e-mails.

If Open Transcript E-mail on Close is selected, the e-mail form with the transcript will be automatically opened when the agent closes the chat session form.

A custom subject and template may be used for the transcript subject and e-mail body. If these values are empty, a default subject and body will be used.

The transcript template may be any standard text base file (*.txt) or html format files (*.htm, *.html). The filename must be entered in UNC format.

Both the subject and template file may contain placeholders which will be replaced by data associated to the chat as well as the transcript itself. All identifiers are valid for the template file. All identifiers except $Transcript$ are available for the subject.

It is important to ensure that replaceable identifiers are entered in a continuous string in the template file. HTML editors such as Microsoft Word may split the text while inserting HTML format tags. This will prevent the identifiers from being replaced. This may occur if text is identified as a misspelled word. The underlining used in Microsoft Word to indicate the misspelled word will be stored in the HTML file as formatting information. Always ensure that replaceable identifiers are ignored for spell checking.

Replaceable Identifiers

IDENTIFIER	REPLACEMENT
$Transcript$	Chat transcript
$Date$	Current date formatted using the short date format of the current locale
$Time$	Current time formatted using the short time format of the current locale
$Received$	Date and time the e-mail was received formatted using the short date and short time formats of the current locale
$Received.Date$	Date the e-mail was received formatted using the short date format of the current locale
$Received.Time$	Time the e-mail was received formatted using the short time format of the current locale
$ServiceGroup$	Service group name
$ServiceGroup.Name$	Service group name
$ServiceGroup.Email$	Service group e-mail address.
$Agent$	Agent name
$Agent.Name$	Agent name
$Customer$	Customer name

$Customer.Name$	Customer name
$Customer.Email$	Customer e-mail address

CHAT RESPONSES

A response file may be setup for each service group allowing the agents to select predefined messages to be inserted into chat messages. If a response file is configured, a hierarchal list of the responses is displayed in the agent chat form. Refer to the section on E-mail, Chat and SMS Response Files in the document **3_1543-LXA119154 – Advanced Configurations** for the response file format.