T-Server Reference Guide

Platform SDK 9.0.x

12/30/2021

Table of Contents

Welcome to the TLib Reference Guide

When working with T-Server it is important to be familiar with and understand what Tlib Functions, Datatypes, and Unstructured Data is available to work with. This guide provides a quick reference to these pieces of data.

- List of TLib Functions
- List of TLib Datatypes
- List of TLib Unstructured Data

List of TLib Functions

The following table provides a convenient list of TLib Functions that are available.

TAgentLogin	TEventGetStringAttr	TNetworkMerge	TSendEvent
TAgentLogout	TFreeEvent	TNetworkPrivateService	TSendEventEx
TAgentSetIdleReason	TGetAccessNumber	TNetworkReconnect	TSendUserEvent
TAgentSetNotReady	TGetMessageTypeName	TNetworkSingleStepTransfer	TSetCallAttributes
TAgentSetReady	TGetReferenceID	TNetworkTransfer	TSetDNDOff
TAlternateCall	TGetRouteTypeNames	TOpenServer	TSetDNDOn
TAnswerCall	TGetTreatmentTypeNames	TOpenServerEx	TSetInputMask
TApplyTreatment	TGetXCaps	TOpenServerX	TSetMessageWaitingOff
TAttachUserData	TGiveMusicTreatment	TOpenVoiceFile	TSetMessageWaitingOn
TCallCancelForward	TGiveRingBackTreatment	TPlayVoice	TSetMuteOff
TCallSetForward	TGiveSilenceTreatment	TPrivateService	TSetMuteOn
TCancelMonitoring	THoldCall	TQueryAddress	TSetParamHA
TCancelReqGetAccessNumber	TInitiateConference	TQueryCall	TSetRefIDLimit
TClearCall	TInitiateTransfer	TQueryLocation	TSetReferenceID
TCloseServer	TLibSetCompatibMode	TQueryServer	TSetSocketChangeCallback
TCloseVoiceFile	TListenDisconnect	TQuerySwitch	TSingleStepConference
TCollectDigits	TListenReconnect	TReconnectCall	TSingleStepTransfer
TCompleteConference	TLoginMailBox	TRedirectCall	TSockInfoStructure
TCompleteTransfer	TLogoutMailBox	TRegisterAddress	TSyncIsSet
TCopyEvent	TMakeCall	TReleaseCall	TSyncSetSelectMask
TDeleteAllUserData	TMakePredictiveCall	TReserveAgent	TUnregisterAddress
TDeleteFromConference	TMergeCalls	TRetrieveCall	TUpdateUserData
TDeleteUserData	TMonitorNextCall	TRouteCall	TXCapsSupported
TDispatch	TMuteTransfer	TScanServer	connid_to_decimal
TEventGetConnID	TNetworkAlternate	TScanServerEx	connid_to_str
TEventGetIntAttr	TNetworkConsult	TSendDTMF	decimal_to_connid
			str_to_connid

connid_to_decimal

Description

Converts a variable of type `TConnectionID` into a string (decimal).

Parameters

Name	Description
conn_id	A variable of `TConnectionID` type.

Return Values

This function returns a pointer to a string value.

connid_to_str

Description

Converts a variable of type `TConnectionID` into a string (hexadecimal).

Parameters

Name	Description
conn_id	A variable of `TConnectionID` type.

Return Values

The function returns a pointer to a string value.

decimal_to_connid

Description

Converts a variable of string type (decimal) into one of type `TConnectionID`.

Parameters

Name	Description
s	A pointer to a string variable (decimal) that should be converted.

Return Values

The function returns a value of `TConnectionID` type.

str_to_connid

Description

Converts a variable of string type (hexadecimal) into one of type `TConnectionID`.

Parameters

Name	Description
s	A pointer to a string variable (hexadecimal) that should be converted.

Return Values

The function returns a value of `TConnectionID` type.

TAgentLogin

Description

Logs in the agent specified by the `agent_id` parameter to the ACD group specified by the queue parameter.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
queue	Directory number of the ACD group that the agent is requested to be logged in to.
dn	Directory number from which the agent is requested to be logged in to the group.
agent_type	Rank of the agent in question. Refer to the type TAgentType.
agent_id	Identifier of the agent who is requested to be logged in.
passwd	Password that allows the agent to be logged in.
workmode	Work mode to be set for the agent after successful login. Refer to the type TAgentWorkMode.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure. `TAgentLogin()` uses the following extension: `ReasonCode`.

Return Values

Standard (standard-return-values.)

TAgentLogout

Description

Logs the agent out of the ACD group specified by the parameter queue.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
queue	Directory number of the ACD group that the agent is requested to be logged out of.
dn	Directory number of the phone set from which the agent is requested to be logged out of the group.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure. `TAgentLogout()` uses the following extension: `ReasonCode`.

Return Values

Standard (standard-return-values.)

TAgentSetIdleReason

Description

Warning
This function is obsolete.

Sets the `idle reason` code for the telephony object specified by the parameter `dn`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that is requested to be set to idle.
idle_reason	A pointer to the string that contains the idle reason.
reasons	A pointer to a data structure intended to provide additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (standard-return-values.)

TAgentSetNotReady

Description

Sets a state in which the agent is not ready to receive calls. The agent's telephone set is specified by the parameter `dn`; the ACD group into which the agent is logged is specified by the parameter queue.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
queue	Directory number of the ACD group for which the agent is requested to be set to `Not-Ready`.
dn	Directory number of the phone set the agent uses to handle calls.
workmode	Work mode that is requested to be set for the agent. Refer to the type TAgentWorkMode.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure. `TAgentSetNotReady()` uses the following extension: `ReasonCode`.

Return Values

Standard (See standard-return-values.)

Comments

The basic status of unavailability set by the function itself may be further modified by the value of `workmode`.

TAgentSetReady

Description

Sets a state in which the agent is ready to receive calls. The agent's phone set is specified by the parameter dn; the ACD group into which the agent is logged is specified by the parameter queue.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
queue	Directory number of the ACD group for which the agent is requested to be set to Ready.
dn	Directory number of the telephone set the agent uses to handle calls.
workmode	Work mode that is requested to be set for the agent. Refer to the type TAgentWorkMode.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure. `TAgentSetReady()` uses the following extension: `ReasonCode`.

Return Values

Standard (standard-return-values.)

Comments

The basic status of availability (that is, the status at login time) set by the function itself may be further modified by the value of `workmode`.

TAlternateCall

Description

On behalf of the telephony object specified by the parameter dn, places the active call specified by the parameter `current_conn_id` on hold and connects the call specified by the parameter `held_conn_id`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the calls are requested to be swapped.
held_conn_id	Connection identifier of the call that is requested to be connected.
current_conn_id	Connection identifier of the active call that is requested to be placed on hold.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TAnswerCall

Description

Answers the alerting call specified by the parameter `conn_id`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that will answer the call.
conn_id	Connection identifier of the call that should be answered.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TApplyTreatment

Description

On behalf of the telephony object specified by the parameter dn, applies a treatment specified by treatment to a call specified by `conn_id`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number that specifies the telephony object to be treated.
conn_id	Connection identifier of the call to be treated.
treatment	Type of treatment being requested. Refer to the type TTreatmentType.
parameters	A pointer to a data structure that provides additional parameters associated with the treatment type. Refer to the type TTreatmentType.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TAttachUserData

Description

Warning
This function is obsolete.

On behalf of the telephony object specified by the parameter dn, attaches the User Data structure specified by the parameter `user_data` to the T-Server information that is related to the call specified by the parameter `conn_id`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the data is requested to be attached.
conn_id	Connection identifier of the call in question.
user_data	A pointer to the user data that should be attached to the call.

Return Values

Standard (See standard-return-values.)

T-Server is not concerned with the contents of the data structure specified by the parameter user_data. The new data specified by the parameter user_data is added to a call without removing the existing data. As you can see from the table below, new data does not replace existing data. Instead, T-Server adds to the user_data structure, even if it has the same key.

Example of Addition (Attaching) to User_Data

Existing user_data	user_data to Be Added	Result
data1=1234 data2=5678 data3=9012	data3=9999 data4=1212 data5=5555	data1=1234 data2=5678 data3=9012 data3=9999 data4=1212 data5=5555

TCallCancelForward

Description

Sets the Forwarding feature to `Off` for the telephony object that is specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object for which forwarding is requested to be turned off.
forward_mode	Mode of forwarding. Refer to the type TForwardMode.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TCallSetForward

Description

Sets the Forwarding feature to On for the telephony object that is specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object for which forwarding is to be turned on.
destination	Directory number of the party calls that are to be forwarded to.
forward_mode	Mode of forwarding. Refer to the type TForwardMode.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TCancelMonitoring

Description

A request by a supervisor to cancel the monitoring of the calls delivered to the agent. If this request is successful, T-Server distributes `EventMonitoringCancelled` to all clients registered on the supervisor's and agent's DNs.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Supervisor's DN from which the monitoring will be canceled.
agent_dn	Agent's DN that was monitored.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TCancelReqGetAccessNumber

Description

Allows an application to send a cancellation request for a previously executed `TGetAccessNumber()` function.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
xref_id	The value of the reference ID returned by the `TGetAccessNumber` request.

Return Values

Standard (See standard-return-values.)

TClearCall

Description

Deletes all parties, that is, all telephony objects, from the call specified by `conn_id` and disconnects the call.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that is requested to be released from the call.
conn_id	Connection identifier of the call, from which the telephony object in question is requested to be released.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TCloseServer

Description

Closes the client/T-Server communications session specified by the parameter `server`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.

Return Values

0 — Successful execution of the requested function.
-1 — Error condition. Returned to the application if the request has not been processed due to an incorrect function statement.

TCloseVoiceFile

Description

Closes the voice file specified by the parameter `file_handle` on behalf of the telephony object specified by the parameter dn, which earlier logged in to a mailbox using the function `TLoginMailBox()`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the file is closed.
file_handle	Handle of the file in question. Returned as a value of the `FileHandle` parameter in the `EventVoiceFileOpened` event.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (standard-return-values.)

TCollectDigits

Description

Collects digits from the caller.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the digits are collected.
conn_id	Connection identifier of the call for which the digits need to be collected.
num_digits	Number of digits to be collected.
term_digits	A pointer to the symbol(s) that marks the end of input.
cflag	Clear flag indicates whether previously gathered information should be cleared before digit collection starts. Refer to the type TClearFlag.
timeout	Time interval, in seconds, allocated for digits collection.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (standard-return-values.)

TCompleteConference

Description

Completes a previously initiated conference by merging the held call specified by the parameter `held_conn_id` with the active consultation call specified by the parameter `current_conn_id` on behalf of the telephony object specified by dn. Assigns `held_conn_id` to the resulting conference call. Clears the consultation call specified by the parameter `current_conn_id`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the conference is made.
held_conn_id	Connection identifier of the original call; will be assigned to the resulting conference call.
current_conn_id	Connection identifier of the consultation call; the consultation call will be deleted after completion of operation.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TCompleteTransfer

Description

On behalf of the telephony object specified by the parameter dn, completes a previously initiated two-step transfer by merging the held call specified by the parameter `conn_id` with the active consultation call specified by the parameter `current_conn_id`. Assigns `held_conn_id` to the resulting call. Releases the telephony object specified by the parameter dn from both calls and clears the consultation call specified by the parameter `current_conn_id`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the transfer is made.
held_conn_id	Connection identifier of the original call; will be assigned to the resulting transferred call.
current_conn_id	Connection identifier of the consultation call; will be deleted after completion of operation.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

Comments

The transfer may be completed both before and after the consulted party answers.

TCopyEvent

Description

Creates a copy of the event specified by the parameter `event` in the application memory.

Parameters

Name	Description
event	A pointer to the event data structure that is requested to be copied.

Return Values

A pointer to the copy of the event that this function creates.

Comments

This function does not generate any messages to T-Server.

TDeleteAllUserData

Description

On behalf of the telephony object specified by the parameter `dn`, deletes all the user data attached to the call specified by the parameter `conn_id`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the user data is requested to be deleted.
conn_id	Connection identifier of the call in question.

Return Values

Standard (See standard-return-values.)

TDeleteFromConference

Description

A telephony object specified by `dn` deletes the telephony object specified by `dn_to_drop` from the conference call specified by `conn_id`. The client that invokes this service must be a party on the call in question.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that invokes delete-from-the-conference service.
conn_id	Connection identifier of the conference call from which the object is requested to be deleted.
dn_to_drop	Reference to the telephony object that is requested to be deleted from the conference call. This parameter can be either a `dn` or a `party_id`, where `party_id` is the party identifier given by the switch.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TDeleteUserData

Description

On behalf of the telephony object specified by the parameter dn, deletes the key-value pair specified by the parameter key from the user data attached to the call specified by the parameter `conn_id`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the key-value pair is requested to be deleted.
conn_id	Connection identifier of the call in question.
key	A pointer to the key of the pair that should be deleted.

Return Values

Standard (See standard-return-values.)

TDispatch

Description

Whenever there is any activity on the socket, this function dispatches the data communicated by T-Server to the user-defined callback function (designated in the call to the functions `TOpenServer()` and `TOpenServerEx()`), which handles incoming events.

Parameters

Name	Description
server	Local server handle to the T-Server in question.

Return Values

0 — Returned when the function could not process any events.
> 0 — The number of events the function has been able to read from the input buffer and deliver to the user-defined callback function designated in the call to `TOpenServer()` or `TOpenServerEx()`. If a connection to T-Server has been lost, the function causes `EventServerDisconnected` to be delivered in addition to the events communicated by T-Server and returns the number of communicated events, including `EventServerDisconnected`.

Comments

This function does not generate any messages to T-Server.

TEventGetConnID

Description

Returns the connection ID value of the attribute specified by the parameter `attr` (or the value `NO_CONN_ID` if this attribute is not present in `TEvent`).

Parameters

Name	Description
ev	A pointer to the event data structure from which the connection ID is returned.
attr	The attribute for which the connection ID is sought.

Return Values

The function returns a value of `TConnectionID` type.

TEventGetIntAttr

Description

Returns the integer value of the attribute specified by the parameter `attr` (for attributes with values of `integer` type).

Parameters

Name	Description
ev	A pointer to the event data structure from which the integer value is returned.
attr	The attribute whose integer value is sought.

Return Values

This function returns an integer value.

TEventGetStringAttr

Description

Returns the string value of the attribute specified by the parameter `attr` (for attributes with values of `string` type).

Parameters

Name	Description
ev	A pointer to the event data structure from which the string value is returned.
attr	The attribute whose string value is sought.

Return Values

This function returns a pointer to a string value.

TFreeEvent

Description

Deletes a copy of the event specified by the parameter `event` from the application memory.

Parameters

Name	Description
event	A pointer to the event data structure that is requested to be deleted.

Return Values

None

Comments

This function does not generate any messages to T-Server.

TGetAccessNumber

Description

Allows an application to get an access number to reach the destination switch.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object from which the request is initiated.
conn_id	Connection identifier of the call that is requested to be transferred to the destination switch.
destination	Directory number of the party where the call is requested to be transferred.
location	Name of the remote location in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
xroute_type	Type of routing. Refer to the type TXRouteType.
user_data	A pointer to call-related user data that should be attached to the call.
reasons	Reserved for future use. For forward compatibility, this parameter must be `NULL`.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

Comments

When a client application sends a `TGetAccessNumber` request to T-Server, T-Server will return the access number depending on the location parameter value:

- If the location parameter is `NULL` or identical to the location of the T-Server (T-Server A) receiving this request, T-Server A will process this request and return the access number that will be used by an application to reach the switch that this T-Server A is connected to.

- If the location parameter is the location of another T-Server (T-Server B), T-Server A after receiving this request will pass it to T-Server B. After T-Server A receives the access number from T-Server B, T-Server A will return the access number that will be used by an application to reach the switch that T-Server B is connected to.

The application can cancel the pending request by sending a `TCancelReqAccessNumber` request.

TGetMessageTypeName

Description

Returns the text of the message type specified by the parameter `msg_type`.

Parameters

Name	Description
msg_type	A message type as received in the `TEvent` structure of the event.

Return Values

A pointer to a static buffer that contains the requested message name. When an invalid message type is specified, the buffer contains the message ID converted to text.

TGetReferenceID

Description

Gets the reference ID value that is used by T-Library for the next T-Library request.

Return Values

Returns the current reference ID.

Comments

This function does not generate any messages to T-Server.

TGetRouteTypeNames

Description

Returns the text of the route type specified by the parameter `rtype`.

Parameters

Name	Description
rtype	A route type as received in the `TEvent` structure of the event.

Return Values

A pointer to a static buffer that contains the requested route-type name. When an invalid route type is specified, the buffer contains the route ID converted to text.

TGetTreatmentTypeNames

Description

Returns the text of the treatment type specified by the parameter `ttype`.

Parameters

Name	Description
ttype	A treatment type as received in the `TEvent` structure of the event.

Return Values

A pointer to a static buffer that contains the requested treatment name. When an invalid treatment type is specified, the buffer contains the treatment ID converted to text.

TGetXCaps

Description

Identifies which of the capabilities listed in the tables below are supported by the T-Server in use.

Parameters

Name	Description
eventServerInfo	Pointer to the `TEvent` containing an event of type `EventServerInfo`.
type	A representation of the extended capabilities available for the T-Server in question. This parameter is in one of two forms: one of the message IDs listed in the tables below, or the value `XCAPS_ANY_MESSAGE` (applicable only for agent work mode capabilities). Use of `XCAPS_ANY_MESSAGE` returns the capabilities for the given subtype supported in any message (either function call or `TEvent`).
subtype	A representation of the extended subtype capabilities available for the T-Server in question. This parameter is limited to the subtypes listed in the following tables.

Return Values

The value returned by the function `TGetXCaps()` contains a bit mask describing which enumerator values are supported in the specified combination of request/event ID and attribute.

Important

The mask returned for this function is not intended for direct use. Instead, you should analyze it with the function `TXCapsSupported()`.

TGiveMusicTreatment

Description

On behalf of the telephony object specified by the parameter `dn`, connects the call specified by the parameter `conn_id` to the music source specified by the parameter `music_dn`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the call is connected to the music source.
conn_id	Connection identifier of the call that is requested to be connected to the music source.
music_dn	Directory number of the music source.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TGiveRingBackTreatment

Description

On behalf of the telephony object specified by the parameter `dn`, connects the call specified by the parameter `conn_id` to a Ring Back Tone source, which is specified in the switch.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the call is connected to a Ring Back Tone source.
conn_id	Connection identifier of the call that is requested to be connected to a Ring Back Tone source.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TGiveSilenceTreatment

Description

On behalf of the telephony object specified by the parameter `dn`, connects the call specified by the parameter `conn_id` to a source of silence, which is specified in the switch.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object.
conn_id	Connection identifier of the call that is requested to be connected to the silence source.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

THoldCall

Description

On behalf of the telephony object specified by the parameter dn, places on hold the call specified by the parameter `conn_id`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the call should be placed on hold.
conn_id	Connection identifier of the call that is requested to be placed on hold.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TInitiateConference

Description

On behalf of the telephony object specified by the parameter `dn`, places the existing call specified by the parameter `conn_id` on hold and originates a consultation call from the same telephony object to the called party, which is specified by the parameter `destination` with the purpose of a conference call.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that initiates the conference.
conn_id	Connection identifier of the call that is requested to be placed on hold.
destination	Directory number of the party to be dialed.
location	Name of the remote location in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
user_data	A pointer to the user data that should be attached to the call.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure. `TInitiateConference()` uses the following extension: `ConsultUserData`.

Return Values

Standard (See standard-return-values.)

TInitiateTransfer

Description

On behalf of the telephony object specified by the parameter `dn`, places the existing call specified by the parameter `conn_id` on hold and originates a consultation call from the same telephony object to the called party, which is specified by the parameter `destination` for the purpose of a two-step transfer.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that initiates the conference.
conn_id	Connection identifier of the call that is requested to be placed on hold.
destination	Directory number of the party to be dialed.
location	Name of the remote location in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
user_data	A pointer to the user data that should be attached to the call.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure. `TInitiateConference()` uses the following extension: `ConsultUserData`.

Return Values

Standard (See standard-return-values.)

TLibSetCompatibMode

Description

Provides an indication as to which attribute should be used to apply to a call's connection ID. This function allows for backward compatibility to T-Server versions which assumed a given call would have a unique identifier across multiple sites. `AttributeFirstTransferConnID` is used instead of `AttributeConnID` if `AttributeFirstTransferConnID` is present in the message from the T-Server.

Parameters

Name	Description
mode	Mode specifying whether `AttributeFirstTransferConnID` is to be substituted for `AttributeConnID` when referring to the connection ID for this call.

Return Values

Standard (See standard-return-values.)

Comments

This function does not generate any messages to T-Server.

TListenDisconnect

Description

On an existing conference call, sets Deaf mode for the party specified by the parameter `listener_dn`. For example, if two agents wish to consult privately, the subscriber may temporarily be placed in Deaf mode.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf this request is made; must be a party in the specified call.
listener_dn	Directory number of the telephony object that is requested to be set to Deaf mode.
conn_id	Connection identifier of the call from which the telephony object is requested to be released.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TListenReconnect

Description

On an existing conference call, cancels Deaf mode for the party defined by the parameter `listener_dn`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf this request is made; must be a party in the specified call.
listener_dn	Directory number of the telephony object for which Deaf mode should be canceled.
conn_id	Connection identifier of the call that the telephony object in question is requested to be connected to after Deaf mode is canceled.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TLoginMailBox

Description

Logs in the telephony object specified by the parameter dn to the mailbox specified by the parameter `mbox_number`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that will be logged in to the mailbox.
mbox_number	A pointer to the directory number of the mailbox number.
mbox_passwd	A pointer to the mailbox password.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (standard-return-values.)

TLogoutMailBox

Description

Logs the telephony object specified by the parameter dn out of the mailbox that it is logged in to.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that will be logged out of the mailbox.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (standard-return-values.)

TMakeCall

Description

Originates a regular call from the telephony object specified by the parameter dn to the called party specified by the parameter `destination`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object from which the call is requested to be made.
destination	Directory number of the party that should be dialed.
location	Name of the remote location in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
call_type	Refer to the type TMakeCallType.
user_data	A pointer to the user data that should be attached to the call.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TMakePredictiveCall

Description

Originates a predictive call from the telephony object specified by the parameter dn to the called party specified by the parameter `destination`.

A predictive call occurs before any agent-subscriber interaction. For example, if the call is answered by a fax machine, no agent connection is made. Agent connection is made only if there is an actual subscriber present on the line.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object from which the call is requested to be made. (See note below.)
destination	Directory number of the party to be dialed.
ring_timeout	Timeout, in seconds, after which the call should be considered unanswered.
user_data	A pointer to the user data that should be attached to the call.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure. `TMakePredictiveCall()` uses the following extensions: `VoiceDest`, `AnsMachine`, and `FaxDest`. To specify a Calling Party Number (CPN), it may also use `CPNType`, `CPNPlan`, `CPNPresentation`, `CPNScreening`, and `CPNDigits`.

Return Values

Standard (See standard-return-values.)

Important

When selection of a target DN depends on the results of the call answer detection, T-Server generates `EventDialing` and `EventQueued` for the call origination DN specified by the dn parameter. Then, T-Server diverts a call to the target DN specified for this call type in the `extensions` parameter and generates `EventDiverted` for the DN specified by the dn parameter and appropriate events for the target DN.

TMergeCalls

Description

On behalf of the telephony object specified by the parameter `dn`, merges the held call specified by the parameter `held_conn_id` with the active call specified by the parameter `current_conn_id` in a manner specified by the parameter `merge_type`. The resulting call will have the same `conn_id` as the held call.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the calls are merged.
held_conn_id	Connection identifier of the original call.
current_conn_id	Connection identifier of the active call.
merge_type	Specifies whether merging should result in a transferred or conferenced call. Refer to the type TMergeType.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TMonitorNextCall

Description

A request by a supervisor to monitor (be automatically conferenced in as a party on) the next call delivered to an agent. Supervisors can request to monitor one subsequent call or all calls until the request is explicitly canceled. If a request is successful, `EventMonitoringNextCall` is distributed to all clients registered on the supervisor's and agent's DNs.

Supervisors start monitoring each call in Mute mode. To speak, they must execute the function `TSetMuteOff()`.

Supervisors can request to monitor only one agent at a time. If they make a request to monitor another agent, the first request is automatically canceled. Only one supervisor can monitor the next call of a particular agent—if another supervisor places a request to monitor the same agent, the request is rejected.

The monitoring is automatically canceled when either the supervisor or the agent logs out.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Supervisor's DN from which the monitoring will be made.
agent_dn	Agent's DN that will be monitored.
monitor_type	Indicates whether a supervisor wants to monitor one call or all subsequent calls.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TMuteTransfer

Description

Initiates a transfer of the call specified by the parameter `conn_id` from the telephony object specified by the parameter dn to the party specified by the parameter `destination`; completes the transfer without waiting for the destination party to pick it up. Releases the telephony object specified by the parameter dn from the call.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the call is transferred.
conn_id	Connection identifier of the call that should be transferred.
destination	Directory number of the party the call is transferred to.
location	Name of the remote location in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
user_data	A pointer to the user data that is to be attached to the call.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure. `TMuteTransfer()` uses the following extension: `ConsultUserData`.

Return Values

Standard (See standard-return-values.)

TNetworkAlternate

Description

Places the current call on hold (disconnects it from a conference) and connects to another call.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
ctrl_dn	DN of the controlling party, for a premise T-Server, or the route point where the call was initially routed, for a network T-Server.
conn_id	Connection ID of the call in question.
home_location	The name of the remote location that carries out the request in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
reasons	A pointer to a data structure which provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TNetworkConsult

Description

For network T-Servers, makes a request to place the origination party on hold and to connect a new leg.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
ctrl_dn	DN of the controlling party, for a premise T-Server, or the route point where the call was initially routed, for a network T-Server.
conn_id	Connection ID of the call in question.
home_location	The name of the remote location that carries out the request in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
destination	Destination DN for the consultation. (This may be absent if URS uses other data for the selection of the destination.)
dest_location	The name of the remote location to which the consultation call is being made, in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
reasons	A pointer to a data structure which provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TNetworkMerge

Description

Initiates a request for the consult leg be joined with the existing primary call.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
ctrl_dn	DN of the controlling party, for a premise T-Server, or the route point where the call was initially routed, for a network T-Server.
conn_id	Connection ID of the call in question.
home_location	The name of the remote location that carries out the request in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
reasons	A pointer to a data structure which provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TNetworkPrivateService

Description

Initiates a request to perform a device-specific service.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
serviceID	Service identifier for the information being passed.
ctrl_dn	DN of the controlling party, for a premise T-Server, or the route point where the call was initially routed, for a network T-Server.
conn_id	Connection ID of the call in question.
home_location	A pointer to the name of the remote location that carries out the request in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
reasons	A pointer to a data structure which provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TNetworkReconnect

Description

Initiates a request to drop the consultation leg and reconnect to the origination party of the primary call.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
ctrl_dn	DN of the controlling party, for a premise T-Server, or the route point where the call was initially routed, for a network T-Server.
conn_id	Connection ID of the call in question.
home_location	The name of the remote location that carries out the request in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
reasons	A pointer to a data structure which provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TNetworkSingleStepTransfer

Description

Initiates a request for the network T-Server to transfer a call in one step (without a consultation phase).

Parameters

Name	Description
server	Local server handle to the T-Server in question.
ctrl_dn	DN of the controlling party, for a premise T-Server, or the route point where the call was initially routed, for a network T-Server.
conn_id	Connection ID of the call in question.
home_location	The name of the remote location that carries out the request in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
destination	Destination DN for the consultation. (This may be absent if URS uses other data for the selection of the destination.)
dest_location	The name of the remote location to which the consultation call is being made, in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
reasons	A pointer to a data structure which provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TNetworkTransfer

Description

Initiates a request that the existing called party in a consultation call be connected to the main call, and that the requestor be dropped. (That is, complete the transfer.)

Parameters

Name	Description
server	Local server handle to the T-Server in question.
ctrl_dn	DN of the controlling party, for a premise T-Server, or the route point where the call was initially routed, for a network T-Server.
conn_id	Connection ID of the call in question.
home_location	The name of the remote location that carries out the request in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
reasons	A pointer to a data structure which provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TOpenServer

Description

This function is available for backward compatibility only. It is used when configuration information is specified in a configuration file, and not derived from Configuration Server. If the configuration file is not located in the same directory from which an application is running, then the location of the configuration file has to be described in the environment variable `TSERVER_HOME`. This function opens a communication session to T-Server specified by the parameter `server_name` and designates the event-processing function that will be invoked each time an event from that T-Server is detected.

Parameters

Name	Description
server_name	A pointer to the name of T-Server to be contacted. The name of T-Server specifies the section name in the configuration file. If it is not specified, the default section name is `default`.
dispatch_function	A pointer to the function that will be called when an event from T-Server arrives.
application_name	A pointer to the application name.
application_password	A pointer to the string containing the tenant name and the tenant password separated by a slash (`/`).
open_mode	Communication mode (synchronous or asynchronous). Refer to the type TOpenMode for information on the implications of using one mode versus the other.

Return Values

`> 0` — A local server handle to the specific T-Server; in other words, a unique identifier assigned by T-Library to the connection between a client and T-Server. The handle will be used subsequently to send requests to T-Server throughout this communication session.

`< 0` — Error condition. Returned to the application if the communication session with the T-Server could not be established.

TOpenServerEx

Description

Opens a communication session to the T-Server specified by the parameters `server_host_name` and `server_port`, which are taken from Configuration Server. It designates the event-processing function that will be invoked each time an event from T-Server is detected.

Use the `TOpenServerEx()` function (rather than `TOpenServer()`) to reconnect after a T-Server session has been unexpectedly disconnected or if an application already knows the parameters `server_host_name` and `server_port`.

Parameters

Name	Description
server_host_name	A pointer to the host name of T-Server to be contacted.
server_port	The port number.
dispatch_function	A pointer to the function that will be called when an event from T-Server arrives.
application_name	A pointer to the application name.
application_password	A pointer to the string containing the tenant name and the tenant password separated by a slash (/).
open_mode	Communication mode (synchronous or asynchronous). Refer to the type TOpenMode for information on the implications of using one mode versus the other.

Return Values

> `0` - A local server handle to the specific T-Server; in other words, a unique identifier assigned by T-Library to the connection between a client and T-Server. The handle will be used subsequently to send requests to T-Server throughout this communication session.

< `0` - Error condition. Returned to the application if the communication session with the T-Server could not be established.

TOpenServerX

Description

Opens a communication session to the T-Server specified by the `conn_param` argument, and designates the event-processing function to be invoked each time an event from T-Server is detected. Use this function (instead of `TOpenServer()` or TOpenServerEx()) to set up an ADDP-compliant connection to T-Server. ADDP provides timely detection of TCP/IP connections, or failures, where the TCP/IP itself fails to provide the same information in a timely manner.

Parameters

Name	Description
conn_param	A pointer to an XKVList type containing a description of the connection. The following table describes the information contained in this XKVList .
dispatch_function	A pointer to the function that will be called when an event from T-Server arrives.
application_name	A pointer to the application name.
application_password	A pointer to the string containing the tenant name and the tenant password separated by a slash (`/`).
open_mode	Set of flags specifying the mode in which the connection is open.

Return Values

`> 0` — A local server handle to the specific T-Server; in other words, a unique identifier assigned by T-Library to the connection between a client and T-Server. The handle is used subsequently to send requests to T-Server throughout this communication session.

`< 0` — Error condition. Returned to the application if the communication session with the T-Server could not be established.

TOpenVoiceFile

Description

Opens the voice file that is specified by the parameter `file_name` on behalf of the telephony object specified by the parameter `dn` and that has earlier logged in to a mailbox using the function `TLoginMailBox()`.

`TOpenVoiceFile()` returns a file handle upon generation of `TEventVoiceFileOpen`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the file is requested to be opened.
file_name	A pointer to the name of the file that is requested to be opened.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (standard-return-values.)

TPlayVoice

Description

Plays the voice message contained in the file specified by the parameter `file_handle` for the telephony object specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the message is played.
conn_id	Connection identifier of the call to which the message will be played.
file_handle	Handle of the file in question. Returned as a value of the `FileHandle` parameter in the `EventVoiceFileOpened` event.
segments	A pointer to the prerecorded speech segment that is to be played.
iflag	Interrupt flag indicating whether playback should be interrupted whenever a touchtone button is pressed. Refer to the type TInterruptFlag.
cflag	Clear flag indicating whether previously gathered information should be cleared before playback starts. Refer to the type TClearFlag.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure that takes into account switch-specific features that cannot be described by the above parameters.

Return Values

Standard (standard-return-values.)

TPrivateService

Description

Passes information and requests services (such as Split Call, Set Feature, change T-Server behavior, and so on) that are supported only by certain T-Servers, and which are not covered by general feature requests.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
serviceID	Service identifier for the information being passed.
dn	Directory number of the telephony object on whose behalf the information is provided.
conn_id	A variable of `TConnectionID` type referring to the call, if any, related to this action.
user_data	A pointer to the user data that should be attached to the call.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

Comments

Note that the function `TPrivateService()` is for use only with a select number of T-Servers which require this additional means for exchanging particular information with the switch. Consult T-Server-specific documentation for the applicability of this request.

Depending on the circumstances, T-Server may send `EventACK`, `EventPrivateInfo`, or `EventError` to the client calling the function `TPrivateService()`. In cases of conventional events (such as `EventRinging`) being direct results of a private request, T-Server will not send private events in response to a private request.

`EventACK` will be sent when:

- T-Server cannot link the requested service and the resulting events.
- The service being requested on behalf of the requesting device is to be executed on a different device.

`EventPrivateInfo` will be sent when:

- The resulting event is a direct consequence of the requested service.
- T-Server, a device, or a call has changed its state, and this change cannot be presented via an existing Genesys protocol, and support for representing this change is required by the business needs of the enterprise.

`EventError` will be sent when:

- The requested service is not supported.
- The request parameters are not correct.
- The requested service fails while processing on the media device.

TQueryAddress

Description

Requests the information specified by the parameter `info_type` about the telephony object specified by the parameter `dn`, or the parameter `queue`, or both. Refer to the type TAddressInfoType. If the query type is supported, the requested information will be returned in `EventAddressInfo`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
queue	Directory number of the ACD group about which the information is requested (used only if `info_type=AddressInfoAgentStatus;`).
dn	Directory number of the telephony object in question.
addr_type	Type of telephony object in question. Refer to the type TAddressType.
info_type	Type of the requested information. Refer to the type TAddressInfoType.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

Important

If `info_type` is set to `AddressInfoDNStatus`, T-Server will use the available information without sending a request to the switch. This is the only `info_type` supported by all T-Servers.

TQueryCall

Description

Requests the information specified by `info_type` about the telephony object specified by `conn_id`. If the query type is supported, the requested information will be returned in `EventPartyInfo`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object in question. This parameter is reserved for access control.
conn_id	Current connection identifier of the call.
info_type	Type of the requested information. Refer to the type TCallInfoType.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TQueryLocation

Description

Allows a client to receive information specified by the `info_type` parameter about one or more remote locations specified by the location `parameter`. T-Server generates an `EventLocationInfo` in response to the function `TQueryLocation()`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
info_type	The type of requested information.
location	The name of the location in question.
extensions	Reserved for future use. For forward compatibility, set this parameter to `NULL`.

Return Values

Standard (See standard-return-values.)

TQueryServer

Description

Requests information about T-Server specified by the parameter `server`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
extensions	A pointer to an additional T-Server–specific data structure.

Return Values

Standard (See standard-return-values.)

TQuerySwitch

Description

Requests from the T-Server, specified by the parameter `server`, information about the switch to which T-Server is connected.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
info_type	The type of requested information.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TReconnectCall

Description

Releases the telephony object specified by the parameter dn from the active call specified by the parameter `current_conn_id` and retrieves the previously held call, specified by the parameter `held_conn_id`, to the same object. This function is commonly used to clear an active call and to return to a held call, or to cancel a consult call (because of no answer, called device busy, and so on) and then to return to a held call.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that should be reconnected to the original call.
held_conn_id	Connection identifier of the original call that should be retrieved.
current_conn_id	Connection identifier of the active call from which the telephony object in question should be released.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TRedirectCall

Description

Requests that the call be redirected, without an answer, from the party specified by the parameter dn to the party specified by the parameter `dest_dn`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that the call redirects from. For most switches, this DN must be in Ringing state.
dest_dn	Directory number of the destination telephony object that the call is redirected to.
conn_id	Connection identifier of the call to be redirected.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TRegisterAddress

Description

Registers an application so that it can send feature requests and receive events regarding the telephony object specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number that specifies the telephony object in question.
mode	Registration mode specifying whether information about this telephony object will be available to other applications. Refer to the type TRegisterMode.
ctrmode	Control mode specifying whether the fact of registration is to be relayed to the switch. Refer to the type TControlMode. Specifying this parameter will make sense only if the CTI link protocol of the switch in question supports the telephony object registration request.
type	Address type specifying the telephony object in question. Refer to the type TAddressType.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

Comments

This function should only be used after receiving `EventLinkConnected` from T-Server.

TReleaseCall

Description

Releases the telephony object specified by the parameter dn from the call specified by the parameter `conn_id`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that should be released from the call.
conn_id	Connection identifier of the call from which the telephony object in question should be released.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TReserveAgent

Description

The Agent Reservation feature provides a way to resolve race conditions between several T-Server clients attempting to access the same agent. T-Server collects concurrent reservation requests during a configurable time interval (set with the T-Server `request-collection-time` option). Once this collection time expires, T-Server randomly chooses one of the highest priority requests, positively responds to it with `EventAgentReserved`, and fails the rest of requests, rejecting each with `EventError`.

The successful agent reservation record remains active for some duration of time (the function's `duration` parameter). Setting `duration` to `0` cancels the reservation record. An attempt to cancel a non-existing reservation generates `EventError`. Setting the duration to `-1` forces T-Server to use a preconfigured duration. (See the `reservation-time` T-Server option.)

Any subsequent attempt to reserve an already-reserved agent fails and results in the generation of `EventError`. There is an exception, however, if the `reject-subsequent-request` option is set to `false`. In this case, the T-Server's client whose reservation request succeeded has the ability to reset the length of duration for an agent that is already reserved.

The agent being reserved can be defined by `agent_dn`, `agent_id`, or `agent_place`—at least one of these parameters must be present in the request. If more than one of these parameters is specified in the request, the reservation occurs on all specified entities.

The Agent Reservation feature does not prevent the reserved agent from changing state, receiving direct calls or calls distributed from ACD Queues. This feature is only intended as a way of synchronizing the operation of several T-Server clients. The agent is temporarily reserved for exclusive use, and other potential clients must either wait for the reservation to end, or choose an alternative agent.

Parameters

Name	Description
server	Local server handle to T-Server in question.
agent_dn	DN number to be reserved (may be `NULL`).
agent_id	Agent to be reserved (may be `NULL`).
agent_place	Place to be reserved (may be `NULL`).
duration	Required reservation time, in milliseconds. Specify `-1` to use default of `100000` milliseconds. The default value is set by the `reservation-time` configuration option.
priority	Request priority. Specify `-1` to use default of `0`, the

Name	Description
	lowest priority value.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TRetrieveCall

Description

Connects the held call specified by the parameter `conn_id` to the telephony object specified by the parameter `dn`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object to which the held call is requested to be connected.
conn_id	Connection identifier of the held call.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TRouteCall

Description

Routes the call identified to the specified destination.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object where the call is residing.
conn_id	Connection identifier of the call that is requested to be routed.
destination	Directory number of the telephony object to which the call is requested to be routed.
location	Name of the remote location in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
route_type	Type of routing. Refer to the type TRouteType.
dnis	The new DNIS if the route_type parameter is set to `RouteTypeOverwriteDNIS`.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

TScanServer

Description

Scans T-Server for one or more incoming messages for a time interval specified by the parameter `timeout`. This function does not affect delivery of pending messages and does not process timers, so it should not be used with connections opened in asynchronous mode or if ADDP protocol is used. (Use `TScanServerEx()` to process timers.)

Parameters

Name	Description
server	Local server handle to the T-Server in question.
timeout	Time interval, in seconds, during which a message from T-Server will be sought.

Return Values

`>0` — Number of received messages.

`0` — The timeout is expired.

`-1` — Error condition. Returned to the application if the request has not been processed due to an incorrect function statement.

Comments

This function does not generate any messages to T-Server.

TScanServerEx

Description

Scans T-Server for one or more incoming messages and special events for a time interval specified by the parameter `timeout`, and processes the timers and the pending messages as specified by the parameter `scan_mode`. This function is used to synchronize with the socket operation while operating in asynchronous mode, or in synchronous mode when ADDP is used.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
timeout	Time interval, in seconds, during which a message from T-Server will be sought.
scan_mode	The scanning mode.

Return Values

`>0` — Number of received messages.

`0` — The timeout is expired.

`-1` — Error condition. Returned to the application if the request has not been processed due to an incorrect function statement.

Comments

This function does not generate any messages to T-Server.

TSendDTMF

Description

On behalf of the telephony object specified by the parameter dn, sends digits that are expected by an interactive voice response system.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the digits are requested to be sent.
conn_id	Connection identifier of the call in question.
digits	A pointer to the digits that should be sent.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (standard-return-values.)

TSendEvent

Description

Sends the T-Library event specified by the parameter `event` to T-Server. This makes the data from the event available to the clients registered to receive events about the telephony object specified by the `AttributeThisDn` in `event`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
event	A pointer to the data structure that should be sent. T-Server does not check the contents of the data structure. T-Server distributes the data structure to client(s) registered on the telephony object specified by the `ThisDN` attribute of the event in question.

Return Values

Standard (See standard-return-values.)

Comments

T-Server is not concerned with the contents of the event.

TSendEventEx

Description

Sends the T-Library event specified by the parameter `event` to T-Server. This makes the data from the event available to the clients registered to receive events about the telephony object specified by the attribute `ThisDn` in the event. As distinct from the function `TSendEvent()`, this function also allows for a client specified by the parameter `client_id` to receive the event in question. (See below for details.)

Parameters

Name	Description
server	Local server handle to the T-Server in question.
event	A pointer to the data structure that should be sent. T-Server checks the contents of the data structure only to see if reference ID exists. When there exists a value for a reference ID, that attribute is passed only to the client specified by the parameter `client_id`. The remaining data structure is distributed both to that client and to the client(s) registered on the telephony object specified by the `ThisDN` attribute of the event in question.
client_id	Client identified for receipt of the data structure, including the reference ID, being sent.

Return Values

Standard (See standard-return-values.)

Comments

T-Server is not concerned with the contents of the user event.

TSendUserEvent

Description

Sends the application-defined data structure specified by the parameter `event` to T-Server, making this data available to the applications registered to receive events about the telephony object specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object the user event relates to.
event	A pointer to a data structure that should be sent. T-Library and T-Server do not distribute type events other than `EventUserEvent` for this T-Library request, even if the pointer indexes the structure with the different event type. T-Server does not check the contents of the data structure, but copies the data structure attributes into `EventUserEvent`, then distributes it to the client(s) registered on the telephony object specified by the dn attribute of the function request.

Return Values

Standard (See standard-return-values.)

Comments

T-Server is not concerned with the contents of the user event.

TSetCallAttributes

Description

Changes the call attributes.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
op	Type of the action that must be performed with attributes of the call specified by `conn_id`. Refer to the type TSetOpType.
conn_id	Connection identifier of the call whose attributes are requested to be changed.
new_conn_id	A new connection identifier that is requested to be assigned to the call in question.
origination	Directory number that identifies the calling party.
destination	Directory number that identifies the called party.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TSetDNDOff

Description

Sets the Do-Not-Disturb (DND) feature to `Off` for the telephony object specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object for which DND is requested to be turned off.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure that takes into account switch-specific features that cannot be described by the above parameters.

Return Values

Standard (See standard-return-values.)

TSetDNDOn

Description

Sets the Do-Not-Disturb (DND) feature to 0n for the telephony object specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object for which DND is requested to be turned on.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TSetInputMask

Description

Directs T-Server to send to an application a subset of `TEvent` specified by `mask` and filter out the rest. When a client application establishes a connection with T-Server, the complete list of events available from the media device is specified during the initialization procedure. This function narrows that list.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
mask	Specifies which events should be transmitted to the application.

Return Values

Standard (See standard-return-values.)

Comments

The corresponding bit in the TMask structure has to be set to `1` to allow an event to be sent to a client. To stop the delivery of an event, the bit must be set to `0`. The mask is set per connection.

TSetMessageWaitingOff

Description

Sets the Message Waiting indication to `Off` for the telephony object specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object for which message waiting is requested to be turned off.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TSetMessageWaitingOn

Description

Sets the Message Waiting indicator to 0n for the telephony object specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object for which message waiting is requested to be turned on.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TSetMuteOff

Description

On an existing conference call, cancels the Mute mode for the party specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object for which the Mute mode is requested to be terminated.
conn_id	Connection identifier of the conference call in which the party is participating.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TSetMuteOn

Description

On an existing conference call, sets Mute mode for the party specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that is requested to be set to Mute mode.
conn_id	Connection identifier of the conference call in which the party is participating.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TSetParamHA

Description

Sets the values of timeouts to be used in an HA configuration that is opened with Syncmode. In each case T-Library blocks certain operations according to the values of parameters set with `TSetParamHA()` until a specific notification is received.

The available parameters are:

- `connect_timeout`–a limit on the time spent trying to (re)connect to a remote server (not including the time used for DNS lookup).
- `switchover_timeout`–the time T-Library will wait before switching over to backup after losing a connection with primary T-Server.
- `reconnect_interval`–the interval between successive attempts to reconnect to backup T-Server.

Parameters

Name	Description
key	Timeout-related parameter to be set with `TSetParamHA()`. Possible values consist of `connect_timeout`, `switchover_timeout`, `reconnect_attempts` (`obsolete–ignored` by T-Library), and `reconnect_interval`.
value	A value for the parameter identified in key. The integer here should take into account the specific key used.

Return Values

Standard (See standard-return-values.)

TSetReferenceID

Description

Sets the reference ID value to be used by T-Library for the next T-Library request.

Parameters

Name	Description
reference_id_for_next_req	A new reference identifier to be assigned to the next T-Library call.

Return Values

`reference_id_for_next_req` — Returned to the application if the request has been actually processed by T-Library.

< `0` — Error condition. Returned to the application if the request has not been processed by T-Library due to incorrect function statement.

Important

The return will never be a zero (`0`) value.

Comments

This function does not generate any messages to T-Server.

TSetRefIDLimit

Description

Sets the maximum allowed for the reference ID value to be used by T-Library.

Parameters

Name	Description
new_limit	A new maximum value for the reference identifier to be assigned to T-Library calls.

Return Values

Standard (See standard-return-values.)

Important

The maximum permitted value for the reference ID is 0x7FFF.

Comments

This function does not generate any messages to T-Server.

TSetSocketChangeCallback

Description

Sets the callback function provided by a client. `Client_callback` is called by T-Library any time a new socket is created or an old one is deleted.

Parameters

Name	Description
client_callback	Pointer to client's callback function. The input parameter for the callback is a pointer to `TSockInfo` structure.

Return Values

None.

TSingleStepConference

Description

Adds a new party to an existing call and creates a conference.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that requests this function.
conn_id	Connection identifier of the call that is requested to be conferenced.
destination	Directory number of the party to which the call will be conferenced.
location	Name of the remote location in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
user_data	A pointer to the user data that should be attached to the call.
reasons	A pointer to a data structure which provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TSingleStepTransfer

Description

Transfers the call from a specified directory number dn that is currently engaged in the call specified by the parameter `conn_id` to a destination DN that is specified by the parameter `destination` .

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object that will be replaced.
conn_id	Connection identifier of the call that is requested to be transferred.
destination	Directory number of the party the call will be transferred to.
location	Name of the remote location in the form of `<SwitchName>` or `<T-ServerApplicationName>@<SwitchName>`.
user_data	A pointer to the user data that should be attached to the call.
reasons	A pointer to a data structure that provides additional information associated with this action.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TSockInfoStructure

Description

Description of `TSockInfoStructure`.

Parameters

Name	Description
socket	A socket number.
sync_id	ID associated with the socket and returned by `TOpenServer()` and `TOpenServerEx()` functions.
reason	A reason for callback function. Possible values are `TSocketCreated` or `TSocketDeleted`.
ptr	Reserved for future use.

TSyncIsSet

Description

Checks if there is activity on sockets associated with `sync_id`.

Parameters

Name	Description
sync_id	ID returned by `TOpenServer()` or `TOpenServerEx()` functions.
pfd	Pointer to `fd_set` structure used for the `TSyncSetSelectMask()` function.

Return Values

A non-zero value if data is present on one or more sockets associated with `sync_id`.

TSyncSetSelectMask

Description

Sets the `fd_set` structure used by the `TSyncSetSelectMask()` function with values of sockets associated with `sync_id`.

Parameters

Name	Description
sync_id	ID returned by `TOpenServer()` or `TOpenServerEx()` functions.
pfd	Pointer to `fd_set` structure used for the `TSyncSetSelectMask()` function.

Return Values

The largest socket number associated with `sync_id`. If there are no sockets, returns `0`.

TUnregisterAddress

Description

Cancels the application's registration to send feature requests and receive events regarding the telephony object specified by the parameter dn.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number that specifies the telephony object in question.
ctrmode	Control mode specifying whether the fact of registration canceling is to be relayed to the switch. Refer to the type TControlMode. Specifying this parameter makes sense only if the CTI link protocol of the switch in question supports the telephony object registration request.
extensions	A pointer to an additional data structure.

Return Values

Standard (See standard-return-values.)

TUpdateUserData

Description

On behalf of the telephony object specified by the parameter dn, updates the user data that is attached to the call specified by the parameter `conn_id` with the data specified by the parameter `user_data`.

Parameters

Name	Description
server	Local server handle to the T-Server in question.
dn	Directory number of the telephony object on whose behalf the information is to be updated.
conn_id	Connection identifier of the call in question.
user_data	A pointer to the user data that should be attached to the call.

Return Values

Standard (See standard-return-values.)

Comments

If the existing call has a `user_data` key-value pair with the same key as specified in the request, T-Server replaces that value with a new value, as shown in the following table.

Example of Update to User_Data

Existing user_data	user_data to Be Added	Result
data1=1234 data2=5678 data3=9012	data3=9999 data4=1212 data5=5555	data1=1234 data2=5678 data3=9999 data4=1212 data5=5555

TXCapsSupported

Description

Allows you to test the value returned by the function `TGetXCaps()`. This function tests if the enumerator specified by the parameter `enum_value` is supported by the T-Server in question.

Parameters

None

Return Values

`+1` — The capability in question is supported.

`0` — The capability in question is not supported.

`-1` — There is no information from the T-Server regarding this combination of request/event ID and attribute.

List of TLib Datatypes

The following table provides a convenient list of TLib Datatypes that are available.

Important

The names of most of these datatypes start with a "T", but the Voice Platform SDK uses names that do not contain an initial "T". For example, the TRegisterMode datatype mentioned in this section is known to the Voice Platform SDK as `RegisterMode`.

AddressStatusInfoType	TCallState	TKVResult	TRegisterMode
AssociationInfoType	TCallType	TKVType	TReliability
MsgWaitingInfoType	TClearFlag	TLocationInfoType	TRemoteParty
TAddressInfoStatus	TConnectionID	TMakeCallType	TRouteType
TAddressInfoType	TControlMode	TMediaType	TScanServerMode
TAddressType	TDNRole	TMergeType	TServer
TAgentID	TDirectoryNumber	TMessageType	TServerRole
TAgentPassword	TEvent	TMonitorNextCallType	TSetOpType
TAgentType	TEventMask	TNetworkCallState	TSwitchInfoType
TAgentWorkMode	TFile	TNetworkDestState	TTime
TAttribute	TForwardMode	TNetworkPartyRole	TTimeStamp
TCallHistoryInfo	TInterruptFlag	TOpenMode	TTreatmentType
TCallID	TKVList	TPartyState	TXCaps
TCallInfoType	TKVPair	TPrivateMsgType	TXRouteType

TKVListAddBinary

Description

Adds one data item of binary type (that is, a specified number of unsigned characters), specified as the parameters key and `value`, to the data structure specified by the parameter `list`.

Syntax

```
TKVResult TKVListAddBinary(
        TKVList *list,
        char *key,
        unsigned char *val,
        int length
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure to which the specified key-value pair should be added.
key	A pointer to the string representing the parameter name.
val	A pointer to the value of the parameter specified by the parameter `key`.
length	The length of the parameter specified by the parameter `val`.

Return Values

Refer to the type TKVResult.

TKVListAddInt

Description

Adds one data item of integer type, specified as the parameters key and `value`, to the data structure specified by the parameter `list`.

Syntax

```
TKVResult TKVListAddInt(
        TKVList *list,
        char *key,
        int value
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure to which the specified key-value pair should be added.
key	A pointer to the string representing the parameter name.
value	The value of the key-value pair specified by the parameter key.

Return Values

Refer to the type TKVResult.

Comments

See also TKVListAddString, TKVListAddList, and TKVListAddBinary.

TKVListAddList

Description

Adds one data item of `TKVList` type, specified as the parameters `key` and `value`, to the data structure specified by the parameter `list`.

Syntax

```
TKVResult TKVListAddList(
        TKVList *list,
        char *key,
        TKVList *value
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure to which the specified key-value pair should be added.
key	A pointer to the string representing the parameter name.
value	A pointer to the value of the parameter specified by the parameter key.

Return Values

Refer to the type TKVResult.

Comments

See also TKVListAddInt, TKVListAddString, and TKVListAddBinary.

TKVListAddString

Description

Adds one data item of character pointer (that is, string) type, specified as the parameters key and value, to the data structure specified by the parameter `list`.

Syntax

```
TKVResult TKVListAddString(
        TKVList *list,
        char *key,
        char *value
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure to which the specified key-value pair should be added.
key	A pointer to the string representing the parameter name.
value	A pointer to the value of the parameter specified by the parameter key.

Return Values

Refer to the type TKVResult.

Comments

See also TKVListAddInt, TKVListAddBinary, and TKVListAddList.

TKVListAddUnicode

Description

Adds one data item of a unicode type, specified as the parameters key and `value`, to the data structure specified by the parameter `list`.

Syntax

```
TKVResult TKVListAddUnicode(
        TKVList *list,
        const char *key,
        const wchar_t *value,
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure to which the specified key-value pair should be added.
key	A pointer to the string representing the parameter name.
value	A pointer to the value of the parameter specified by the parameter key.

Return Values

Refer to the type TKVResult.

TKVListBinaryLength

Description

Returns the length of a binary type value specified by `pair`.

Syntax

```
int *TKVListBinaryLength(
        TKVPair *pair
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
pair	A pointer to the key-value pair that contains the value in question.

Return Values

None

Comments

See also TKVListGetIntValue, TKVListGetListValue, TKVListGetStringValue, TKVListGetUnicodeValue, and TKVListGetBinaryValue().

TKVListBinaryValue

Description

Returns the value (of binary type) of the key-value pair specified by the parameter `pair`.

Syntax

```
unsigned char TKVListBinaryValue(
        TKVPair *pair
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
pair	A pointer to the key-value pair that contains the value in question.

Return Values

The value of the parameter contained in a key-value pair specified by the parameter `pair`.

Comments

See also TKVListType, TKVListIntValue, TKVListUnicodeValue, TKVListStringValue, and TKVListListValue.

TKVListCreate

Description

Creates an empty data structure.

Syntax

```
TKVList *TKVListCreate();
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Return Values

Returns a pointer to the created data structure.

Comments

See also TKVListFree.

TKVListDeleteAll

Description

Deletes all data from the data structure specified by the parameter `list`.

Syntax

```
TKVResult TKVListDeleteAll(
        TKVList *list
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure from which all data should be deleted.

Return Values

Refer to the type TKVResult.

Comments

See also TKVListDeletePair.

TKVListDeletePair

Description

Deletes one data item, specified by the parameter key, from the data structure specified by the parameter `list`.

Syntax

```
TKVResult TKVListDeletePair(
        TKVList *list,
        char *key
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure from which the specified key-value pair should be deleted.
key	A pointer to the key of the key-value pair that should be deleted.

Return Values

Refer to the type TKVResult.

Comments

See also TKVListDeleteAll.

TKVListDup

Description

Duplicates the data structure specified by `list`.

Syntax

```
TKVList TKVListDup(
        TKVList *list
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure that is requested to be duplicated.

Return Values

Returns a pointer to the duplicated data structure.

Comments

See also TKVListCreate and TKVListFree.

TKVListFree

Description

Deletes the data structure specified by the parameter `list`.

Syntax

```
TKVResult TKVListFree(
        TKVList *list
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure that is requested to be deleted.

Return Values

Refer to the type TKVResult.

Comments

See also TKVListCreate.

TKVListGetBinaryValue

Description

Returns a pointer to the value (of a binary type) of the key value specified by the parameter key contained in the data structure specified by the parameter `list`.

Syntax

```
unsigned char *TKVListGetBinaryValue(
        TKVList *list,
        char *key,
        TKVResult *result
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure that contains the key in question.
key	A pointer to the string representing the parameter name.
result	A pointer to the memory cell that will contain the result of the requested procedure. Refer to the type TKVResult.

Return Values

A pointer to the value of the key-value pair specified by the parameter key contained in the data structure specified by the parameter `list`, if the parameter is found otherwise unspecified.

Comments

See also TKVListGetIntValue, TKVListGetListValue, TKVListGetUnicodeValue, and TKVListGetStringValue.

TKVListGetIntValue

Description

Returns a value (of an integer type) of the key-value pair specified by the parameter key contained in the data structure specified by the parameter `list`.

Syntax

```
int TKVListGetIntValue(
        TKVList *list,
        char *key,
        TKVResult *result
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure that contains the key in question.
key	A pointer to the string representing the parameter name.
result	A pointer to the memory cell that will contain the result of the requested procedure. (Refer to the type TKVResult).

Return Values

A pointer to the value of the key-value pair specified by the parameter key contained in the data structure specified by the parameter `list`, if the parameter is found otherwise unspecified.

Comments

See also TKVListGetListValue, TKVListGetBinaryValue, TKVListGetUnicodeValue, and

TKVListGetStringValue.

TKVListGetListValue

Description

Returns a pointer to the value (of TKV`List` type) of the key-value pair specified by the parameter key contained in the data structure specified by the parameter `list`.

Syntax

```
TKVList *TKVListGetListValue(
        TKVList *list,
        char *key,
        TKVResult *result
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure that contains the key in question.
key	A pointer to the string representing the parameter name.
result	A pointer to the memory cell that will contain the result of the requested procedure. (Refer to the type TKVResult).

Return Values

A pointer to the value of the key-value pair specified by the parameter key contained in the data structure specified by the parameter `list`, if the parameter is found otherwise unspecified.

Comments

See also TKVListGetIntValue, TKVListGetStringValue, TKVListGetUnicodeValue, and TKVListGetBinaryValue.

TKVListGetStringValue

Description

Returns a pointer to the value (of a character pointer type) of the key-value pair specified by the parameter key contained in the data structure specified by the parameter `list`.

Syntax

```
char *TKVListGetStringValue(
        TKVList *list,
        char *key,
        TKVResult *result
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure that contains the parameter in question.
key	A pointer to the string representing the parameter name.
result	A pointer to the memory cell that will contain the result of the requested procedure. (Refer to the type TKVResult).

Return Values

A pointer to the value of the key-value pair specified by the parameter key contained in the data structure specified by the parameter `list`, if the parameter is found otherwise unspecified.

Comments

See also TKVListGetListValue, TKVListGetUnicodeValue, and TKVListGetBinaryValue.

TKVListGetUnicodeValue

Description

Returns a pointer to the value (of a unicode type) of the key value specified by the parameter key contained in the data structure specified by the parameter `list`.

Syntax

```
wchar_t *TKVListGetUnicodeValue(
        TKVList *list,
        const char *key,
        TKVResult *result
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure that contains the key in question.
key	A pointer to the string representing the parameter name.
result	A pointer to the memory cell that will contain the result of the requested procedure. Refer to the type TKVResult).

Return Values

A pointer to the value of the key-value pair specified by the parameter key contained in the data structure specified by the parameter `list`, if the parameter is found otherwise unspecified.

Comments

See also TKVListGetIntValue, TKVListGetListValue, TKVListGetBinaryValue, and TKVListGetStringValue.

TKVListInitScanLoop

Description

Initialize internal pointer to first key-value pair of data structure specified by the parameter `list`, for subsequent use by `TKVListNextPair()`.

Syntax

```
TKVResult TKVListInitScanLoop(
        TKVList *list
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure for which the data search loop should be initiated.

Return Values

Refer to the type TKVResult.

Comments

See also TKVListNextPair.

TKVListIntValue

Description

Returns the value (of integer type) of the parameter contained in a key-value pair specified by the parameter `pair`.

Syntax

```
int TKVListIntValue(
        TKVPair *pair
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
pair	A pointer to the key-value pair that contains the value in question.

Return Values

The value of the integer contained in a key-value pair specified by the parameter `pair`.

Comments

See also TKVListType, TKVListStringValue, TKVListListValue, TKVListUnicodeValue, and TKVListBinaryValue.

TKVListKey

Description

Returns a pointer to the key contained in a key-value pair specified by the parameter pair.

Syntax

```
char *TKVListKey(
        TKVPair *pair
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
pair	A pointer to the key-value pair that contains the key in question.

Return Values

A pointer to the key contained in a key-value pair specified by the parameter `pair`.

Comments

See also TKVListType, TKVListIntValue, TKVListStringValue, TKVListUnicodeValue, TKVListBinaryValue, and TKVListListValue.

TKVListListValue

Description

Returns a pointer to the value (of `TKVList` type) of a key-value pair specified by the parameter `pair`.

Syntax

```
TKVList *TKVListListValue(
        TKVPair *pair
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
pair	A pointer to the key-value pair that contains the value in question.

Return Values

A pointer to the value of the parameter contained in a key-value pair specified by the parameter `pair`.

Comments

See also TKVListType, TKVListIntValue, TKVListStringValue, TKVListUnicodeValue, and TKVListBinaryValue.

TKVListNextPair

Description

Returns a pointer to the next key-value pair of the data structure specified by the parameter `list`.

Syntax

```
TKVPair *TKVListNextPair(
        TKVList *list
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
list	A pointer to the data structure that contains the key-value pair in question.

Return Values

A pointer to the next key-value pair of the data structure specified by the parameter `list`. (When the end of the list in question is reached, the function will return `NULL`.) Refer to the type TKVPair.

Comments

See also TKVListInitScanLoop.

TKVListPrint

Description

Sends the data contained in a data structure specified by the parameter `list` to the file specified by the parameter `f`.

Syntax

```
void TKVListPrint(
        void *f,
        TKVList *list,
        char *header
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
f	A pointer to the file where the data is requested to be printed.
list	A pointer to the data structure that is requested to be printed.
header	A pointer to the string containing additional text to be printed (date and time, source ID, comments, and so on). Set to `0` if no additional text should be printed.

Return Values

None

TKVListStringValue

Description

Returns a pointer to the value (of character pointer type) of the parameter contained in a key-value pair specified by the parameter `pair`.

Syntax

```
char *TKVListStringValue(
        TKVPair *pair
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
pair	A pointer to the key-value pair that contains the value in question.

Return Values

A pointer to the value of the parameter contained in a key-value pair specified by the parameter `pair`.

Comments

See also TKVListType, TKVListIntValue, TKVListBinaryValue, TKVListUnicodeValue, and TKVListListValue.

TKVListType

Description

Returns the type of variable contained in a key-value pair specified by the parameter `pair`.

Syntax

```
TKVType TKVListType(
        TKVPair *pair
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
pair	A pointer to the key-value pair that contains the variable in question.

Return Values

Refer to the type TKVType.

Comments

See also TKVListIntValue, TKVListStringValue, TKVLIstUnicodeValue, TKVListBinaryValue, and TKVListListValue.

TKVListUnicodeValue

Description

Returns the value (of unicode type) of the key-value pair specified by the parameter `pair`.

Syntax

```
wchar_t *TKVListUnicodeValue(
        TKVPair *pair
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
pair	A pointer to the key-value pair that contains the value in question.

Return Values

The value of the parameter contained in a key-value pair specified by the parameter `pair`.

Comments

See also TKVListType, TKVListIntValue, TKVListStringValue, TKVListBinaryValue, and TKVListListValue.

TKVListGetPair

Description

Returns a pointer to the pair of the parameter specified by key contained in the data structure specified by `kvl`.

Syntax

```
TKVPair *TKVListGetPair(
        TKVList *kvl,
        char *key,
        TKVResult *result
        );
```

Parameters

For basic type information for each parameter, consult the Syntax section, above.

Name	Description
kvl	A pointer to the data structure that contains the pair in question.
key	A pointer to the string representing the parameter name.
result	A pointer to the memory cell that will contain the result of the requested procedure. (Refer to the type TKVResult).

Return Values

A pointer to the value of the key-value pair specified by the parameter key contained in the data structure specified by the parameter `list`, if the parameter is found otherwise unspecified.

Comments

See also TKVListGetIntValue, TKVListGetListValue, TKVListGetUnicodeValue, and

TKVListGetStringValue.

List of TLib Unstructured Data

The following table provides a convenient list of TLib Unstructured Data functions that are available.

These functions deal exclusively with transaction-related user data on the client side and allow you to work with all three categories of unstructured data: *User Data*, *Extensions*, and *Reasons*. None of these functions generate any requests to T-Server. The result of the function execution is confirmed by the value that the function returns.

TKVListAddBinary	TKVListCreate	TKVListGetListValue	TKVListListValue
TKVListAddInt	TKVListDeleteAll	TKVListGetPair	TKVListNextPair
TKVListAddList	TKVListDeletePair	TKVListGetStringValue	TKVListPrint
TKVListAddString	TKVListDup	TKVListGetUnicodeValue	TKVListStringValue
TKVListAddUnicode	TKVListFree	TKVListInitScanLoop	TKVListType
TKVListBinaryLength	TKVListGetBinaryValue	TKVListIntValue	TKVListUnicodeValue
TKVListBinaryValue	TKVListGetIntValue	TKVListKey	

AddressStatusInfoType

Syntax

```
typedef enum {
        AddressStatusIdle,
        AddressStatusOrigination,
        AddressStatusDialing,
        AddressStatusTalking,
        AddressStatusRinging,
        AddressStatusHeld,
        AddressStatusTreatment,
        AddressStatusLockedOut,
        AddressStatusMaintenance,
        AddressStatusAvailable,
        AddressStatusVacant
} AddressStatusInfoType;
```

Values

- `AddressStatusIdle` — A state in which there is no relationship between the call and object specified by `ThisDN`.
- `AddressStatusOrigination` — A state in which an object specified by `ThisDN` is requesting a service, or is in the process of dialing the necessary digit sequence to initiate a call to another object. The object enters this state when the device goes off-hook, or the device is prompted to go off-hook as a result of some service that is initiated for the device. The connection leaves the `origination` state when the dialing sequence is completed, aborted, or has failed.
- `AddressStatusDialing` — A state in which an object specified by `ThisDN` completed a dialing sequence. `AddressStatusDialing` has the same meaning as `AddressStatusRinging` but for an object originating the call.
- `AddressStatusTalking` — A state in which an object specified by `ThisDN` is actively participating in a call. This state includes logical participation in a call as well as physical participation (for example, a connected device cannot be on hold).
- `AddressStatusRinging` — A state in which the attempt is being made to connect an object specified by `ThisDN` to a call and it is now up to the device to take some action.
- `AddressStatusHeld` — A state in which an object specified by `ThisDN` is inactively participating in a call. This state includes logical participation in a call while physical participation is suspended.
- `AddressStatusTreatment` — A state in which a telephony object specified by `ThisDN` actively participating in a call is under treatment at the time of the request.
- `AddressStatusLockedOut` — The switch puts an object specified by `ThisDN` in this state when it encountered an interdigit timeout while dialing.
- `AddressStatusMaintenance` — A state in which the telephony object specified by `ThisDN` has been taken out of service and can no longer accept calls or service requests.

- `AddressStatusAvailable` — A state in which the telephony object specified by `ThisDN` is busy with a call, but still available to receive another call.
- `AddressStatusVacant` — The status of the telephony object in question is `Vacant`.

AssociationInfoType

Syntax

```
typedef enum {
        AssociatedToThisSession,
        AssociatedToAnotherSession,
        NotAssociated,
        AssociationNotAllowed
} AssociationInfoType;
```

Values

- `AssociatedToThisSession` — Association type: associated with this session.
- `AssociatedToAnotherSession` — Association type: associated with another session.
- `NotAssociated` — Association type: not associated.
- `AssociationNotAllowed` — Association type: not allowed.

MsgWaitingInfoType

Syntax

```
typedef enum {
        MsgWaitngActivated,
        MsgWaitingDeactivated,
        ExecMsgWaitingActivated,
        ExecMsgWaitingDeactivated,
        MsgWaitingNotAllowed
} MsgWaitingInfoType;
```

Values

- `MsgWaitingActivated` — Message Waiting status is activated.
- `MsgWaitingDeactivated` — Message Waiting status is deactivated.
- `ExecMsgWaitingActivated` — Executive Message Waiting status is activated.
- `ExecMsgWaitingDeactivated` — Executive Message Waiting status is deactivated.
- `MsgWaitingNotAllowed` — Message Waiting and Executive Message Waiting are not allowed.

TAddressInfoStatus

Syntax

```
typedef union {
        AssociationInfoType                 AssociationStatus;
        AddressStatusInfoType                 AddressStatus;
        MsgWaitingInfoType                         MsgWaitingStatus;
        TForwardMode                         CallForwardingStatus;
        TAgentWorkMode                         AgentStatus;
        int                                 NumberOfAgentsInQueue;
        int                                 NumberOfAvailableAgentsInQueue;
        int                                 NumberOfCallsInQueue;
        TAddressType                         AddressType
        TForwardMode                         SendAllCallsStatus;
        int                                 NumberOfIdleClassifiers;
        int                                 NumberOfClassifiersInUse;
        int                                 NumberOfIdleTrunks;
        int                                 NumberOfTrunksInUse;
        int                                 NumberOfListElements;
} TAddressInfoStatus;
```

Attributes

- `AssociationStatus` — See AssociationInfoType.
- `AddressStatus` — See AddressStatusInfoType.
- `MsgWaitingStatus` — See MsgWaitingInfoType.
- `CallForwardingStatus` — See TForwardMode.
- `AgentStatus` — See TAgentWorkMode.
- `NumberOfAgentsInQueue` — Specifies the number of agents currently logged in a queue.
- `NumberOfAvailableAgentsInQueue` — Specifies the number of agents in a queue currently ready to receive a call.
- `NumberOfCallsInQueue` — Specifies the number of calls in a queue.
- `AddressType` — See TAddressType.
- `SendAllCallsStatus` — See TForwardMode.
- `NumberOfIdleClassifiers` — Specifies the number of idle classifiers.
- `NumberOfClassifiersInUse` — Specifies the number of classifiers in use.
- `NumberOfIdleTrunks` — Specifies the number of idle trunks.
- `NumberOfTrunksInUse` — Specifies the number of trunks in use.
- `NumberOfListElements` — Specifies the number of list elements. The value is passed in the TEvent

attribute `Extensions`.

TAddressInfoType

Syntax

```
typedef enum {
        AddressInfoAddressStatus,
        AdderssInfoMsgWaitingStatus,
        AddressInfoAssociationStatus,
        AddressInfoCallForwardingStatus,
        AddressInfoAgentStatus,
        AddressInfoNumberOfAgentsInQueue,
        AddressInfoNumberOfAvailableAgentsInQueue,
        AddressInfoNumberOfCallsInQueue,
        AddressInfoAddressType,
        AddressInfoCallsQuery,
        AddressInfoSendAllCallsStatus,
        AddressInfoQueueLoginAudit,
        AddressInfoNumberOfIdleClassifiers,
        AddressInfoNumberOfClassifiersInUse,
        AddressInfoNumberOfIdleTrunks,
        AddressInfoNumberOfTrunksInUse,
        AddressInfoDatabaseValue,
        AddressInfoDNStatus
        AddressInfoQueueStatus
} TAddressInfoType;
```

Values

- `AddressInfoAddressStatus` — Information type: address status.
- `AdderssInfoMsgWaitingStatus` — Information type: message waiting status.
- `AddressInfoAssociationStatus` — Information type: association status.
- `AddressInfoCallForwardingStatus` — Information type: call forwarding status.
- `AddressInfoAgentStatus` — Information type: status of the current agent.
- `AddressInfoNumberOfAgentsInQueue` — Information type: number of agents currently logged in to a queue.
- `AddressInfoNumberOfAvailableAgentsInQueue` — Information type: number of agents in a queue currently ready to receive a call.
- `AddressInfoNumberOfCallsInQueue` — Information type: number of calls in a queue.
- `AddressInfoAddressType` — Information type: type of object in question.
- `AddressInfoCallsQuery` — Information type: calls query.
- `AddressInfoSendAllCallsStatus` — Information type: send all calls status.
- `AddressInfoQueueLoginAudit` — Information type: queue login audit.

- `AddressInfoNumberOfIdleClassifiers` — Information type: number of idle classifiers.
- `AddressInfoNumberOfClassifiersInUse` — Information type: number of classifiers in use.
- `AddressInfoNumberOfIdleTrunks` — Information type: number of idle trunks.
- `AddressInfoNumberOfTrunksInUse` — Information type: number of trunks in use.
- `AddressInfoDatabaseValue` — Information type: database value.
- `AddressInfoDNStatus` — Information type: DN status.
- `AddressInfoQueueStatus` — Information type: queue status.

TAddressType

Syntax

```
typedef enum {
        AddressTypeUnknown,
        AddressTypeDN,
        AddressTypePosition,
        AddressTypeQueue,
        AddressTypeRouteDN,
        AddressTypeTrunk,
        AddressTypeVoiceChannel,
        AddressTypeDataChannel,
        AddressTypeAnnouncement,
        AddressTypeASAI;
        AddressTypeACDGroup,
        AddressTypeVSP,
        AddressTypeRouteQueue,
        AddressTypeAgentID,
        AddressTypeOther
} TAddressType;
```

Values

- `AddressTypeUnknown` — The type of the telephony object is unknown.
- `AddressTypeDN` — An extension line that does not belong to an ACD group.
- `AddressTypePosition` — An extension line that belongs to an ACD group.
- `AddressTypeQueue` — An ACD group.
- `AddressTypeRouteDN` — A logical point inside the switch where the call resides while waiting to be routed.
- `AddressTypeTrunk` — A trunk.
- `AddressTypeVoiceChannel` — A voice-mail channel.
- `AddressTypeDataChannel` — A data channel.
- `AddressTypeAnnouncement` — An announcement-machine channel.
- `AddressTypeASAI` — A resource on a switch that uses the `ASAI` protocol.
- `AddressTypeACDGroup` — An ACD group used for domain control of splits. (This value is specific to the Avaya Communication Manager.)
- `AddressTypeVSP` — Telephony object is a Virtual Soft Phone.
- `AddressTypeRouteQueue` — Type of the telephony object is a `RouteQueue`. The behavior of this object includes routing point and queue functionality at the same time.
- `AddressTypeAgentID` — Type of the object is `AgentID`.

- `AddressTypeOther` — Type of the telephony object is `Other`. This type is used when an object cannot be specified by any of the above described types.

Comments

When configuring DNs, you must set DN types in the Configuration Layer. These configured DN types correspond to `TAddressType` values, as shown in the following table.

CME DN Type and TAddressType Correspondance

DN Type in CME	TAddressType
CFGExtension	AddressTypeDN
CFGEAPort	AddressTypeDN
CFGACDPosition	AddressTypePosition
CFGACDQueue	AddressTypeQueue
CFGRoutingPoint	AddressTypeRouteDN
CFGExtRoutingPoint	AddressTypeRouteDN
CFGServiceNumber	AddressTypeRouteDN
CFGTrunk	AddressTypeTrunk
CFGTieLine	AddressTypeTrunk
CFGVoiceMail	AddressTypeVoiceChannel
CFGData	AddressTypeDataChannel
CFGMusic	AddressTypeAnnouncement
CFGRoutingQueue	AddressTypeRouteQueue

Important

For all other Configuration Server DN types there are no corresponding `AddressTypes` in T-Library. Such DNs must be registered on T-Server with their `ControlMode` set to `RegisterLocal`.

TAgentID

Syntax

```
typedef char *TAgentID;
```

TAgentPassword

Syntax

```
typedef char *TAgentPassword;
```

TAgentType

Syntax

```
typedef enum {
        AgentTypeAgent,
        AgentTypeSupervisor
} TAgentType;
```

Values

- `AgentTypeAgent` — This is a regular agent.
- `AgentTypeSupervisor` — This is a supervisor.

TAgentWorkMode

Syntax

```
typedef enum {
        AgentWorkModeUnknown        = 0,
        AgentManualIn                = 1,
        AgentAutoIn                = 2,
        AgentLegalGuard                = AgentAutoIn,
        AgentAfterCallWork                = 3,
        AgentAuxWork                = 4,
        AgentNoCallDisconnect        = 5,
        AgentWalkAway                = 6,
        AgentReturnBack                = 7
} TAgentWorkMode;
```

Values

- `AgentWorkModeUnknown` — The agent work mode is unknown or not specified.

Ready States

- `AgentManualIn` — The agent has to perform a manual operation to become available.
- `AgentAutoIn` — The switch's control system decides agent availability.
- `AgentReturnBack` — The agent has indicated his return to the agent workstation.

Not Ready States

- `AgentAfterCallWork` — If the agent work mode is set to `AgentManualIn`, this status specifies the condition assigned to the agent after the previous call has cleared and before the agent becomes available to receive the next call.
- `AgentAuxWork` — The agent is not ready to receive calls (specific to the Avaya Communication Manager).
- `AgentLegalGuard` — The switch's control system decides agent availability. (This is equivalent to `AgentAutoIn`, but as a sub state of `Not Ready`.)
- `AgentNoCallDisconnect` — The agent work mode is set to `NoCallDisconnect` rather than the default value. (See specifics of `NoCallDisconnect IE` in Meridian Link Release 5 Interface Specification).
- `AgentWalkAway` — The agent has solicited a request not to be available for receiving calls, and is away from the agent station.

TAttribute

Syntax

```
enum TAttribute {
        AttributeProtocolVersion,
        AttributeErrorMessage,
        AttributeReferenceID,
        AttributeCallID,
        AttributeNodeID,
        AttributeNetworkCallID,
        AttributeNetworkNodeID,
        AttributeTransferredNetworkCallID,
        AttributeTransferredNetworkNodeID,
        AttributeConnID,
        AttributePreviousConnID,
        AttributeTransferConnID,
        AttributeConferenceConnID,
        AttributeCallState,
        AttributeAgentID,
        AttributeAgentStateReasonUnused,
        AttributeAgentWorkMode,
        AttributeReason,
        AttributeDNIS,
        AttributeANI,
        AttributeThisDN,
        AttributeThisQueue,
        AttributeThisTrunk,
        AttributeThisDNRole,
        AttributeOtherDN,
        AttributeOtherQueue,
        AttributeOtherTrunk,
        AttributeOtherDNRole,
        AttributeThirdPartyDN,
        AttributeThirdPartyQueue,
        AttributeThirdPartyTrunk,
        AttributeThirdPartyDNRole,
        AttributeMusicPath,
        AttributeVoiceFileName,
        AttributeFileHandle,
        AttributeUserID,
        AttributePassword,
        AttributeVoiceSegment,
        AttributeVoiceIFlag,
        AttributeVoiceCFlag,
        AttributeNumberOfDigits,
        AttributeTerminateDigits,
        AttributeTimeout,
        AttributeCollectedDigits,
        AttributeLastDigit,
        AttributeInputMask,
        AttributeErrorCode,
        AttributeAddressType,
        AttributeAddressMode,
        AttributeRegisterMode,
        AttributeControlMode,
        AttributeUserData,
        AttributeDataKey,
        AttributeExtensions,
        AttributeCallType,
        AttributeCallingLineName,
        AttributeAddressInfoType,
        AttributeAddressInfoStatus,
        AttributeUserEvent,
        AttributeCommunicationDN,
        AttributeRouteType,
        AttributeServerVersion,
        AttributeServerRole,
        AttributeServerCapabilityMask,
        AttributeApplicationName,
        AttributeApplicationPassword,
        AttributeMergeType,
        AttributeForwardMode,
        AttributeDTMFDigits,
        AttributeLocation,

        AttributeRemoteLocation,
        AttributeRemoteConnID,
        AttributeRemoteDN,

        AttributeMakeCallType,
        AttributeCallInfoType,

        AttributeHomeLocation,
        AttributeCustomerID,

        AttributeFirstTransferHomeLocation,
        AttributeFirstTransferOrigDN,
        AttributeLastTransferHomeLocation,
        AttributeLastTransferOrigDN,

        AttributeSetOperation,
        AttributeNewConnID,

        AttributeTreatmentType,
        AttributeTreatmentParms,

        AttributeCLID,

        AttributeTimeinSecs,
        AttributeTimeinuSecs,

        AttributeSwitchInfoType,

        AttributeXRouteType,
        AttributeAccessNumber,
        AttributeXReferenceID,

        AttributePlace,
        AttributePriority,

        AttributeAgentIdleReason,

        AttributeRegistrationCode,
        AttributeDNSyncInfo,

        AttributeMediaType, /* TMediaType enumeration */

        AttributeLocationInfoType,
        AttributeOriginalConnID,
        AttributeConsultType,
        AttributeMonitorNextCallType,

        AttributeClientID,
        AttributeProtocolID,
        AttributePreviousCallID,
        AttributePrivateMsgID,

        AttributeFirstTransferConnID,
        AttributeLastTransferConnID,

        AttributeSessionID,
        AttributeHost,
        AttributePort,

        AttributeDBID,
        AttributeOtherDBID,
        AttributeThirdPartyDBID,

        AttributeCause,
        AttributeDN,
        AttributeDialedNumber,
        AttributeOriginConnID,
        AttributeRefConnID,
        AttributeReliability,
        AttributeTimeStamp,
        AttributeSwitchTime,

        AttributePartyID,
        AttributePartyType,
        AttributePartyState,
        AttributeRefParty,
        AttributeCtrlParty,

        AttributeRemoteLoc,
        AttributeRemoteParty,
        AttributeDevice,

        AttributeReserved_1,
        AttributeReserved_2,
        AttributeUpdateRevision,

        AttributeReturnReceipt,
        AttributeReturnReceiptDN,

        AttributeServerXCapabilities,

        AttributeNetworkCallState,
        AttributeNetworkDestState,
        AttributeNetworkPartyRole,
        AttributeNetworkOrigDN,
        AttributeNetworkDestDN,

        AttributeIDMAX
};
```

TCallHistoryInfo

Syntax

```
typedef struct TCallHistoryInfo_tag {
        TRemoteParty        FirstTransfer;
        TRemoteParty        LastTransfer;
} TCallHistoryInfo;
```

Values

- `FirstTransfer` — First remote party has initiated a transfer.
- `LastTransfer` — Last remote party has initiated a transfer.

TCallID

Syntax

```
typedef unsigned long TCallID;
```

TCallInfoType

Syntax

```
typedef enum {
        CallInfoPartiesQuery,
        CallInfoStatusQuery
} TCallInfoType;
```

Values

- `CallInfoPartiesQuery` — Specifies the type of information (call parties) that is requested.
- `CallInfoStatusQuery` — Specifies the type of information (call parties) that is provided by T-Server, based on both the switch info and the internal T-Server info collected during a T-Server operation.

TCallState

Syntax

```
typedef enum {
        CallStateOk,
        CallStateTransferred,
        CallStateConferenced,
        CallStateGeneralError,
        CallStateSystemError,
        CallStateRemoteRelease,
        CallStateBusy,
        CallStateNoAnswer,
        CallStateSitDetected,
        CallStateAnsweringMachineDetected,
        CallStateAllTrunksBusy,
        CallStateSitInvalidnum,
        CallStateSitVacant,
        CallStateSitIntercept,
        CallStateSitUnknown,
        CallStateSitNocircuit,
        CallStateSitReorder,
        CallStateFaxDetected,
        CallStateQueueFull,
        CallStateCleared,
        CallStateOverflowed,
        CallStateAbandoned,
        CallStateRedirected,
        CallStateForwarded,
        CallStateConsult,
        CallStatePickedup,
        CallStateDropped,
        CallStateDroppednoanswer,
        CallStateUnknown,
        CallStateCovered,
        CallStateConverseOn,
        CallStateBridged,
        CallStateDeafened
        CallStateHeld
} TCallState;
```

Values

Tip

For information about call models that may be associated with these values, refer to *Genesys 7 Events and Models Reference Manual*.

- `CallStateOk` — Any call status apart from those specified below.
- `CallStateTransferred` — This is a transferred call.
- `CallStateConferenced` — This is a multi-party call.
- `CallStateGeneralError` — The call failed because of a general error.
- `CallStateSystemError` — The call failed because of a system error.
- `CallStateRemoteRelease` — The call is released from another telephony object.
- `CallStateBusy` — The call is receiving the busy tone.
- `CallStateNoAnswer` — The call has not been answered in a specified time interval.
- `CallStateSitDetected` — The call is receiving a special information tone.
- `CallStateAnswerMachineDetected` — The call is receiving an answering machine greeting.
- `CallStateAllTrunksBusy` — The call is receiving the network congestion tone.
- `CallStateSitInvalidnum` — The call is receiving the invalid number tone.
- `CallStateSitVacant` — The call is receiving the vacant tone.
- `CallStateSitIntercept` — The call is receiving the intercept tone.
- `CallStateSitUnknown` — The call is receiving an unknown special information tone.
- `CallStateSitNocircuit` — The call is receiving the no circuit tone.
- `CallStateSitReorder` — The call is receiving the reorder tone.
- `CallStateFaxDetected` — The call is receiving a fax/modem answer signal.
- `CallStateQueueFull` — The call has been released because the call queue in question was full.
- `CallStateCleared` — One party in a two-party call has disconnected after the call was answered.
- `CallStateOverflowed` — The call has been overflowed away from the party in question by an ACD Overflow feature.
- `CallStateAbandoned` — Caller in a two-party call has disconnected before the call was answered, for example, while in an incoming call queue.
- `CallStateRedirected` — The call has been redirected away from the party in question by another application.
- `CallStateForwarded` — The call has been forwarded to another telephony object.
- `CallStateConsult` — This call is a consultation call with respect to another call.
- `CallStatePickedup` — The call has been picked up by another telephony object after it was offered to the telephony object in question.
- `CallStateDropped` — The call has been dropped or released from an established three-way call. Or the switch has dropped the call in a predictive call scenario.
- `CallStateDroppednoanswer` — The call has been dropped or released from an established three-way call before being answered.
- `CallStateUnknown` — The call state is unknown.
- `CallStateCovered` — The call has been covered by another telephony object after it was offered to the telephony object in question. The CoveragePass functionality has to be activated on the switch for the DN where the call appears.

- `CallStateConverseOn` — The call has been delivered to a treatment device (IVR). This call state is used when a call is being placed into several queues.
- `CallStateBridged` — Third party (supervisor) has been connected to a call for monitoring purposes (Service Observing). The Service-Observing functionality has to be activated on the switch for the DN or Queue where the call appears.
- `CallStateDeafened` — The party cannot listen to the conversation, but can be heard by the conference members.
- `CallStateHeld` — The party cannot hear or be heard by the conference members.

TCallType

Syntax

```
typedef enum {
        CallTypeUnknown,
        CallTypeInternal,
        CallTypeInbound,
        CallTypeOutbound,
        CallTypeConsult
} TCallType;
```

Values

- `CallTypeUnknown` — The type of this call is unknown.
- `CallTypeInternal` — This is an internal call (between two DNs on the local switch).
- `CallTypeInbound` — This is an inbound call (from an off-PBX number).
- `CallTypeOutbound` — This is an outbound call (to an off-PBX number).
- `CallTypeConsult` — This is a consultation call (initiated on the local switch), regardless of what DNs participate in the call.

Comments

`CallType` is determined with respect to the local switch only. In multi-site environments the `CallType` of a call may be different for each of its different legs. For example, one T-Server may report a call as an `Outbound` or `Consult` call, but on the receiving end this call may be reported as `Inbound`.

TClearFlag

Syntax

```
typedef enum {
        Clear = 1,
        NotClear = 2
} TClearFlag;
```

Values

- `Clear` — Information that has been input before beginning the playback of a voice message will be deleted.
- `NotClear` — Information that has been input before beginning the playback of a voice message will not be deleted.

TConnectionID

Comments

This variable is platform dependent.

The following is a list of valid operations for `TConnectionID`:

- Creating variables of the type `TConnectionID`.
- Copying variables of this type.
- Comparing the variables of this type.
- Converting the variables of this type to and from strings, as follows:
  - `connid_to_str` — `TConnectionID` to a hexadecimal string
  - `connid_to_decimal` — `TConnectionID` to a decimal string
  - `str_to_connid` — a hexadecimal string to `TConnectionID`
  - `decimal_to_connid` — a decimal string to `TConnectionID`

Important

Pointers to the strings returned by `connid_to_[xxx]` point to the static memory and, hence, are not persistent. The duration of the returned strings is limited by several calls to the corresponding function.

TControlMode

Syntax

```
typedef enum {
        RegisterDefault,
        RegisterForce,
        RegisterLocal
} TControlMode;
```

Values

- `RegisterDefault` — T-Server decides whether to relay the registration request to the switch.
- `RegisterForce` — The registration request will be relayed to the switch.
- `RegisterLocal` — The registration request will not be relayed to the switch.

Important

If registration is required on the switch, T-Server may not distribute any events or accept requests for the DN in question.

Comments

Specifying this parameter makes sense only if the CTI link protocol of the switch in question supports the telephony object registration request.

TDirectoryNumber

Syntax

```
typedef char *TDirectoryNumber;
```

TDNRole

Syntax

```
typedef enum {
        RoleUnknown,
        RoleOrigination,
        RoleDestination,
        RoleConferenceMember,
        RoleNewParty,
        RoleAddedBy,
        RoleDeletedBy,
        RoleTransferredBy,
        RoleDeletedParty,
        RoleConferencedBy,
        RoleObserver
} TDNRole;
```

Values

- `RoleUnknown` — The party in question has an unknown role for this call.
- `RoleOrigination` — The party in question has made this call.
- `RoleDestination` — The party in question has received this call.
- `RoleConferenceMember` — The party in question has been engaged in the conference call.
- `RoleNewParty` — The party in question has been added to the call.
- `RoleAddedBy` — The party in question has added a new member to the call.
- `RoleDeletedBy` — The party in question has deleted another party from the call.
- `RoleTransferredBy` — The party in question has transferred the call.
- `RoleDeletedParty` — The party in question has been deleted from the conference.
- `RoleConferencedBy` — The party in question has conferenced the call.
- `RoleObserver` — The party in question is participating in the call as Service Observer.

Important

The following values provide redundant information that is available from the event type: `RoleConferenceMember`, `RoleNewParty`, `RoleAddedBy`, `RoleDeletedBy`, `RoleTransferredBy`, `RoleDeletedParty`, and `RoleConferencedBy`.

TEvent

Important

Although listed here, certain components of the TEvent structure are reserved for internal use only.

Syntax

```
typedef struct TEvent_tag {
        enum TMessageType                Event;
        TServer                        Server;
        int                        ReferenceID;
        char                        *HomeLocation;
        char                        *CustomerID;
        TConnectionID                ConnID;
        TConnectionID                PreviousConnID;
        TCallID                        CallID;
        int                        NodeID;
        TCallID                        NetworkCallID;
        int                        NetworkNodeID;
        TCallHistoryInfo                CallHistory;
        TCallType                        CallType;
        TCallState                CallState;
        TAgentID                        AgentID;
        TAgentWorkMode                WorkMode;
        long                        ErrorCode;
        char                        *ErrorMessage;
        TFile                        FileHandle;
        char                        *CollectedDigits;
        char                        LastCollectedDigit;
        TDirectoryNumber                ThisDN;
        TDirectoryNumber                ThisQueue;
        unsigned long                ThisTrunk;
        TDNRole                        ThisDNRole;
        TDirectoryNumber                OtherDN;
        TDirectoryNumber                OtherQueue;
        unsigned long                OtherTrunk;
        TDNRole                        OtherDNRole;
        TDirectoryNumber                ThirdPartyDN;
        TDirectoryNumber                ThirdPartyQueue;
        unsigned long                ThirdPartyTrunk;
        TDNRole                        ThirdPartyDNRole;
        TDirectoryNumber                DNIS;
        TDirectoryNumber                ANI;
        TAddressInfoType                InfoType;
        TAddressInfoStatus                InfoStatus;
        TTreatmentType                TreatmentType;
        TRouteType                RouteType;
        char                        *ServerVersion;
        TServerRole                ServerRole;
        TMask                        Capabilities;
        TKVList                        *UserData;
        TKVList                        *Reasons;
        TKVList                        *Extensions;
        TTimeStamp                Time;
        void                        *RawData;
        TDirectoryNumber                AccessNumber;
        TXRouteType                XRouteType;
        TReferenceID                XReferenceID;
        TKVList                        *TreatmentParameters;
        char                        *Place;
        int                        Timeout;
        TMediaType                MediaType;
        TLocationInfoType                LocationInfo;
        TMonitorNextCallType        MonitorNextCallType;
        TPrivateMsgType                PrivateEvent;
        /* Application data (set by TSetApplicationData) */
        void                        *ApplicationData;
} TEvent;
```

TEventMask

Syntax

```
#define T_MASK_LENGTH 64;
typedef unsigned char TMask[T_MASK_LENGTH];
#define TMaskSetAll(M) memset((M), 0xff, T_MASK_LENGTH)
#define TMaskClearAll(M) memset((M), 0, T_MASK_LENGTH)
#define TMaskSet(E, M) ((M)[(E) / 8] |= (1 << ((E) % 8)))
#define TMaskClear(E, M) ((M)[(E)/8] &= ~(1 << ((E) % 8)))
#define TMaskIsSet(E, M) ((M)[(E) / 8] & (1 << ((E) % 8)))
```

Comments

Important

The associated macros defined in the syntax for this type are used to set and clear the user-defined input mask.

For setting and clearing the user-defined input mask, use:

- `TMaskSetAll`
- `TMaskClearAll`
- `TMaskSet`
- `TMaskClear`

For checking the server capabilities mask returned by `TQueryServer()`, use `TMaskIsSet()`.

TFile

Syntax

```
typedef unsigned long TFile;
```

TForwardMode

Syntax

```
typedef enum {
        ForwardModeNone,
        ForwardModeUnconditional,
        ForwardModeOnBusy,
        ForwardModeOnNoAnswer,
        ForwardModeOnBusyAndNoAnswer,
        ForwardModeSendAllCalls
} TForwardMode;
```

Values

- `ForwardModeNone` — No forward mode specified.
- `ForwardModeUnconditional` — All incoming calls will be forwarded immediately.
- `ForwardModeOnBusy` — Incoming calls will be forwarded if this party is busy.
- `ForwardModeOnNoAnswer` — Incoming calls will be forwarded if not answered within the default time interval.
- `ForwardModeOnBusyAndNoAnswer` — Incoming calls will be forwarded if this party is busy or if not answered within the default time interval.
- `ForwardModeSendAllCalls` — All incoming calls will be forwarded immediately to the party specified in switch configuration to a specified destination.

TInterruptFlag

Syntax

```
typedef enum {
        Uninterruptable                = 1,
        Interruptable                = 2
} TInterruptFlag;
```

Values

- `Uninterruptable` — Playback of a voice message will not be interrupted when a touchtone button is pressed.
- `Interruptable` — Playback of a voice message will be interrupted when a touchtone button is pressed.

TKVList

Syntax

```
struct kv_list {
        struct _kv_pair                *list;
        struct _kv_pair                *current;
        struct _kv_pair                *tail;
};
typedef struct kv_list TKVList;
```

Values

- `list` — A pointer to the first pair on the list.
- `current` — A pointer to the current pair on the list.
- `tail` — A pointer to the last pair on the list.

TKVPair

Syntax

```
struct _kv_pair {
        KVType                        type;
        char                        *key;
        int                        length;
        union {
        char                        *_string_value;
        int                        _int_value;
        unsigned char                *_binary_value;
        struct kv_list                *_list_value;
        } _value;

        #define string_value        _value._string_value
        #define int_value                _value._int_value
        #define binary_value        _value._binary_value
        #define list_value                _value._list_value

        struct _kv_pair                *kv_next;
        struct _kv_pair                *kv_prev;
};
typedef struct _kv_pair TKVPair;
```

Values

- `type` — Variable type of `_value`.
- `key` — A pointer to the key name.
- `length` — Length of `_value` in bytes.
- `_value` — Current value of key.
- `kv_next` — A pointer to the next pair on the list.
- `kv_prev` — A pointer to the previous pair on the list.

TKVResult

Syntax

```
typedef enum {
        KVResultError                = -1,
        KVResultNotFound,
        KVResultSuccessful
} TKVResult;
```

Values

- `KVResultError` — Error condition. Returned to the application if the request has not been processed due to incorrect function statement.
- `KVResultNotFound` — Error condition. Returned to the application if the specified data structure has not been found.
- `KVResultSuccessful` — Successful completion of the requested function.

TKVType

Syntax

```
typedef enum {
        KVTypeString,
        KVTypeInt,
        KVTypeBinary,
        KVTypeList
        KVTypeUnicode
} TKVType;
```

Values

- `KVTypeString` — The variable is of a character type.
- `KVTypeInt` — The variable is of an integer type.
- `KVTypeBinary` — The variable is of a binary type.
- `KVTypeList` — The variable is of a `kv_list` type.
- `KVTypeUnicode` — The variable is of a unicode type.

TLocationInfoType

Syntax

```
typedef enum {
        LocationInfoAllLocations,
        LocationInfoLocationData,
        LocationInfoMonitorLocation,
        LocationInfoCancelMonitorLocation,
        LocationInfoMonitorAllLocations,
        LocationInfoCancelMonitorAllLocations,
        LocationInfoLocationMonitorCanceled,
        LocationInfoAllLocationsMonitorCanceled
} TLocationInfoType;
```

Values

- `LocationInfoAllLocations` — Requests information about all remote locations currently configured.
- `LocationInfoLocationData` — Requests information about the specified location.
- `LocationInfoMonitorLocation` — Requests that the specified location be monitored.
- `LocationInfoCancelMonitorLocation` — Requests that the specified location no longer be monitored.
- `LocationInfoMonitorAllLocations` — Requests that all locations be monitored.
- `LocationInfoCancelMonitorAllLocations` — Requests that all locations no longer be monitored.
- `LocationInfoLocationMonitorCanceled` — Notifies clients that monitoring of the specified location was canceled.
- `LocationInfoAllLocationsMonitorCanceled` — Notifies clients that monitoring of all locations was canceled.

TMakeCallType

Syntax

```
typedef enum {
        MakeCallRegular,
        MakeCallDirectAgent,
        MakeCallSupervisorAssist,
        MakeCallPriority,
        MakeCallDirectPriority
} TMakeCallType;
```

Values

- `MakeCallRegular` — The call type is regular.
- `MakeCallDirectAgent` — The call type is a direct call to agent.
- `MakeCallSupervisorAssist` — The call type is a call to supervisor.
- `MakeCallPriority` — The call type is regular with `Priority` feature.
- `MakeCallDirectPriority` — The call type is a direct call to agent with `Priority` feature.

TMediaType

Syntax

```
typedef enum {
        TMediaVoice                = 0,
        TMediaVoIP                = 1,
        TMediaEMail                = 2,
        TMediaVMail                = 3,
        TMediaSMail                = 4,
        TMediaChat                = 5,
        TMediaVideo                = 6,
        TMediaCobrowsing                = 7,
        TMediaWhiteboard                = 8,
        TMediaAppSharing                = 9,
        TMediaWebForm                = 10,
        TMediaWorkItem                = 11,
        TMediaCustom0                = 100
} TMediaType;
```

Values

Important
T-Server sets the TMediaVoice value in the messages it generates.

- TMediaVoice — Voice (PSTN) call
- TMediaVoIP — Voice over IP
- TMediaEMail — Electronic Mail
- TMediaVMail — Voice Mail
- TMediaSMail — Scanned Mail
- TMediaChat — Chat Session
- TMediaVideo — Video
- TMediaCobrowsing — Cobrowsing
- TMediaWhiteboard — Whiteboard
- TMediaAppSharing — Application Sharing
- TMediaWebForm — WebForm
- TMediaWorkItem — Work item (generic)

- `TMediaCustom0` — All values greater than this one are reserved for custom media types.

Comments

In order to support `MediaType` in T-Servers that work with multiple media, the field `media_type` is included to indicate the type of media associated with a given party in an interaction. This field appears in extensions of `EventRegistered` and `EventAddressInfo` with the key `mt-%d`.

TMergeType

Syntax

```
typedef enum {
        MergeForTransfer,
        MergeForConference
} TMergeType;
```

Values

- `MergeForTransfer` — The requested merge will result in a transfer of the held call.
- `MergeForConference` — The requested merge will result in a conference call.

TMessageType

Syntax

```
enum TMessageType {
        RequestRegisterClient,
        RequestQueryServer,
        RequestQueryAddress,
        RequestRegisterAddress,
        RequestUnregisterAddress,
        RequestRegisterAll,
        RequestUnregisterAll,
        RequestSetInputMask,
        RequestAgentLogin,
        RequestAgentLogout,
        RequestAgentReady,
        RequestAgentNotReady,
        RequestSetDNDOn,
        RequestSetDNDOff,
        RequestMakeCall,
        RequestMakePredictiveCall,
        RequestAnswerCall,
        RequestReleaseCall,
        RequestHoldCall,
        RequestRetrieveCall,
        RequestInitiateConference,
        RequestCompleteConference,
        RequestDeleteFromConference,
        RequestInitiateTransfer,
        RequestMuteTransfer,
        RequestSingleStepTransfer,
        RequestCompleteTransfer,
        RequestMergeCalls,
        RequestAlternateCall,
        RequestReconnectCall,
        RequestAttachUserData,
        RequestUpdateUserData,
        RequestDeleteUserData,
        RequestDeletePair,
        RequestCallForwardSet,
        RequestCallForwardCancel,
        RequestRouteCall,
        RequestGiveMusicTreatment,
        RequestGiveSilenceTreatment,
        RequestGiveRingBackTreatment,
        RequestLoginMailBox,
        RequestLogoutMailBox,
        RequestOpenVoiceFile,
        RequestCloseVoiceFile,
        RequestPlayVoiceFile,
        RequestCollectDigits,
        RequestSetMessageWaitingOn,
        RequestSetMessageWaitingOff,
        RequestDistributeUserEvent,
        RequestDistributeEvent,
        EventServerConnected,
        EventServerDisconnected,
        EventError,
        EventRegistered,
        EventUnregistered,
        EventRegisteredAll,
        EventUnregisteredAll,
        EventQueued,
        EventDiverted,
        EventAbandoned,
        EventRinging,
        EventDialing,
        EventNetworkReached,
        EventDestinationBusy,
        EventEstablished,
        EventReleased,
        EventHeld,
        EventRetrieved,
        EventPartyChanged,
        EventPartyAdded,
        EventPartyDeleted,
        EventRouteRequest,
        EventRouteUsed,
        EventAgentLogin,
        EventAgentLogout,
        EventAgentReady,
        EventAgentNotReady,
        EventDNDOn,
        EventDNDOff,
        EventMailBoxLogin,
        EventMailBoxLogout,
        EventVoiceFileOpened,
        EventVoiceFileClosed,
        EventVoiceFileEndPlay,
        EventDigitsCollected,
        EventAttachedDataChanged,
        EventOffHook,
        EventOnHook,
        EventForwardSet,
        EventForwardCancel,
        EventMessageWaitingOn,
        EventMessageWaitingOff,
        EventAddressInfo,
        EventServerInfo,
        EventLinkDisconnected,
        EventLinkConnected,
        EventUserEvent,

        RequestSendDTMF,
        EventDTMFSent,

        EventResourceAllocated,
        EventResourceFreed,

        EventRemoteConnectionSuccess,
        EventRemoteConnectionFailed,

        RequestRedirectCall,
        RequestListenDisconnect,
        RequestListenReconnect,
        EventListenDisconnected,
        EventListenReconnected,
        RequestQueryCall,
        EventPartyInfo,
        RequestClearCall,

        RequestSetCallInfo,
        EventCallInfoChanged,

        RequestApplyTreatment,
        EventTreatmentApplied,
        EventTreatmentNotApplied,
        EventTreatmentEnd,

        EventHardwareError,
        EventAgentAfterCallWork,
        EventTreatmentRequired,

        RequestSingleStepConference,
        RequestQuerySwitch,
        EventSwitchInfo,

        RequestGetAccessNumber,
        RequestCancelReqGetAccessNumber,
        EventAnswerAccessNumber,
        EventReqGetAccessNumberCanceled,

        RequestReserveAgent,
        EventAgentReserved,
        RequestReserveAgentAndGetAccessNumber,

        RequestAgentSetIdleReason,
        EventAgentIdleReasonSet,

        EventRestoreConnection,
        EventPrimaryChanged,
        RequestLostBackupConnection,
        RequestSetDNInfo,

        RequestQueryLocation,
        EventLocationInfo,

        EventACK,

        RequestMonitorNextCall,
        RequestCancelMonitoring,
        EventMonitoringNextCall,
        EventMonitoringCancelled,

        RequestSetMuteOn,
        RequestSetMuteOff,
        EventMuteOn,
        EventMuteOff,

        EventDNOutOfService,
        EventDNBackInService,

        RequestPrivateService,
        EventPrivateInfo,

        EventBridged,
        EventQueueLogout,

        EventReserved_1,
        EventReserved_2,
        EventReserved_3,
        EventReserved_4,
        EventReserved_5,
        EventReserved_6,
        EventReserved_7,

        EventCallCreated,
        EventCallDataChanged,
        EventCallDeleted,
        EventCallPartyAdded,
        EventCallPartyState,
        EventCallPartyMoved,
        EventCallPartyDeleted,

        RequestStartCallMonitoring,
        RequestStopCallMonitoring,

        RequestSendReturnReceipt,
        EventReturnReceipt,

        RequestNetworkConsult,
        RequestNetworkAlternate,
        RequestNetworkTransfer,
        RequestNetworkMerge,
        RequestNetworkReconnect,
        RequestNetworkSingleStepTransfer,
        RequestNetworkPrivateService,
        EventNetworkCallStatus,
        EventNetworkPrivateInfo,

        MessageIDMAX
};
```

Comments

`TMessageType` is a unique identifier assigned to all types of requests and events.

`MessageIDMAX` defines the maximum value of a request.

TMonitorNextCallType

Syntax

```
typedef enum {
        MonitorOneCall,
        MonitorAllCalls
} TMonitorNextCallType;
```

Values

- `MonitorOneCall` — Supervisor will monitor one call to the agent.
- `MonitorAllCalls` — Supervisor will monitor all subsequent calls to the agent until monitoring is canceled.

TNetworkCallState

Syntax

```
typedef enum TNetworkCallState_tag {
        NetworkCallStateUnknown                = 0,
        NetworkCallStateConsulting                = 1,
        NetworkCallStateConsultHeld                = 2,
        NetworkCallStateTransferred                = 3,
        NetworkCallStateConferenced                = 4,
        NetworkCallStateReconnected                = 5,
        NetworkCallStateDisconnected        = 6,
        NetworkCallStateNull                = 7
} TNetworkCallState;
```

Values

- `NetworkCallStateUnknown` — The network call state is unknown.
- `NetworkCallStateConsulting` — The original call is on hold, and a consultation leg for the call has been added. (Depending on the call's `NetworkDestState`, this leg may be logical, corresponding to the call being routed by URS. Or the call may exist physically on the network device in an initiated or connected state.)
- `NetworkCallStateConsultHeld` — A network consultation call has been parked, with Agent 1 reconnected to the origination party. Or, the consultation request has been made so as not to interrupt the active conversation (usually applicable to targets that URS selects).
- `NetworkCallStateTransferred` — The network call has been transferred to Agent 2, and Agent 1 has been dropped from the call.
- `NetworkCallStateConferenced` — Three parties have been bridged for a network consultation call.
- `NetworkCallStateReconnected` — After a network consultation, the call has been reconnected to the original target, and the consult leg has been dropped.
- `NetworkCallStateDisconnected` — The party to the network consultation call has disconnected. However, the network device has not ended the call, but has instead connected the two agents.
- `NetworkCallStateNull` — The caller has disconnected from the call. The network device is about to drop the call, or has dropped it already.

TNetworkDestState

Syntax

```
typedef enum TNetworkDestState_tag {
        NetworkDestStateUnknown                = 0,
        NetworkDestStateRouting                = 1,
        NetworkDestStateDelivering                = 2,
        NetworkDestStateNoParty                = 3,
        NetworkDestStateOk                        = 4
} TNetworkDestState;
```

Values

- `NetworkDestStateUnknown` — For a network call, the state of the destination DN is unknown.
- `NetworkDestStateRouting` — A consultation has been requested on a route point. This state is entered when the `EventRouteRequest` is sent.
- `NetworkDestStateDelivering` — Either a consultation request has been made directly to a DN, or `TRouteCall()` has been received from URS. `NetworkDestStateDelivering` is entered when a consultation request is sent to the network device.
- `NetworkDestStateNoParty` — `NetworkDestStateNoParty = NetworkDestStateFailed`. The consultation call failed to reach the intended destination (for instance, because of a busy condition), or the consulted party has disconnected from the call.
- `NetworkDestStateOk` — The consultation call has been connected to its destination.

TNetworkPartyRole

Syntax

```
typedef enum TNetworkPartyRole_tag {
        RoleNtwUnknown                = 0,
        RoleNtwOrigParty                = 1,
        RoleNtwDestParty                = 2
} TNetworkPartyRole;
```

Values

- `RoleNtwUnknown` — The party in question has an unknown role for this call.
- `RoleNtwOrigParty` — The party in question has made this call.
- `RoleNtwDestParty` — The party in question has received this call.

TOpenMode

Syntax

```
typedef enum {
        SyncMode,
        AsyncMode
} TOpenMode;
```

Values

- `SyncMode` — Synchronous client-server communication mode. This is a blocking mode, which means the read or write function does not return to the calling process until the operation is complete and the requested data has been written into or read from the socket.
- `AsyncMode` — Asynchronous client-server communication mode. The asynchronous communication mode requires the use of the `TScanServerEx` function. To force delivery of pending messages, use `TScanServerEx` with the `TSCAN_MODE_WRITE` scanning mode. This is a non-blocking mode. In this mode, the result of the request is returned immediately after the operation is started or queued.

TPartyState

Syntax

```
type def enum TPartyState_tag{
        PtState_NULL                = 0,
        PtState_Initiated                = 0x0001,
        PtState_Queued                = 0x0009,
        PtState_Alerting                = 0x000A,
        PtState_Busy                = 0x000B,
        PtState_Connected                = 0x000C,
        PtState_CP_Detect                = 0x000D,
        PtState_Held                = 0x000E,
        PtState_Failed                = 0x000F,
        PtState_NoListen                = 0x0010,
        PtState_NoTalk                = 0x0020,
        PtState_Bridged                = 0x0040,
        PtState_Audit                = 0x0080,
        PtState_SvcObserving        = 0x00A0,
        PtState_TreatmentReq        = 0x0100,
        PtState_Treatment                = 0x0200,
        PtState_Routing                = 0x0800,
        PtStateMod_Dialing                = 0x10000,
        PtStateMod_Uncertain        = 0x20000,
} TPartyState;
```

Values

- `PtState_Initiated` — A call has been initiated on behalf of a device, but that device has not been connected to the call yet.
- `PtState_Queued` — A call is queued on a given device, and the call awaits the availability of some service (for example, ACD queue distribution) or of some device (for example, a phone line).
- `PtState_Alerting` — A call is alerting on a device, indicating an incoming call (for example, the phone is ringing); or a call is in the process of being distributed to a destination (for example, is being processed by the telephony network).
- `PtState_Busy` — A call cannot reach the intended device, which is busy.
- `PtState_Connected` — A given device has a voice connection with other participants.
- `PtState_CP_Detected` — The originating device (a queue or route point) for a predictive dialing call is between the initiation of the call (`EventDialing` has been sent) and the moment when the call is either queued or released.
- `PtState_Held` — A device has temporarily suspended its connection to other call participants.
- `PtState_Failed` — A call originating on a given device has not succeeded (either the dialed number is wrong, or the switch was not able to allocate the trunk). This state is used instead of Busy when a destination party was never created for the call.

- `PtState_NoListen` — A party cannot hear other participants on the call.
- `PtState_NoTalk` — A party is muted, and other participants on the call do not hear that party.
- `PtState_Bridged` — A party is attached to the call with the "Bridged Call Appearance" feature.
- `PtState_Audit` — A party is attached to a call as "Service Observer" or "Service Assistant."
- `PtState_SvcObserving` — A party is attached to a call as "Service Observer" or "Service Assistant" and is muted such that other participants on the call do not hear that party.
- `PtState_TreatmentReq` — The switch is waiting for a treatment to be applied to the call.
- `PtState_Treatment` — A treatment has been applied while the call is located on a given device.
- `PtState_Routing` — The switch is waiting for routing instructions.
- `PtStateMod_Dialing` — A telephony object related to a party has completed a dialing sequence.
- `PtStateMod_Uncertain` — A telephony object related to a party is in an unknown state.

TPrivateMsgType

Syntax

```
typedef int TPrivateMsgType;
```

Comments

Used in `TPrivateService()`, `EventPrivateInfo`, `TNetworkPrivateService()`, and `EventNetworkPrivateInfo`.

TRegisterMode

Syntax

```
typedef enum {
        ModeShare,
        ModePrivate,
        ModeMonitor
} TRegisterMode;
```

Values

- `ModeShare` — This registration mode allows other client applications to receive events about the same telephony objects.
- `ModePrivate` — This registration mode does not allow other client applications to receive events about the same telephony objects.
- `ModeMonitor` — The application that has registered in this mode will only receive events about the telephony objects in question; it will not be able to request any functions.

Comments

On some switching platforms, T-Server does not register RouteDNs at the switch if there are no clients registered, or if all clients are registered in `ModeMonitor`. This allows the switch to apply routing by default, without waiting for a response from T-Server.

For all applications not responding to Route Requests, Genesys recommends using `ModeMonitor` when registering at a RouteDN.

TReliability

Syntax

```
typedef enum {
        TReliabilityOk                = 0,
        TReliabilityInPast                = 1,
        TReliabilityUncertain        = 2,
        TReliabilityExternal        = 3
} TReliability;
```

Values

- `TReliabilityOk` — A `TEvent`'s time stamp reliably indicates the time of the event's occurrence. (`TEvent` was generated based on its corresponding switch notification.)
- `TReliabilityInPast` — Indicates an event which occurred prior to the time stamp of this `TEvent`. (`TEvent` was constructed based on subsequent switch notifications.)
- `TReliabilityUncertain` — Indicates an event which may or may not have yet occurred at the time stamp of this `TEvent`. (`TEvent` was generated based on a configured timeout or as a result of internal cleanup.)
- `TReliabilityExternal` — Indicates an event has been generated as a result of a request from external controller. (`TEvent` was generated based on a corresponding external request—SNMP.)

TRemoteParty

Syntax

```
typedef struct TRemoteParty_tag {
        char                        *HomeLocation;
        TDirectoryNumber                OrigDN;
} TRemoteParty;
```

Values

- `HomeLocation` — A pointer to the location of the T-Server that has activated the transfer.
- `OrigDN` — Value of the DN of the T-Server that has activated transfer.

TRouteType

Syntax

```
typedef enum {
        RouteTypeUnknown,
        RouteTypeDefault,
        RouteTypeLabel,
        RouteTypeOverwriteDNIS,
        RouteTypeDDD,
        RouteTypeIDDD,
        RouteTypeDirect,
        RouteTypeReject,
        RouteTypeAnnouncement,
        RouteTypePostFeature,
        RouteTypeDirectAgent,
        RouteTypePriority,
        RouteTypeDirectPriority
        RouteTypeAgentID
        RouteTypeCallDisconnect
        RouteTypeIDMAX
} TRouteType;
```

Values

- `RouteTypeUnknown` — Route type is unknown.
- `RouteTypeDefault` — Routing should be performed according to configuration preset by the switch.

Important

The routing application instructs T-Server to route the call to a selected target DN. If the selected target DN is empty, T-Server may instruct the switch to route the call to the default target defined by the switch.

- `RouteTypeLabel` — Routing should be performed according to the switch-specific label.
- `RouteTypeOverwriteDNIS` — The original DNIS is requested to be overwritten with the number specified by the parameter `dnis`.
- `RouteTypeDDD` — Route to the domestic (in-country) number specified by the parameter `destination`.
- `RouteTypeIDDD` — Route to the international number specified by the parameter `destination`.
- `RouteTypeDirect` — Reserved.
- `RouteTypeReject` — The request for routing has been rejected.

- `RouteTypeAnnouncement` — Specific to an AT&T Network.
- `RouteTypePostFeature` — Specific to an AT&T Network.
- `RouteTypeDirectAgent` — Route to the specific agent number specified by the parameter `destination` (specific to the Avaya Communication Manager).
- `RouteTypePriority` — Routing should be performed according to configuration for calls that have priority feature preset by the switch (specific to the Avaya Communication Manager).
- `RouteTypeDirectPriority` — Routing should be performed to the specific agent number specified by the parameter `destination` according to configuration for calls that have priority feature preset by the switch (specific to the Avaya Communication Manager).
- `RouteTypeAgentID` — Routing should be performed to the agent specified by the parameter `AgentID`.
- `RouteTypeCallDisconnect` — Call is to be released from the route point using the CTI link.
- `RouteTypeIDMAX` — See comments in the TMessageType entry.

TScanServerMode

Syntax

```
typedef enum {
        TSCAN_MODE_DEFAULT,
        TSCAN_MODE_SYNC,
        TSCAN_MODE_ASYNC,
        TSCAN_MODE_CONNECT,
        TSCAN_MODE_WRITE,
        TSCAN_MODE_WRITE_ALL
} TScanServerMode;
```

Values

- `TSCAN_MODE_DEFAULT` — Use with `SyncMode` connections. This provides the same functionality as `TScanServer()`, with the addition of processing ADDP timeouts, if ADDP protocol is configured.
- `TSCAN_MODE_SYNC` — Provides the same functionality as `TSCAN_MODE_DEFAULT`.
- `TSCAN_MODE_ASYNC` — Use with `AsyncMode` connections. In this case, `TScanServerEx()` also sends pending messages for a given server, if applicable.
- `TSCAN_MODE_CONNECT` — Use after `TOpenServer()` to wait for a connection that is opening in asynchronous mode so that the connection is properly opened and registered.

Important

This mode is deprecated, and remains available primarily for backward compatibility. `TSCAN_MODE_ASYNC` should be used for `AsyncMode` connections in most cases.

- `TSCAN_MODE_WRITE` — When added to `TSCAN_MODE_SYNC`, pending messages for a given server are sent. This parameter does not affect `TSCAN_MODE_ASYNC`.
- `TSCAN_MODE_WRITE_ALL` — When added to either `TSCAN_MODE_SYNC` or `TSCAN_MODE_ASYNC`, pending messages for all registered connections are sent.

TServer

Syntax

```
typedef int TServer;
```

TServerRole

Syntax

```
typedef enum {
        ServerRolePrimary,
        ServerRoleBackup
} TServerRole;
```

Values

- `ServerRolePrimary` — The role of this T-Server is primary.
- `ServerRoleBackup` — The role of this T-Server is backup.

TSetOpType

Syntax

```
typedef enum {
        CreateCallInfo,
        SetCallInfo,
        DeleteCallInfo,
        AddParty,
        DeleteParty
} TSetOpType;
```

Values

- `CreateCallInfo` — Request to create new call information.
- `SetCallInfo` — Request to set call information.
- `DeleteCallInfo` — Request to delete call information.
- `AddParty` — Request to add a party to the existing call information.
- `DeleteParty` — Request to delete a party from the existing call information.

TSwitchInfoType

Syntax

```
typedef enum {
        SwitchInfoDateTime
        SwitchInfoClassifierStat
} TSwitchInfoType;
```

Values

- `SwitchInfoDateTime` — Requests current date and time information as stated on the switch.
- `SwitchInfoClassifierStat` — Returns the numbers of idle classifiers and of classifiers in use.

TTime

Syntax

```
typedef unsigned long TTime;
```

TTimeStamp

Syntax

```
typedef struct TTimestamp_tag{
        unsigned long        seconds;
        unsigned long        useconds;
} TTimeStamp;
```

Values

- `seconds` — Time calculated since 00:00:00 GMT January 1, 1970, measured in seconds.
- `useconds` — Amount of time, up to `1000` milliseconds, for use as a more precise interval within the `TTimeStamp` structure.

TTreatmentType

Syntax

```
typedef enum {
        TreatmentUnknown,
        TreatmentIVR,
        TreatmentMusic,
        TreatmentRingBack,
        TreatmentSilence,
        TreatmentBusy,
        TreatmentCollectDigits,
        TreatmentPlayAnnouncement,
        TreatmentPlayAnnouncementAndDigits,
        TreatmentVerifyDigits,
        TreatmentRecordUserAnnouncement,
        TreatmentDeleteUserAnnouncement,
        TreatmentCancelCall,
        TreatmentPlayApplication,
        TreatmentSetDefaultRoute,
        TreatmentTextToSpeech,
        TreatmentTextToSpeechAndDigits,
        TreatmentFastBusy,
        TreatmentRAN,
} TTreatmentType;
```

Attributes

- `TreatmentUnknown` — Reserved to catch improperly formatted requests.
- `TreatmentIVR` — Treatment type is IVR. Used to connect the call specified by `conn_id` to the IVR.
  The following table lists key-value pair(s) used in the attribute parameters for `TreatmentIVR`:

Key-Value Pairs for TreatmentIVR

Key	Value
`LABEL`	Routing target address if a connection to T-Server is lost.
`DNIS`	Value is used instead of existing DNIS if connection to T-Server is lost.

- `TreatmentMusic` — Treatment type is music. Music Treatment is used to connect the call specified by `conn_id` to the music source.
  The following table lists key-value pair(s) used in the attribute parameters for `TreatmentMusic`:

Key-Value Pairs for TreatmentMusic

Key	Value
MUSIC_DN	Specifies the directory number of the music source. The key is used only for network interface.

- `TreatmentRingBack` — Treatment type is ring back. Ring back treatment is used to connect the call specified by `conn_id` to a Ring Back Tone source.
- `TreatmentSilence` — Treatment type is silence. Silence treatment is used to connect the call specified by `conn_id` to a silence source.
- `TreatmentBusy` — Treatment type is busy. Busy Treatment is used to connect the call specified by `conn_id` to a busy tone source.
- `TreatmentCollectDigits` — Treatment type is collect digits. Collect digits is used to collect digits from the calling party.
  The following table lists key-value pair(s) used in the attribute parameters for `TreatmentCollectDigits`:

Key-Value Pairs for TreatmentCollectDigits

Key	Value
`MAX_DIGITS`	The maximum number of digits to be collected. Maximum number of digits that may be collected is 31. Note that the Maximum number of digits may be equal to 0. In this case, no time should be spent waiting for the calling party to input digits, and a response should be returned indicating that `0` digits have been collected.
`ABORT_DIGITS`	This sequence of up to two keys causes the digits collection operation to be aborted. If this sequence appears, the IVR considers this as a failed digits collection attempt.
`IGNORE_DIGITS`	This sequence of up to two keys is treated as though they have not been pressed.
`BACKSPACE_DIGITS`	This sequence of up to two keys causes the previous keystroke to be thrown away.
`TERM_DIGITS`	This sequence of up to two keys causes all the digits, not including the TERM_DIGITS, to be returned to the calling application as the collected digits.
`RESET_DIGITS`	This sequence of up to two keys causes all the previous keystrokes to be thrown away. The digits collection resumes.
`CLEAR_FLAG`	Clear flag indicates whether any input information should be cleared before digit collection starts. Not supported in `GR-1129-CORE` protocol implementation. Valid values are 1 and 2 of type `KVTypeInt`. See TClearFlag.
`START_TIMEOUT`	The number of seconds the IVR should wait for the calling party to begin the DTMF digits input.
`DIGIT_TIMEOUT`	The number of seconds the IVR should wait between the DTMF digits.

Key	Value
TOTAL_TIMEOUT	The total number of seconds the IVR should wait for the calling party to provide the requested DTMF input.

- `TreatmentPlayAnnouncement` — Treatment type is play announcement. Used to play an announcement block to the calling party. The entire announcement block can consist of a series of announcement elements pieced together. Each announcement element can be Interruptable or non-Interruptable. For more information on the use of key-value pairs with `TreatmentPlayAnnouncement`, see the example in the Comments section below.

> **Important**
> For `TreatmentPlayAnnouncement`, if the `INTERRUPTABLE` flag is missing, then the resulting behavior is device specific.

- The following table lists key-value pair(s) used in the attribute parameters for `TreatmentPlayAnnouncement`:

Key-Value Pairs for TreatmentPlayAnnouncement

Key	Value
PROMPT	Contains a set of up to 10 elements. Each set is numbered in ranges from 1 to 10.
PROMPT.1 ... PROMPT.10	Each element contains the number of entries describing an announcement element: • `INTERRUPTABLE` (Boolean `0/1`)—indicates whether a caller can interrupt the announcement. • TEXT—ASCII text to pronounce using text-to-speech technology (if supported by the IVR equipment). One of the following three options (mutually exclusive): • `ID integer`—ID of a message to play • `DIGITS`—numbers to pronounce. First digit defines how the digits are to be pronounced: • 0—one at a time • 1—date • 2—time • 3—phone number • 4—money • `USER_ID`—contains a user ID string,

Key	Value
	`USER_ANN_ID`, User Announcement ID (integer), is returned earlier after successful `RecordUserAnnouncement()` request.
`LANGUAGE`	Optional language indicator. Contains a string specifying a language in which the announcement should be made. The valid languages include: • English (US) • Spanish • Mandarin • Cantonese • Vietnamese • French • French (Canada) • German • Italian • Japanese • Korean • Russian

- `TreatmentPlayAnnouncementAndDigits` — Treatment type is play announcement and digits. It is used to play an announcement block and collect digits from the calling party. Typically, an announcement includes instructions asking the calling party to provide some information.
  Key-value pair(s) used in the attribute parameters for `TreatmentPlayAnnouncementAndDigits`: All of the `PlayAnnouncement` and `CollectDigit` parameters are recognized.

Important

For `TreatmentPlayAnnouncementAndDigits`, if the `INTERRUPTABLE` flag is missing, then the resulting behavior is device specific.

- `TreatmentVerifyDigits` — Treatment type is digits verification. It is used to prompt a calling party to enter digits that will be compared with a desired response.
  The following table lists key-value pair(s) used in the attribute parameters for TreatmentVerifyDigits:

Key-Value Pairs for TreatmentVerifyDigits

Key	Value
`PROMPT`	Contains the elements specifying the initial

Key	Value
	announcement block to be played.
`REPROMPT`	Contains the elements specifying the announcement block to be played after verification has failed and asking the caller to reenter the input.
`SUCCESS`	Contains the elements specifying the success announcement.

`FAILURE`	Contains the elements specifying the failure announcement. The general format for the preceding 4 blocks is the same as described in `PlayAnnouncement`, with additional limitations imposed by the current version of the `GR-1129-CORE` document. Each announcement block may contain only one announcement element. The announcement element must be of announcement ID type. Digit announcements or user announcements must not be used. This means that the only currently accessible format for these announcements is:
	`“PROMPT.1.ID"   = <integer announcement ID>`
	`"REPROMPT.1.ID" = <integer announcement ID>`
	`"FAILURE.1.ID"  = <integer announcement ID>`
	`"SUCCESS.1.ID"  = <integer announcement ID>`

`TIMEOUT`	Contains sublists specifying a no input timeout announcement.
`LANGUAGE`	Optional language indicator. Contains a string specifying a language in which the announcement should be made.
`COMPARE_DIGITS`	Contains the actual digits that should be compared against the caller input. Note that three `COMPARE_xxx` options are mutually exclusive and only one may be present at a time.
`COMPARE_USER_ID`	Contains the user ID string that should be used by the IVR to index into the local table for verification.
`COMPARE_PLAN_ID`	Contains the dialing plan index. Dialing plan is used to compare with the general format compliance.
`NUM_ATTEMPTS`	The number of attempts the calling party can make before failing the verification.
`NUM_ATTEMPTS_TIMEOUT`	Number of attempts where caller enters no input until timeout.
`NUM_DIGITS`	The number of digits to be collected. The maximum number of digits that may be collected is 31. Note that `NUM_DIGITS` may be equal to `0`. In this case, no time should be spent waiting for the caller input digits, and a response should be returned indicating `0` digits collected.
`ABORT_DIGITS`	This sequence of up to two keys causes the digits collection operation to be aborted. If this sequence

Key	Value
	appears, the IVR considers this as a failed digits collection attempt.
`IGNORE_DIGITS`	This sequence of up to two keys is treated as though they have not been pressed.
`BACKSPACE_DIGITS`	This sequence of up to two keys causes the previous keystroke to be thrown away.
`TERM_DIGITS`	This sequence of up to two keys causes all the digits, not including `TERM_DIGITS`, to be returned to the calling application as collected digits.
`RESET_DIGITS`	This sequence of up to two keys causes all previous keystrokes to be thrown away. The digits collection resumes.
`CLEAR_FLAG`	Clear flag indicates whether any input information should be cleared before digit collection starts. Not supported in `GR-1129-CORE` protocol implementation. Valid values are 1 and 2 of `KVTypeInt` type. See TClearFlag.
`START_TIMEOUT`	The number of seconds the IVR should wait for the caller to begin the DTMF digits dialing.
`DIGIT_TIMEOUT`	The number of seconds the IVR should wait between the DTMF digits.
`TOTAL_TIMEOUT`	The total number of seconds the IVR should wait for the caller to provide the requested DTMF input.

- `TreatmentRecordUserAnnouncement` — Treatment type is record user announcement. It is used to create a user-specific announcement. Treatment device returns an announcement ID for a newly created announcement, which the application may use later to trigger playback of the user-specific announcement.
  The following table lists key-value pair(s) used in the attribute parameters for `TreatmentRecordUserAnnouncement`:

Key-Value Pairs for TreatmentRecordUserAnnouncement

Key	Value
`PROMPT`	Contains elements specifying an initial announcement block to be played. The general format for the elements is the same as described in `PlayAnnouncement`, with additional limitations imposed by the current version of the `GR-1129-CORE` document. Each announcement block may contain up to 10 announcement elements. In addition, the announcement element must be of announcement ID type. Digit announcements or user announcements must not be used.
`USER_ID`	User ID string specifies the user for which the announcement will be recorded.
`ABORT_DIGITS`	The sequence of up to two keys that may be entered by the calling party to abort the recording

Key	Value
	process. The IVR considers this as a failed recording attempt.
`TERM_DIGITS`	The sequence of up to two keys that may be entered by the calling party to indicate that recording of the message is finished.
`RESET_DIGITS`	The sequence of up to two keys that may be entered by the caller to restart recording the message. Any message recorded up to the point of these keystrokes will be thrown away.
`START_TIMEOUT`	The number of seconds the IVR should wait for the caller to begin recording the message.
`TOTAL_TIMEOUT`	The total number of seconds the IVR should wait for the caller to finish recording the message.

- `TreatmentDeleteUserAnnouncement` — Treatment type is delete user announcement. It is used to remove the user-specific announcement from an IVR.
  The following table lists key-value pair(s) used in the attribute parameters for `TreatmentDeleteUserAnnouncement`:

Key-Value Pairs for TreatmentDeleteUserAnnouncement

Key	Value
`USER_ID`	User ID string specifies the user for which the announcement will be deleted.
`USER_ANN_ID`	User Announcement ID (integer) as returned in the EventTreatmentEnd event after the successful RecordUserAnnouncement request.

- `TreatmentCancelCall` — Treatment type is cancel. It is used to terminate processing asynchronously for a given call in progress (that is, the call should be disconnected). This message should not be used for the normal ending of a conversation between T-Server and IVR. `EventReleased` is expected to be sent in response to the request to cancel. If partial or full input is collected prior to receiving this treatment, then the appropriate `EventTreatmentApplied` and `EventTreatmentEnd` events containing the input data may be sent just before `EventReleased`.
  The following table lists key-value pair(s) used in the attribute parameters for `TreatmentCancelCall`:

Key-Value Pairs for TreatmentCancelCall

Key	Value
`REPORT`	If set to 1, T-Server waits for a response from the IVR that a request was or was not processed. If set to `0` or not specified, then T-Server does not wait for a confirmation from the IVR and follows a call scenario.

- `TreatmentPlayApplication` — Treatment type is play application. It is used to execute an application or a script on the IVR device. It is possible to pass parameters to the application and to get some return values. The standards do not limit what can be exchanged in parameters; they only specify a way these parameters should be encoded.

The following table lists key-value pair(s) used in the attribute parameters for `TreatmentPlayApplication`:

Key-Value Pairs for TreatmentPlayApplication

Key	Value
`APP_ID`	Application ID (integer) specifies the application to be run.

Important

All application-specific parameters have an integer ID number. Though an integer, it is represented as a string in a key field of KVList, but is converted to or from an integer type by T-Server. Actual values of these parameters are application specific. The type of KVList entries can be as follows:

- Integer—passed as an integer to the IVR.
- String—passed as digits to the IVR (digits only, no characters).
- Binary—passed as a string to IVR.

Return parameters from the application are passed in the `Extensions` attribute in `EventTreatmentEnd`. Integer IDs are used as the keys; the values are application specific; and the type conversion is as follows:

- Integer—passed as a KVList integer.
- Boolean—passed as a KVList boolean.
- Digits—passed as a KVList string.
- String—passed as a KVList string.
- Real—not supported.

- `TreatmentSetDefaultRoute` — Treatment type is `SetDefaultRoute`. Sets default routing destination that is specified in the parameter dn in the call to `TApplyTreatment()`. The default routing destination is common for every object controllable by the IVR. The IVR can forward calls to that destination when there is no response from Genesys Interaction Router (IR) within the specified timeout defined in the IVR or when the connection to T-Server is lost.
- `TreatmentTextToSpeech` — Treatment type is text to speech. It is essentially the same as `TreatmentPlayAnnouncement`, where all announcement elements are of type TEXT. `TextToSpeech`, however, is a less flexible option because it does not allow recorded announcements with text-to-speech synthesis to be intermixed in a single block; on less sophisticated equipment, it may be the only option available.
  The following table lists key-value pair(s) used in the attribute parameters for `TreatmentTextToSpeech`:

Key-Value Pairs for TreatmentTextToSpeech

Key	Value
`PROMPT`	Contains a number of sublists (from 1 to 10); each

Key	Value
	sublist is named with a number from 1 to `10`.
`PROMPT.1` ... PROMPT.10	Each sublist contains entries describing an announcement element. The entries are: • `INTERRUPTABLE (Boolean 0/1)`—indicates whether caller can interrupt the announcement. • TEXT— ASCII text to pronounce using text-to-speech technology.
LANGUAGE	Optional language indicator. Contains a string specifying a language in which the announcement should be made.

Important

For `TreatmentTextToSpeech`, if the `INTERRUPTABLE` flag is missing, then the resulting behavior is device specific.

- `TreatmentTextToSpeechAndDigits` — Treatment type that requires generating speech from the text, then collecting digits. Typically the speech request would include instructions asking the caller to provide some input.
  Key-value pair(s) used in the attribute parameters for `TreatmentTextToSpeechAndDigits`: All of the `TextToSpeech` and `CollectDigit` parameters are recognized.
- `TreatmentFastBusy` — Treatment type is `FastBusy`. It is used to connect the call specified by `conn_id` to a fast busy tone source.
  Key-value pair(s) used in the attribute parameters for `TreatmentFastBusy`: none.
- `TreatmentRAN` — Treatment type is RAN. It is used to connect the call specified by `conn_id` to the RAN source.
  One key-value pair is used in the attribute parameters for `TreatmentRAN`:

Key-Value Pairs for TreatmentRAN

Key	Value
`RAN`	The directory number of RAN source.

Comments

The following example shows how to use the key-value pairs for `TreatmentPlayAnnouncement`.

In this example, the `KVList` parameter requests T-Server to play three announcements in sequence. The first, an announcement with ID 123, should be played; the second, an amount of money ($10 US), should be pronounced; and, finally, a user-defined announcement (with ID 456 and user ID 1234) should be played. Only the first announcement is non-interruptable.

```
"PROMPT.1.INTERRUPTABLE"=0
"PROMPT.1.ID"=123
"PROMPT.2.INTERRUPTABLE"=1
"PROMPT.2.DIGITS"="31000"
"PROMPT.3.INTERRUPTABLE"=1
"PROMPT.3.USER_ID"="1234"
"PROMPT.3.USER_ANN_ID"=456
```

Important

In this example, a dot (.) means child. That is, a `KVPair "INTERRUPTABLE"=0` belongs to `KVList 1`, which in turn belongs to `KVList PROMPT`.

TXCaps

Syntax

```
typedef int TXCaps;
```

Comments

Used in `TXCapsSupported()` to indicate support for `enum_value` in the function call `TGetXCaps` for the T-Server in question.

TXRouteType

Syntax

```
typedef enum {
        XRouteTypeDefault                      = -1,
        XRouteTypeMIN                      = 0,
        XRouteTypeRoute                      = 0,
        XRouteTypeDirect                      = 1,
        XRouteTypeReroute                      = 2,
        XRouteTypeDirectUUI                 = 3,
        XRouteTypeDirectANI                 = 4,
        XRouteTypeDirectNoToken                 = 5,
        XRouteTypeDNISPooling                 = 6,
        XRouteTypeDirectDigits                 = 7,
        XRouteTypePullBack                       = 8,
        XRouteTypeExtProtocol                 = 9,
        XRouteTypeRouteUUI                       = 10,
        XRouteTypeMAX
} TXRouteType;
```

Values

- XRouteTypeRoute — Specifies ISCC strategy type `route`.
- XRouteTypeDirect — Specifies ISCC strategy type `direct-callid`.
- XRouteTypeMIN — For internal use only.
- XRouteTypeReroute — Specifies ISCC strategy type `reroute`.
- XRouteTypeDirectUUI — Specifies ISCC strategy type `direct-uui`.
- XRouteTypeDirectANI — Specifies ISCC strategy type `direct-ani`.
- XRouteTypeDirectNoToken — Specifies ISCC strategy type `direct-notoken`.
- XRouteTypeDNISPooling — Specifies ISCC strategy type `dnis-pool`.
- XRouteTypeDirectDigits — Specifies ISCC strategy type `direct-digits`.
- XRouteTypePullBack — Specifies ISCC strategy type `pullback`.
- XRouteTypeExtProtocol — Specifies ISCC strategy type defined by ISCC parameters.
- XRouteTypeRouteUUI — Specifies ISCC strategy type `route-uui`.
- XRouteTypeMAX — For internal use only.