CSTA

CSTADLL
Version 2.3.4
CSTADLL
Reference Manual

Objective Systems, Inc. version 2.3.4 February 2021

Author's Contact Information

Comments, suggestions, and inquiries regarding CSTADLL may be submitted via electronic mail to info@obj-sys.com.

Contents

Chapter 1

CSTABERPhase1

CSTABERPhase1 is a Microsoft .NET 4.5 DLL that allows client code to communicate with a PBX device.

The DLL uses the following namespaces:

- `Com.Objsys.Csta.Devices`
- `Com.Objsys.Csta.Phase1`

The `Com.Objsys.Csta.Devices` namespace contains classes that allow a caller to use specific PBX devices.

The `Com.Objsys.Csta.Phase(n)` namespaces contain classes that are specific to the indicated phase. Most of these classes are generated by ASN1C from the CSTA and ACSE ASN.1 specifications. These generated classes are not documented here, but you can consult the ASN1C C# User Guide for information about how ASN.1 constructions are translated into C# classes.

Each namespace also contains several classes that are not generated by ASN1C. These classes are the ones documented in this manual.

A typical way to use the DLL is to use the `PBXSession` class to set up the communication to the PBX device via the constructor. If the PBX will be sending asynchronous data, such as monitor packets, to the client, the `ClientCallback` or `XMLClientCallback` property can be used to define a callback method to receive the asynchronous data. If no callback method is defined, asynchronous data will be ignored.

If the PBX will be sending Call Detail Records Report or Call Detail Records Notification messages to the client, the `CDRCallback` or `XMLCDRCallback` property can be used to define a callback method to receive the messages. If no callback method is defined, Call Detail messages will be ignored.

The CSTADLL kit includes some samples to guide you in writing your own code. The samples are evenly split between those implemented in C# and those implemented in Visual BASIC. Each language has samples for communicating with PBX devices that use BER CSTA and with PBX devices that use XML CSTA.

The classes and methods exposed by the DLL are probably sufficient to handle operations for most PBX devices. But if needed, you can write a class of your own to handle operations for a PBX device that the software doesn't explicitly support. The sample NewPBX shows how this might be accomplished. This sample contains code for a small separate DLL that could be used to support a fictitious PBX device. The assumption in the sample is that this device uses standard messages for all operations except for the initial association messages. These messages are the ones that are most commonly different from one PBX to the next. The NewPBX sample shows how the EncodeACSEConnectionRequest() method within the GenericCSTAp2 class (for BER PBX devices) or the EncodeStartSession() method within the GenericXMLEd4 class (for XML PBX devices) can be overridden in a class that you can write. The override implementation handles the details that are specific to the device.

The DLL can log message traffic between a client program and the PBX device if so desired. The logging is controlled by the `LoggingEnabled` property with the `PBXSessionHelper` class. The logging is off by default. Both of

the provided sample clients enable the logging. The log file used is named `cstadll_<program>.log`, where <program> is the name of the executable image that is using the DLL. The location of the log file is the folder where the executable image resides. The default behavior is that if the log file grows to more then 5 Mb, it is copied to `cstadll_<program>.backup.log`, and a new log file is opened. If there is already a file with the backup file name, it is overwritten. That default size of 5 Mb can be modified by using the `MaxLogFileSize` property of the `PBXSessionHelper` class.

If your CSTADLL kit is licensed (i.e., not unlimited), then you will need to deploy your application with the DLLs Reprise.dll and rlm1212.dll that are in the kit. The file rlm1212.dll is a 32-bit native DLL as opposed to a .NET DLL. As such, if you build your code with a Makefile, you will need to use the `/platform:x86` qualifier to the `csc` or `vbc` command. If you build your code with a Visual Studio project, you will need to use x86 as the target platform instead of AnyCPU. These steps are to ensure proper interfacing to the native 32-bit rlm1212.dll. There is also a 64-bit version of rlm1212.dll available if you prefer to target the x64 platform.

Chapter 2

Namespace Documentation

2.1 Package Com.Objsys.Csta.Devices

Classes

- class AlcatelOXO
- class PhilipsSopho
- class SiemensHicom300
- class SiemensRealitis
- class TadiranCoral

2.1.1 Detailed Description

The namespace `Com.Objsys.Csta.Devices` contains classes that allow a caller to use specific PBX devices. The caller does not need to know what CSTA phase a device uses unless the device can accept messages formatted according to rules from more than one CSTA phase. In that case the class name ends with 'p(n)', where (n) is the number of the phase.

2.2 Package Com.Objsys.Csta.Phase1

Classes

- class Constants
- class CSTAContext
- class CSTAResponseInfo
- class GenericCSTAp1
- class IETF_CSTAp1
- class LicenseException
- class LicenseOptions
- class PBXSession
- class PBXSessionException
- class PBXSessionHelper
- class PBXSessionHelperPhase1
- class Phase1Opcodes
- class ROSEParseInfo
- class SocketState

2.2.1 Detailed Description

The namespace `Com.Objsys.Csta.Phase1` contains classes that are specific to phase 1. Most of these classes are generated by ASN1C from the CSTA and ACSE ASN.1 specifications. These generated classes are not documented here, but you can consult the ASN1C C# User Guide for information about how ASN.1 constructions are translated into C# classes.

The namespace also contains several classes that are not generated by ASN1C. These classes are the ones documented in this manual.

Chapter 3

Class Documentation

3.1 AlcatelOXO Class Reference

Inherits Com::Objsys::Csta::Phase1::IETF_CSTAp1.

Public Member Functions

- AlcatelOXO (PBXSession sessionObject)
- AlcatelOXO (string pbxSystem, int port)
- override CSTAResponseInfo MakeACSEAssociation ()

Protected Member Functions

- override int EncodeACSEConnectionRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer)

3.1.1 Detailed Description

Implements CSTA phase 1 operations for the Alcatel OXO device.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 AlcatelOXO (string *pbxSystem*, int *port*)

Constructs an instance associated with the given PBX identifier and port.

Parameters

pbxSystem Well-known name or IP address of the PBX.

port Port on which the PBX listens for CSTA messages.

3.1.2.2 AlcatelOXO (PBXSession *sessionObject*)

Constructs an instance associated with the given PBXSession object.

Parameters

sessionObject A PBXSession object.

3.1.3 Member Function Documentation

3.1.3.1 override int EncodeACSEConnectionRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*) `[protected, virtual]`

Encodes an ACSE Association Request message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

Returns

The length of the encoded message, or -1 if an error occurred.

Reimplemented from GenericCSTAp1.

3.1.3.2 override CSTAResponseInfo MakeACSEAssociation () `[virtual]`

Establish an ACSE association with the PBX.

Returns

A CSTAResponseInfo object.

Reimplemented from GenericCSTAp1.

3.2 Constants Class Reference

Public Types

- enum ACSEMessageTypes
- enum CallbackInvocationMechanisms
- enum CommunicationTypes
- enum Encoding
- enum PBXModels
- enum XMLSessionMessageTypes

Public Attributes

- const long MAX_LOGFILE_SIZE = (5 ∗ 1024 ∗ 1024)

3.2.1 Detailed Description

The Constants class contains some helpful constant and enum definitions.

3.2.2 Member Enumeration Documentation

3.2.2.1 enum ACSEMessageTypes

Provides symbolic names for the ACSE message types.

3.2.2.2 enum CallbackInvocationMechanisms

Indicates how an asynchronous callback method should be invoked. This setting influences how the asynchronous callback methods for monitor event report messages, route messages, and Call Detail Record messages are invoked.

The value InvokeCallbackThenPostNextRead causes the callback method to be invoked before the next read from the PBX or UA is posted to the socket. This setting is the default. With this mechanism callback methods can be easily debugged because new packets from the PBX or UA won't be arriving while debugging of the method is in progress. This mechanism also ensures that messages from the PBX or UA will arrive in a predictable order.

The value PostNextReadThenInvokeCallback causes the callback method to be invoked after the next read from the PBX or UA is posted to the socket. Use of this mechanism is necessary if additional synchronous CSTA messages are going to be sent as part of a callback method's processing. If this mechanism is not used in such a case, the response to the CSTA message sent from the callback method will never be seen because no read to the socket was posted. With that said, however, use this mechanism with EXTREME caution. Because the read to the socket is posted before the event is handled, event n+1 may come in and get handled before event n. You may need to add code to ensure that events get handled in an expected order, if such code is even possible for your situation.

3.2.2.3 enum CommunicationTypes

Provides symbolic names for different ways of communicating with a PBX or UA. The values of this enum influence how each message exchange with a PBX or UA is handled.

3.2.2.4 enum Encoding

Provides symbolic names for the mechanisms for encoding CSTA messages.

3.2.2.5 enum PBXModels

Provides symbolic names for different PBX models.

3.2.2.6 enum XMLSessionMessageTypes

Provides symbolic names for the XML session management message types.

3.2.3 Member Data Documentation

3.2.3.1 const long MAX_LOGFILE_SIZE = (5 ∗ 1024 ∗ 1024)

Defines the maximum size, in bytes, that a log file is allowed to grow to before a new log file is opened.

3.3 CSTAContext Class Reference

Properties

- byte[ ] ResponseFromPBX `[get, set]`
- List< byte[ ]> ResponsesFromPBX `[get, set]`
- string XMLResponseFromPBX `[get, set]`
- List< string > XMLResponsesFromPBX `[get, set]`

3.3.1 Detailed Description

The CSTAContext class contains information needed to manage the interaction between the thread and the PBX.

3.3.2 Property Documentation

3.3.2.1 byte [ ] ResponseFromPBX `[get, set]`

See documentation for CSTAResponseInfo.ResponseFromPBX.

3.3.2.2 List<byte[ ]> ResponsesFromPBX `[get, set]`

See documentation for CSTAResponseInfo.ResponsesFromPBX.

3.3.2.3 string XMLResponseFromPBX `[get, set]`

See documentation for CSTAResponseInfo.XMLResponseFromPBX.

3.3.2.4 List<string> XMLResponsesFromPBX `[get, set]`

See documentation for CSTAResponseInfo.XMLResponsesFromPBX.

3.4 CSTAResponseInfo Class Reference

Properties

- byte[ ] ResponseFromPBX `[get, set]`
- List< byte[ ]> ResponsesFromPBX `[get, set]`
- List< byte[ ]> ResponsesFromUA `[get, set]`
- int StatusCode `[get, set]`
- string StatusMessage `[get, set]`
- string XMLResponseFromPBX `[get, set]`
- string XMLResponseFromUA `[get, set]`
- List< string > XMLResponsesFromPBX `[get, set]`

3.4.1 Detailed Description

Contains information about a PBX operation that was attempted.

3.4.2 Property Documentation

3.4.2.1 byte [ ] ResponseFromPBX `[get, set]`

Contains the response from the PBX for messages that generate a single atomic response, or the immediate acknowledgement response for messages that generate multiple data responses (e.g., Get Switching Function Devices). If a message that normally generates multiple response segments encounters an error (e.g., the PBX rejects the message), then the single error message returned by the PBX will be in this property; the ResponsesFromPBX property will be null.

For CSTA operations this property is simply a reference to the ResponseFromPBX property of the thread's CSTAContext object. If the value of that property changes, then the value of this property changes.

3.4.2.2 List<byte[ ]> ResponsesFromPBX `[get, set]`

Contains the responses from the PBX for messages that generate multiple response segments (e.g., Get Switching Function Devices). If such a message encounters an error (e.g., the PBX rejects the message), then the single error message returned by the PBX will be in the ResponseFromPBX property; this property will be null. In all cases the first response, which is the acknowledgement message from the PBX, will be in the ResponseFromPBX property.

This property is simply a reference to the ResponsesFromPBX property of the thread's CSTAContext object. If the value of that property changes, then the value of this property changes.

3.4.2.3 List<byte[ ]> ResponsesFromUA `[get, set]`

Contains the responses from the UA for messages that generate multiple response segments (e.g., Get Switching Function Devices). If such a message encounters an error (e.g., the UA rejects the message), then the single error message returned by the UA will be in the ResponseFromUA property; this property will be null. In all cases the first response, which is the acknowledgement message from the UA, will be in the ResponseFromUA property.

This property is simply a reference to the ResponsesFromPBX property of the thread's CSTAContext object. If the value of that property changes, then the value of this property changes.

3.4.2.4 int StatusCode `[get, set]`

A numeric status code. A value less than zero indicates that something went wrong during the attempted operation.

3.4.2.5 string StatusMessage `[get, set]`

Text containing information about a PBX operation that has completed, either successfully or not.

3.4.2.6 string XMLResponseFromPBX `[get, set]`

Contains the response from the PBX for messages that generate a single atomic XML response, or the immediate XML acknowledgement response for messages that generate multiple XML data responses (e.g., GetSwitchingFunctionDevices). If a message that normally generates multiple response segments encounters an error (e.g., the PBX rejects the message), then the single error message returned by the PBX will be in this property; the ResponsesFromPBX property will be null.

For CSTA operations this property is simply a reference to the XMLResponseFromPBX property of the thread's CSTAContext object. If the value of that property changes, then the value of this property changes.

3.4.2.7 string XMLResponseFromUA `[get, set]`

Contains the response from the UA for messages that generate a single atomic XML response, or the immediate XML acknowledgement response for messages that generate multiple XML data responses (e.g., GetSwitchingFunctionDevices). If a message that normally generates multiple response segments encounters an error (e.g., the UA rejects the message), then the single error message returned by the UA will be in this property; the ResponsesFromUA property will be null.

For CSTA operations this property is simply a reference to the XMLResponseFromPBX property of the thread's CSTAContext object. If the value of that property changes, then the value of this property changes.

3.4.2.8 List<string> XMLResponsesFromPBX `[get, set]`

Contains the responses from the PBX for messages that generate multiple XML response segments (e.g., GetSwitchingFunctionDevices). If such a message encounters an error (e.g., the PBX rejects the message), then the single error message returned by the PBX will be in the ResponseFromPBX property; this property will be null. In all cases the first response, which is the acknowledgement message from the PBX, will be in the ResponseFromPBX property.

This property is simply a reference to the XMLResponsesFromPBX property of the thread's CSTAContext object. If the value of that property changes, then the value of this property changes.

3.5 GenericCSTAp1 Class Reference

Inherited by SiemensHicom300, and IETF_CSTAp1.

Public Member Functions

- virtual CSTAResponseInfo AnswerCall (ConnectionID callToAnswer, string deviceToLift)
- virtual CSTAResponseInfo AnswerCall (ConnectionID callToAnswer)
- virtual CSTAResponseInfo AnswerCall (string deviceToLift)
- virtual CSTAResponseInfo ClearConnection (ConnectionID connToClear)
- virtual CSTAResponseInfo ClearDoNotDisturb (string targetDevice)
- virtual CSTAResponseInfo ClearMessageWaiting (string targetDevice)
- virtual CSTAResponseInfo ConsultationCall (ConnectionID existingCall, string targetDevice)
- virtual CSTAResponseInfo DivertCall (ConnectionID callToDivert, string divertTo)
- virtual CSTAResponseInfo DivertCall (string divertFrom, string divertTo)
- int EncodeROSERequestHeader (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, Phase1Opcodes.Opcodes opcode)
- GenericCSTAp1 (PBXSession sessionObject)
- GenericCSTAp1 (string pbxSystem, int port)
- virtual CSTAResponseInfo MakeACSEAssociation ()
- virtual CSTAResponseInfo MakeCall (string callingDevice, string calledDevice)
- virtual CSTAResponseInfo MonitorStart (ConnectionID callToMonitor)
- virtual CSTAResponseInfo MonitorStart (string deviceToMonitor)
- virtual CSTAResponseInfo MonitorStop (string monitoredDevice)
- virtual CSTAResponseInfo MonitorStop (MonitorCrossRefID crossRefID)
- virtual CSTAResponseInfo MonitorStop (int crossRefInt)
- virtual CSTAResponseInfo QueryDevice (string deviceToQuery, QueryDeviceFeature feature)
- virtual CSTAResponseInfo QueryDevice (string deviceToQuery)
- virtual CSTAResponseInfo ReleaseACSEAssociation ()
- virtual CSTAResponseInfo RetrieveCall (ConnectionID callToRetrieve)
- virtual CSTAResponseInfo SetDoNotDisturb (string targetDevice)
- virtual CSTAResponseInfo SetMessageWaiting (string targetDevice)
- virtual CSTAResponseInfo SnapshotDevice (string deviceToSnapshot)
- virtual CSTAResponseInfo TransferCall (string heldDevice, string connectedDevice)
- virtual CSTAResponseInfo TransferCall (ConnectionID initiatedCall, ConnectionID originalCall)

Protected Member Functions

- virtual int EncodeACSEConnectionRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer)
- virtual int EncodeACSEReleaseRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer)
- virtual int EncodeAnswerCallRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, ConnectionID callToAnswer, string deviceToLift)
- virtual int EncodeAnswerCallRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, ConnectionID callToAnswer)
- virtual int EncodeAnswerCallRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, string deviceToLift)
- virtual int EncodeClearConnectionRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, ConnectionID connToClear)

- virtual int EncodeConsultationCallRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, ConnectionID existingCall, string targetDevice)
- virtual int EncodeDivertCallRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, ConnectionID callToDivert, string divertTo)
- virtual int EncodeDivertCallRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, string divertFrom, string divertTo)
- virtual int EncodeMakeCallRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, string callingDevice, string calledDevice)
- virtual int EncodeMonitorStartRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, ConnectionID callToMonitor)
- virtual int EncodeMonitorStartRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, string deviceToMonitor)
- virtual int EncodeMonitorStopRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, MonitorCrossRefID crossRefObj)
- virtual int EncodeMonitorStopRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, int crossRefInt, out MonitorCrossRefID crossRefID)
- virtual int EncodeQueryDeviceRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, string deviceToQuery, QueryDeviceFeature feature)
- virtual int EncodeQueryDeviceRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, string deviceToQuery)
- virtual int EncodeRetrieveCallRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, ConnectionID callToRetrieve)
- virtual int EncodeSetOrClearDNDRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, string targetDevice, bool dndOn)
- virtual int EncodeSetOrClearMWRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, string targetDevice, bool indicatorOn)
- virtual int EncodeSnapshotDeviceRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, string deviceToSnapshot)
- virtual int EncodeTransferCallRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, string heldDevice, string connectedDevice)
- virtual int EncodeTransferCallRequest (CSTAResponseInfo response, Asn1BerEncodeBuffer encodeBuffer, ConnectionID initiatedCall, ConnectionID originalCall)

Properties

- PBXSession SessionObject `[get]`
- CSTAContext ThreadContext `[get]`

3.5.1 Detailed Description

Implements CSTA phase 1 operations using BER. Note that most PBXes don't support all CSTA messages, so some methods in this class may result in an error status being returned by your PBX.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 GenericCSTAp1 (string *pbxSystem*, int *port*)

Constructs an instance associated with the given PBX identifier and port.

Parameters

pbxSystem Well-known name or IP address of the PBX.

port Port on which the PBX listens for CSTA messages.

3.5.2.2 GenericCSTAp1 (PBXSession *sessionObject*)

Constructs an instance associated with the given PBXSession object.

Parameters

sessionObject A PBXSession object.

3.5.3 Member Function Documentation

3.5.3.1 virtual CSTAResponseInfo AnswerCall (ConnectionID *callToAnswer*, string *deviceToLift*) `[virtual]`

Answers a call.

Parameters

callToAnswer The connection id of the call to answer.

deviceToLift The device (e.g., extension number) that is to answer the call.

Returns

A CSTAResponseInfo object.

3.5.3.2 virtual CSTAResponseInfo AnswerCall (ConnectionID *callToAnswer*) `[virtual]`

Answers a call.

Parameters

callToAnswer The ConnectionID of the call to answer.

Returns

A CSTAResponseInfo object.

3.5.3.3 virtual CSTAResponseInfo AnswerCall (string *deviceToLift*) `[virtual]`

Answers a call.

Parameters

deviceToLift The identification (e.g., phone number) of the device to answer.

Returns

A CSTAResponseInfo object.

3.5.3.4 virtual CSTAResponseInfo ClearConnection (ConnectionID *connToClear*) `[virtual]`

Clears a connection.

Parameters

connToClear The ConnectionID of the connection to clear.

Returns

A CSTAResponseInfo object.

3.5.3.5 virtual CSTAResponseInfo ClearDoNotDisturb (string *targetDevice*) `[virtual]`

Turns off the Do Not Disturb functionality for a phone.

Parameters

targetDevice The device for which the Do Not Disturb functionality is to be turned off.

Returns

A CSTAResponseInfo object.

3.5.3.6 virtual CSTAResponseInfo ClearMessageWaiting (string *targetDevice*) `[virtual]`

Turns off the message waiting indicator on a device's display.

Parameters

targetDevice The device for which the indicator is to be turned off.

Returns

A CSTAResponseInfo object.

3.5.3.7 virtual CSTAResponseInfo ConsultationCall (ConnectionID *existingCall*, string *targetDevice*) `[virtual]`

Instruct the PBX to do a consultation call.

Parameters

existingCall The connection id of the call for which the consultation call will be made.

targetDevice Identifier (e.g., phone number) of the device that is the target of the consultation call.

Returns

A CSTAResponseInfo object.

3.5.3.8 virtual CSTAResponseInfo DivertCall (ConnectionID *callToDivert*, string *divertTo*) `[virtual]`

Diverts a call to a destination.

Parameters

callToDivert ConnectionID of the call to be diverted.

divertTo Identifier (e.g., phone number) of the location to which the call is to be diverted.

Returns

A CSTAResponseInfo object.

3.5.3.9 virtual CSTAResponseInfo DivertCall (string *divertFrom*, string *divertTo*) `[virtual]`

Diverts a call from a source to a destination.

Parameters

divertFrom Identifier (e.g., phone number) of the call to be diverted.

divertTo Identifier (e.g., phone number) of the location to which the call is to be diverted.

Returns

A CSTAResponseInfo object.

3.5.3.10 virtual int EncodeACSEConnectionRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*) `[protected, virtual]`

Encodes an ACSE Association Request message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

Returns

The length of the encoded message, or -1 if an error occurred.

Reimplemented in AlcatelOXO.

3.5.3.11 virtual int EncodeACSEReleaseRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*) `[protected, virtual]`

Encodes an ACSE Release Request message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.12 virtual int EncodeAnswerCallRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, ConnectionID *callToAnswer*, string *deviceToLift*) `[protected, virtual]`

Encodes an AnswerCall message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

callToAnswer The ConnectionID of the call to answer.

deviceToLift The identification (e.g., phone number) of the device to answer.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.13 virtual int EncodeAnswerCallRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, ConnectionID *callToAnswer*) `[protected, virtual]`

Encodes an AnswerCall message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

callToAnswer The ConnectionID of the call to answer.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.14 virtual int EncodeAnswerCallRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, string *deviceToLift*) `[protected, virtual]`

Encodes an AnswerCall message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

deviceToLift The identification (e.g., phone number) of the device to answer.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.15 virtual int EncodeClearConnectionRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, ConnectionID *connToClear*) `[protected, virtual]`

Encodes a ClearConnection message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

connToClear The ConnectionID of the connection to clear.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.16 virtual int EncodeConsultationCallRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, ConnectionID *existingCall*, string *targetDevice*) `[protected, virtual]`

Encodes a ConsultationCall message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

existingCall The connection id of the call for which the consultation call will be made.

targetDevice Identifier (e.g., phone number) of the device that is the target of the consultation call.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.17 virtual int EncodeDivertCallRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, ConnectionID *callToDivert*, string *divertTo*) `[protected, virtual]`

Encodes a DivertCall message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

callToDivert ConnectionID of the call to be diverted.

divertTo Identifier (e.g., phone number) of the device to which the call is to be diverted.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.18 virtual int EncodeDivertCallRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, string *divertFrom*, string *divertTo*) `[protected, virtual]`

Encodes a DivertCall message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

divertFrom Identifier (e.g., phone number) of the device from which the call is to be diverted.

divertTo Identifier (e.g., phone number) of the device to which the call is to be diverted.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.19 virtual int EncodeMakeCallRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, string *callingDevice*, string *calledDevice*) `[protected, virtual]`

Encodes a MakeCall message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

callingDevice Identifier (e.g., phone number) of the device making the call.

calledDevice Identifier (e.g., phone number) of the device being called.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.20 virtual int EncodeMonitorStartRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, ConnectionID *callToMonitor*) `[protected, virtual]`

Encodes a MonitorStart message to monitor a call.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

callToMonitor The call to monitor.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.21 virtual int EncodeMonitorStartRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, string *deviceToMonitor*) `[protected, virtual]`

Encodes a MonitorStart message to monitor a device.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

deviceToMonitor Identifier (e.g., telephone number) of the device to monitor.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.22 virtual int EncodeMonitorStopRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, MonitorCrossRefID *crossRefObj*) `[protected, virtual]`

Encodes a MonitorStop message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

crossRefObj The cross reference id of the monitor request as a MonitorCrossRefID object.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.23 virtual int EncodeMonitorStopRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, int *crossRefInt*, out MonitorCrossRefID *crossRefID*) `[protected, virtual]`

Encodes a MonitorStop message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

crossRefInt The cross reference id of the monitor request as an integer.

crossRefID A reference to a variable of type MonitorCrossRefID that will be populated with a MonitorCrossRefID object that corresponds to the specified integer cross reference id.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.24 virtual int EncodeQueryDeviceRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, string *deviceToQuery*, QueryDeviceFeature *feature*) `[protected, virtual]`

Encodes a QueryDevice message with the feature specified.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

deviceToQuery The identification (e.g., phone number) of the device to query.

feature The feature that identifies what about the device is being requested.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.25 virtual int EncodeQueryDeviceRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, string *deviceToQuery*) `[protected, virtual]`

Encodes a QueryDevice message with deviceInfo as the feature.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

deviceToQuery The identification (e.g., phone number) of the device to query.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.26 virtual int EncodeRetrieveCallRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, ConnectionID *callToRetrieve*) `[protected, virtual]`

Encodes a RetrieveCall message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

callToRetrieve The ConnectionID of the call to retrieve.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.27 int EncodeROSERequestHeader (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, Phase1Opcodes.Opcodes *opcode*)

This method prepends a ROSE header to an already encoded phase 1 CSTA message.

Parameters

response A CSTAResponseInfo object, used to communicate any exception information back to the caller.

encodeBuffer An Asn1BerEncodeBuffer instance containing the already encoded CSTA message.

opcode The opcode enumeration for the operation that the encoded CSTA message describes.

Returns

The length of the encoded message, including both the CSTA message and the ROSE header, or -1 if the encoding fails.

3.5.3.28 virtual int EncodeSetOrClearDNDRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, string *targetDevice*, bool *dndOn*) `[protected, virtual]`

Encodes a SetFeture message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

targetDevice The device for which Do Not Disturb is to be set or cleared.

dndOn If true, DoNotDisturb will be turned on. If false, DoNotDisturb will be turned off.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.29 virtual int EncodeSetOrClearMWRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, string *targetDevice*, bool *indicatorOn*) `[protected, virtual]`

Encodes a SetFeature message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

targetDevice The device for which the message waiting indicator is to be turned on or off.

indicatorOn If true, the message waiting indicator is turned on. If false, the message waiting indicator is turned off.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.30 virtual int EncodeSnapshotDeviceRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, string *deviceToSnapshot*) `[protected, virtual]`

Encodes a SnapshotDevice message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

deviceToSnapshot Identifier (e.g., phone number) of the device for which the snapshot is desired.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.31 virtual int EncodeTransferCallRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, string *heldDevice*, string *connectedDevice*) `[protected, virtual]`

Encodes a TransferCall message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

heldDevice Identifier (e.g., phone number) of the device from which the call is transferred.

connectedDevice Identifier (e.g., phone number) of the device to which the call is transferred.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.32 virtual int EncodeTransferCallRequest (CSTAResponseInfo *response*, Asn1BerEncodeBuffer *encodeBuffer*, ConnectionID *initiatedCall*, ConnectionID *originalCall*) `[protected, virtual]`

Encodes a TransferCall message.

Parameters

response A CSTA ResponseInfo object.

encodeBuffer An encode buffer object into which the message will be encoded.

initiatedCall ConnectionID of the new call initiated by the consultation call. The initiatedCall member of the ConsultationCallResult class, for example, contains this ConnectionID.

originalCall ConnectionID of the original call. The somewhat confusingly named callingDevice member of the MakeCallResult class contains this ConnectionID, as does the establishedConnection member of the EstablishedEvent class.

Returns

The length of the encoded message, or -1 if an error occurred.

3.5.3.33 virtual CSTAResponseInfo MakeACSEAssociation () `[virtual]`

Establish an ACSE association with the PBX.

Returns

A CSTAResponseInfo object.

Reimplemented in AlcatelOXO, and PhilipsSopho.

3.5.3.34 virtual CSTAResponseInfo MakeCall (string *callingDevice*, string *calledDevice*) `[virtual]`

Instruct the PBX to place a call.

Parameters

callingDevice Identifier (e.g., phone number) of the device making the call.

calledDevice Identifier (e.g., phone number) of the device being called.

Returns

A CSTAResponseInfo object.

3.5.3.35 virtual CSTAResponseInfo MonitorStart (ConnectionID *callToMonitor*) `[virtual]`

Issues a MonitorStart request to the PBX to monitor a call.

Parameters

callToMonitor The call to monitor.

Returns

A CSTAResponseInfo object.

3.5.3.36 virtual CSTAResponseInfo MonitorStart (string *deviceToMonitor*) `[virtual]`

Issues a MonitorStart request to the PBX to monitor a device.

Parameters

deviceToMonitor Identifier (e.g., telephone number) of the device to monitor.

Returns

A CSTAResponseInfo object.

3.5.3.37 virtual CSTAResponseInfo MonitorStop (string *monitoredDevice*) `[virtual]`

This method stops all monitors active against the indicated device, regardless of what thread started the monitor. The method will only stop monitors started through the MonitorStart() method.

Parameters

monitoredDevice The monitored device (e.g., extension).

Returns

If no problems are encountered, the method returns a CSTAResponseInfo object containing the response from the PBX for the LAST Monitor Stop message.
If any problems are encountered, the method returns a CSTAResponseInfo object containing information about the error, including any response from the PBX for the problematic Monitor Stop message.

3.5.3.38 virtual CSTAResponseInfo MonitorStop (MonitorCrossRefID *crossRefID*) `[virtual]`

Stop a previously started PBX monitor request.

Parameters

crossRefID The cross reference id of the monitor request as a MonitorCrossRefID object.

Returns

A CSTAResponseInfo object.

3.5.3.39 virtual CSTAResponseInfo MonitorStop (int *crossRefInt*) `[virtual]`

Stop a previously started PBX monitor request.

Parameters

crossRefInt The cross reference id of the monitor request as an integer.

Returns

A CSTAResponseInfo object.

3.5.3.40 virtual CSTAResponseInfo QueryDevice (string *deviceToQuery*, QueryDeviceFeature *feature*) `[virtual]`

Queries a device using the feature specified.

Parameters

deviceToQuery The identification (e.g., phone number) of the device to query.

feature The feature that identifies what about the device is being requested.

Returns

A CSTAResponseInfo object.

3.5.3.41 virtual CSTAResponseInfo QueryDevice (string *deviceToQuery*) `[virtual]`

Queries a device using the deviceInfo feature.

Parameters

deviceToQuery The identification (e.g., phone number) of the device to query.

Returns

A CSTAResponseInfo object.

3.5.3.42 virtual CSTAResponseInfo ReleaseACSEAssociation () `[virtual]`

Releases an ACSE association with a PBX device.

Returns

A CSTAResponseInfo object. For this message the connection with the PBX is closed, so null is returned.

3.5.3.43 virtual CSTAResponseInfo RetrieveCall (ConnectionID *callToRetrieve*) `[virtual]`

Retrieves a held call.

Parameters

callToRetrieve The ConnectionID of the call to retrieve.

Returns

A CSTAResponseInfo object.

3.5.3.44 virtual CSTAResponseInfo SetDoNotDisturb (string *targetDevice*) `[virtual]`

Sets the Do Not Disturb feature for a phone.

Parameters

targetDevice The device for which Do Not Disturb is to be set.

Returns

A CSTAResponseInfo object.

3.5.3.45 virtual CSTAResponseInfo SetMessageWaiting (string *targetDevice*) `[virtual]`

Turns on the message waiting indicator on a device's display.

Parameters

targetDevice The device for which the indicator is to be turned on.

Returns

A CSTAResponseInfo object.

3.5.3.46 virtual CSTAResponseInfo SnapshotDevice (string *deviceToSnapshot*) `[virtual]`

Instruct the PBX to take a snapshot of calls active at a device.

Parameters

deviceToSnapshot Identifier (e.g., phone number) of the device for which the snapshot is desired.

Returns

A CSTAResponseInfo object.

3.5.3.47 virtual CSTAResponseInfo TransferCall (string *heldDevice*, string *connectedDevice*) `[virtual]`

Transfers a call from one device to another.

Parameters

heldDevice Identifier (e.g., phone number) of the device from which the call is transferred.

connectedDevice Identifier (e.g., phone number) of the device to which the call is transferred.

Returns

A CSTAResponseInfo object.

3.5.3.48 virtual CSTAResponseInfo TransferCall (ConnectionID *initiatedCall*, ConnectionID *originalCall*) `[virtual]`

Transfers a call. A consultation call must be done before calling this method.

Parameters

initiatedCall ConnectionID of the new call initiated by the consultation call. This ConnectionID can be mined out of the ConsultationCallResult class.

originalCall ConnectionID of the original call. This ConnectionID can be mined out of the MakeCallResult class or out of a monitor event message.

Returns

A CSTAResponseInfo object.

3.5.4 Property Documentation

3.5.4.1 PBXSession SessionObject `[get]`

The PBXSession object associated with this instance.

3.5.4.2 CSTAContext ThreadContext `[get]`

The CSTAContext structure for this thread.

3.6 IETF_CSTAp1 Class Reference

Inherits Com::Objsys::Csta::Phase1::GenericCSTAp1.

Inherited by AlcatelOXO, PhilipsSopho, SiemensRealitis, and TadiranCoral.

Public Member Functions

- IETF_CSTAp1 (PBXSession sessionObject)
- IETF_CSTAp1 (string pbxSystem, int port)

3.6.1 Detailed Description

Implements CSTA phase 1 operations using IETF encoding, which puts a two-byte length in front of the BER message.

3.6.2 Constructor & Destructor Documentation

3.6.2.1 IETF_CSTAp1 (string *pbxSystem*, int *port*)

Constructs an instance associated with the given PBX identifier and port.

Parameters

pbxSystem Well-known name or IP address of the PBX.

port Port on which the PBX listens for CSTA messages.

3.6.2.2 IETF_CSTAp1 (PBXSession *sessionObject*)

Constructs an instance associated with the given PBXSession object.

Parameters

sessionObject A PBXSession object.

3.7 LicenseException Class Reference

3.7.1 Detailed Description

Defines an exception that occurs while trying to find license information.

3.8 LicenseOptions Class Reference

3.8.1 Detailed Description

This class holds booleans that define what capabilities are defined in the license.

3.9 PBXSession Class Reference

Public Member Functions

- delegate void AsyncCallback (PBXSession sessionObject, byte[ ] asyncData)
- delegate void AsyncExceptionCallback (PBXSession sessionObject, ApplicationException exception)
- void Close (CSTAContext threadContext)
- delegate void ConnectionCallback (PBXSession sessionObject)
- void Open (CSTAContext threadContext)
- PBXSession (string pbxSystem, int port)
- SocketState SendACSEMessage (byte[ ] message, int messageLength, Constants.ACSEMessageTypes messageType, CSTAContext threadContext)
- void SendMessage (string messageType, byte[ ] message, int messageLength, CSTAContext threadContext)
- void SendMessage (byte[ ] message, int messageLength, CSTAContext threadContext)
- void SendXMLMessage (string messageType, string strMessage, CSTAContext threadContext)
- void SendXMLMessage (string strMessage, CSTAContext threadContext)
- SocketState SendXMLSession (string strMessage, Constants.XMLSessionMessageTypes enmMessageType, CSTAContext threadContext)
- void WaitForROSEResponse (CSTAContext threadContext)
- void WaitForXMLResponse (CSTAContext threadContext)
- delegate void XMLAsyncCallback (PBXSession sessionObject, string message)

Properties

- Constants.CallbackInvocationMechanisms CallbackInvocationMechanism `[get, set]`
- AsyncCallback CDRCallback `[get, set]`
- AsyncCallback ClientCallback `[get, set]`
- bool Connected `[get, set]`
- ConnectionCallback ConnectionLostCallback `[get, set]`
- bool DebugMode `[get, set]`
- AsyncExceptionCallback ExceptionCallback `[get, set]`
- int MaxReceiveTimeout `[get, set]`
- Constants.Encoding MessageEncoding `[get, set]`
- string PBXSystem `[get]`
- int Port `[get]`
- AsyncCallback SystemStatusCallback `[get, set]`
- XMLAsyncCallback XMLCDRCallback `[get, set]`
- XMLAsyncCallback XMLClientCallback `[get, set]`
- XMLAsyncCallback XMLSystemStatusCallback `[get, set]`

3.9.1 Detailed Description

This class manages communication with a PBX. One instance of this class should be created for each PBX with which a CSTADLL client application needs to exchange CSTA messages.

The CSTA worker classes (e.g., Alcatel4400, PanasonicNCP) hold a reference to a PBXSession object. If the constructor for the worker class that takes a PBX identification and a PBX port is used, a PBXSession object is created. Alternatively, the client application can create a PBXSession instance and pass a reference to the instance to the other worker class constructor signature.

Only one PBXSession instance for a PBX/port combination should be created. The behavior is undefined if multiple PBXSession instances are created for the same PBX and port.

3.9.2 Constructor & Destructor Documentation

3.9.2.1 PBXSession (string *pbxSystem*, int *port*)

Constructs a PBXSession object.

Parameters

pbxSystem The name or IP address of the PBX system.

port The port on the PBX system to which the client is connecting.

3.9.3 Member Function Documentation

3.9.3.1 delegate void AsyncCallback (PBXSession *sessionObject*, byte[ ] *asyncData*)

Declaration of a callback function to be invoked when one of the following messages is received: a monitor event report message, a route message, a CDR Report message, a CDR Notification message, or a system status request.

Parameters

sessionObject The session object for the PBX that generated the asynchronous message.

asyncData The data received asynchronously from the PBX.

3.9.3.2 delegate void AsyncExceptionCallback (PBXSession *sessionObject*, ApplicationException *exception*)

Declaration of a callback function to be invoked if a condition is encountered in the asynchronous I/O handler that would otherwise result in an exception being thrown. Note that in a couple of cases the asynchronous code will still throw an exception, even if this callback is defined.

Parameters

sessionObject The session object for the PBX that sent a packet that triggered an exception condition.

exception The ApplicationException object that would have been thrown in the asynchronous I/O handling code if this callback were not defined.

3.9.3.3 void Close (CSTAContext *threadContext*)

Terminates the session to the PBX. This method can be used to terminate sessions with PBX devices that don't accept ACSE release association requests.

Parameters

threadContext The context object for the caling thread.

3.9.3.4 delegate void ConnectionCallback (PBXSession *sessionObject*)

Declaration of a callback function to be invoked if the connection to the PBX is lost.

Parameters

sessionObject The session object for the PBX whose connection was lost.

3.9.3.5 void Open (CSTAContext *threadContext*)

This method can be used to establish communication with a PBX device before any messages are actually sent to the device. TCP/IP connectivity is established and an asynchronous read is started to receive messages sent from the PBX.

Parameters

threadContext The thread context object.

3.9.3.6 SocketState SendACSEMessage (byte[ ] *message*, int *messageLength*, Constants.ACSEMessageTypes *messageType*, CSTAContext *threadContext*)

This method sends an ACSE message (either Make Association or Release Association) to the PBX and receives the response. This operation is done synchronously. If the Make Association needs to be done (usually it does), it must be done before any threads for sending and receiving CSTA messages are started.

This method is only intended to be used by client code that encodes its own ACSEMakeAssociation or ACSEReleaseAssociation message. Most clients can probably use the MakeACSEAssociation() and ReleaseACSEAssociation() methods that are in each phase's helper classes.

Parameters

message An encoded ACSE Make Association or Release Association message.

messageLength The length of the encoded message.

messageType A constant telling whether the message is an ACSE Make Association or an ACSE Release Association.

threadContext The thread context object.

Returns

A populated SocketState instance.

3.9.3.7 void SendMessage (string *messageType*, byte[ ] *message*, int *messageLength*, CSTAContext *threadContext*)

This method sends a message to the PBX using TCP/IP.

Parameters

messageType A string token to help identify the message in the CSTADLL log file.

message Byte array containing the encoded message to send.

messageLength The length of the encoded message.

threadContext The thread context object.

3.9.3.8 void SendMessage (byte[ ] *message*, int *messageLength*, CSTAContext *threadContext*)

This method sends a message to the PBX using TCP/IP.

Parameters

message Byte array containing the encoded message to send.

messageLength The length of the encoded message.

threadContext The thread context object.

3.9.3.9 void SendXMLMessage (string *messageType*, string *strMessage*, CSTAContext *threadContext*)

This method sends an XML message to the PBX using TCP/IP.

Parameters

messageType A string token to help identify the message in the CSTADLL log file.

strMessage The XML message to send.

threadContext The thread context object.

3.9.3.10 void SendXMLMessage (string *strMessage*, CSTAContext *threadContext*)

This method sends an XML message to the PBX using TCP/IP.

Parameters

strMessage The XML message to send.

threadContext The thread context object.

3.9.3.11 SocketState SendXMLSession (string *strMessage*, Constants.XMLSessionMessageTypes *enmMessageType*, CSTAContext *threadContext*)

This method sends an XML session management (ECMA-354) message to the PBX.

Parameters

strMessage The text of the XML message to send.

enmMessageType A constant indicating what kind of session management message is being sent.

threadContext The thread context object.

Returns

A populated SocketState instance if the message is a StartSession message. Null if the message is StopSession or ResetSession.

3.9.3.12 void WaitForROSEResponse (CSTAContext *threadContext*)

This method waits for a response to a CSTA message sent with a ROSE header.

Parameters

threadContext The CSTAContext object associated with the calling thread.

3.9.3.13 void WaitForXMLResponse (CSTAContext *threadContext*)

This method waits for a response to an XML CSTA message.

Parameters

threadContext The CSTAContext object associated with the calling thread.

3.9.3.14 delegate void XMLAsyncCallback (PBXSession *sessionObject*, string *message*)

Declaration of a callback function to be invoked when an asynchronous XML monitor event or route messae is received.

Parameters

sessionObject The session object for the PBX or UA that generated the asynchronous message.

message The text of the message received asynchronously from the PBX or UA.

3.9.4 Property Documentation

3.9.4.1 Constants.CallbackInvocationMechanisms CallbackInvocationMechanism `[get, set]`

Indicates what asynchronous callback invocation mechanism to use. See the documentation on Constants.CallbackInvocationMechanisms for a detailed description of the possible options.

3.9.4.2 AsyncCallback CDRCallback `[get, set]`

Holds a reference to an asynchronous callback function that will be invoked when CDR information is received asynchronously. This information could take the form of a CDR Report message or a CDR Notification message. If the latter, the application should use the SendStoredCDR() method to request the stored CDRs from the PBX.

3.9.4.3 AsyncCallback ClientCallback `[get, set]`

Holds a reference to an asynchronous callback function. This function will be invoked if a monitor event or route message is received asynchronously from the PBX.

3.9.4.4 bool Connected `[get, set]`

Indicates whether the session to the PBX is connected.

3.9.4.5 ConnectionCallback ConnectionLostCallback `[get, set]`

Holds a reference to an asynchronous callback function. This function will be invoked if the connection to the PBX is lost.

3.9.4.6 bool DebugMode `[get, set]`

Enables behavior that facilitates debugging of the CSTADLL software. This property is most likely useful only to Objective Systems staff.

3.9.4.7 AsyncExceptionCallback ExceptionCallback `[get, set]`

Holds a reference to an asynchronous callback function. This function will be invoked if a condition occurs in the asynchronous I/O handler that otherwise would have resulted in an exception being thrown if this callback were not defined. Note that in a couple of cases the asynchronous code will still throw an exception, even if this callback is defined.

3.9.4.8 int MaxReceiveTimeout `[get, set]`

Specifies the amount of time, in milliseconds, to wait for a respone to arrive from the PBX. The default value is 5,000 milliseconds (5 seconds).

3.9.4.9 Constants.Encoding MessageEncoding `[get, set]`

Indicates how messages exchanges with this PBX are encoded.

3.9.4.10 string PBXSystem `[get]`

The TCIP/IP address or well-known name of the PBX.

3.9.4.11 int Port `[get]`

The port where the PBX listens for CSTA messages.

3.9.4.12 AsyncCallback SystemStatusCallback `[get, set]`

Holds a reference to an asynchronous callback function. This function will be invoked if a BER System Status message is received asynchronously from the PBX. The client application does NOT need to send a System Status Response, since that is taken care of by CSTADLL.

3.9.4.13 XMLAsyncCallback XMLCDRCallback `[get, set]`

Holds a reference to an asynchronous callback function that will be invoked when XML CDR information is received asynchronously.

This information could take the form of a CDR Report message or a CDR Notification message. If the latter, the application should use the SendStoredCDR() method to request the stored CDRs from the PBX or UA.

3.9.4.14 XMLAsyncCallback XMLClientCallback `[get, set]`

Holds a reference to an asynchronous XML callback function. This function will be invoked if an XML monitor event report message or route message is received asynchronously from the PBX or UA.

3.9.4.15 XMLAsyncCallback XMLSystemStatusCallback `[get, set]`

Holds a reference to an asynchronous callback function. This function will be invoked if an XML System Status message is received asynchronously from the PBX. The client application does NOT need to send a System Status Response, since that is taken care of by CSTADLL.

3.10 PBXSessionException Class Reference

3.10.1 Detailed Description

Defines an exception that occurs while communicating with a PBX.

3.11 PBXSessionHelper Class Reference

Properties

- static bool LoggingEnabled `[get, set]`
- static string LoggingFolder `[get, set]`
- static long MaxLogFileSize `[get, set]`

3.11.1 Detailed Description

This class holds static properties that affect all PBX sessions.

3.11.2 Property Documentation

3.11.2.1 bool LoggingEnabled `[static, get, set]`

Indicates whether logging should be done.

3.11.2.2 string LoggingFolder `[static, get, set]`

Specifies a folder to receive the log file. If not specified, the log file will go into whatever folder the calling .exe resides in.

3.11.2.3 long MaxLogFileSize `[static, get, set]`

Defines the maximum size, in bytes, that a log file is allowed to grow to before a new log file is opened. If no value is specified for this property, the maximum size is Constants.MAX_LOG_FILE_SIZE. Any value specified for this property overrides this default setting.

3.12 PBXSessionHelperPhase1 Class Reference

3.12.1 Detailed Description

This class contains utility methods used by PBXSessionHelper for BER phase 1.

3.13 Phase1Opcodes Class Reference

Public Types

- enum Opcodes

3.13.1 Detailed Description

This class contains a public enum that contains symbolic names for the opcodes that define CSTA phase 1 operations.

3.13.2 Member Enumeration Documentation

3.13.2.1 enum Opcodes

Contains symbolic names for the opcodes that define CSTA phase 1 operations.

3.14 PhilipsSopho Class Reference

Inherits Com::Objsys::Csta::Phase1::IETF_CSTAp1.

Public Member Functions

- CSTAResponseInfo MakeACSEAssociation (bool testMode)
- override CSTAResponseInfo MakeACSEAssociation ()
- PhilipsSopho (PBXSession sessionObject)
- PhilipsSopho (string pbxSystem, int port)

3.14.1 Detailed Description

Implements CSTA phase 1 operations for the Philips Sopho PBX device.

3.14.2 Constructor & Destructor Documentation

3.14.2.1 PhilipsSopho (string *pbxSystem*, int *port*)

Constructs an instance associated with the given PBX identifier and port.

Parameters

pbxSystem Well-known name or IP address of the PBX.

port Port on which the PBX listens for CSTA messages.

3.14.2.2 PhilipsSopho (PBXSession *sessionObject*)

Constructs an instance associated with the given PBXSession object.

Parameters

sessionObject A PBXSession object.

3.14.3 Member Function Documentation

3.14.3.1 CSTAResponseInfo MakeACSEAssociation (bool *testMode*)

Establish an ACSE association with the PBX.

Parameters

testMode True to send a test mode message, false to send a licensed message.

Returns

A CSTAResponseInfo object.

3.14.3.2 override CSTAResponseInfo MakeACSEAssociation () `[virtual]`

Establish a licensed (i.e., not test mode) ACSE association with the PBX.

Returns

A CSTAResponseInfo object.

Reimplemented from GenericCSTAp1.

3.15 ROSEParseInfo Class Reference

3.15.1 Detailed Description

Contains information about a received CSTA message obtained by decoding the message's ROSE header.

3.16 SiemensHicom300 Class Reference

Inherits Com::Objsys::Csta::Phase1::GenericCSTAp1.

Public Member Functions

- SiemensHicom300 (PBXSession sessionObject)
- SiemensHicom300 (string pbxSystem, int port)

3.16.1 Detailed Description

Implements CSTA phase 1 operations for the Siemens Hicom 300 PBX device.

3.16.2 Constructor & Destructor Documentation

3.16.2.1 SiemensHicom300 (string *pbxSystem*, int *port*)

Constructs an instance associated with the given PBX identifier and port.

Parameters

pbxSystem Well-known name or IP address of the PBX.

port Port on which the PBX listens for CSTA messages.

3.16.2.2 SiemensHicom300 (PBXSession *sessionObject*)

Constructs an instance associated with the given PBXSession object.

Parameters

sessionObject A PBXSession object.

3.17 SiemensRealitis Class Reference

Inherits Com::Objsys::Csta::Phase1::IETF_CSTAp1.

Public Member Functions

- SiemensRealitis (PBXSession sessionObject)
- SiemensRealitis (string pbxSystem, int port)

3.17.1 Detailed Description

Implements CSTA phase 1 operations for the Siemens Realitis PBX device.

3.17.2 Constructor & Destructor Documentation

3.17.2.1 SiemensRealitis (string *pbxSystem*, int *port*)

Constructs an instance associated with the given PBX identifier and port.

Parameters

pbxSystem Well-known name or IP address of the PBX.

port Port on which the PBX listens for CSTA messages.

3.17.2.2 SiemensRealitis (PBXSession *sessionObject*)

Constructs an instance associated with the given PBXSession object.

Parameters

sessionObject A PBXSession object.

3.18 SocketState Class Reference

Properties

- byte[ ] AckBuffer `[get, set]`
- byte[ ] ReadBuffer `[get, set]`
- List< byte[ ]> ReadBuffers `[get, set]`
- int TotalLength `[get, set]`

3.18.1 Detailed Description

This class contains the response received from the PBX and state information about the exchange with the PBX that is used internally by CSTADLL.

3.18.2 Property Documentation

3.18.2.1 byte [ ] AckBuffer `[get, set]`

Contains the first response from the PBX for situations where the PBX sends multiple response messages (e.g., Get Switching Function Devices). The data messages that are sent after this ack will be in ReadBuffers.

3.18.2.2 byte [ ] ReadBuffer `[get, set]`

Contains the bytes most recently read from the socket. This buffer will be filled in bit by bit as the message is read.

3.18.2.3 List<byte[ ]> ReadBuffers `[get, set]`

Contains multiple collections of bytes read from the socket. This array is used for situations where a response to a message comes in multiple segments (e.g., Get Switching Function Devices). For these situations the immediate response will be in AckBuffer.

3.18.2.4 int TotalLength `[get, set]`

The total length of a complete message received from the PBX. This is also used as an offset into the read buffer so we can build the message as it's received.

3.19 TadiranCoral Class Reference

Inherits Com::Objsys::Csta::Phase1::IETF_CSTAp1.

Public Member Functions

- TadiranCoral (PBXSession sessionObject)
- TadiranCoral (string pbxSystem, int port)

3.19.1 Detailed Description

Implements CSTA phase 1 operations for the Tadiran Coral PBX device.

3.19.2 Constructor & Destructor Documentation

3.19.2.1 TadiranCoral (string *pbxSystem*, int *port*)

Constructs an instance associated with the given PBX identifier and port.

Parameters

pbxSystem Well-known name or IP address of the PBX.

port Port on which the PBX listens for CSTA messages.

3.19.2.2 TadiranCoral (PBXSession *sessionObject*)

Constructs an instance associated with the given PBXSession object.

Parameters

sessionObject A PBXSession object.

Index