CoAP Provider

CoAP (Constrained Application Protocol)

Version 1.0.0

User's guide

May 15, 2015

[Remarks]

[Revision History]

Version	Date	Contents
1.0.0	2015-05-15	First edition

[Supported devices]

Model	Version	Description

Contents

1. Introduction

This document is a user’s guide of CAO provider that communicates through CoAP (Constrained Application Protocol). CAO provider (CaoProvCoAP.dll) described in this document is called CoAP provider.

The next chapter shows the overview of CoAP, and Chapter 3 shows the command reference.

2. Overview of CoAP provider

2.1. Overview

CoAP provider is a CAO provider that establishes CoAP communication through UDP Ethernet communication. The file format is DLL (Dynamic Link Library) and it is dynamically uploaded from CAO engine when it is used. To use CoAP provider, you need to install ORiN2SDK, or, complete registration manually based on the information on Table 2-1.

Table 2-1 CoAP provider

File name	CaoProvCoAP.dll
ProgID	CaoProv.IETF.CoAP
Registry registration	regsvr32 CaoProvCoAP.dll
Delete registry registration	regsvr32 /u CaoProvCoAP.dll

2.2. Mode

CoAP provider has two operation modes (server mode and client mode) and two communication modes (synchronous mode and asynchronous mode). The following three types of combination are available.

2.2.1. Server mode

If a computer is in server mode, it starts running as soon as receiving data from the communication partner. A computer in server mode cannot spontaneously send data to the communication partner. In server mode, communication mode is fixed to the asynchronous mode.

In server mode, computer issues an OnMessage event when it receives data from the communication partner. Response data is send by CaoMessage::Reply() of CaoMessage object that is obtained by OnMessage event.

2.2.2. Client mode

If a computer is in client mode, it spontaneously sends data to the communication partner. The difference between the synchronous mode and asynchronous mode is whether the computer sends data to the communication partner.

・**Synchronous mode**

In this mode, a computer waits the response after sending data to the communication partner. To send data, use CaoCommand::Execute(), CaoVariable::get_Value(), or CaoVariable::put_Value() commands.

・**Asynchronous mode**

In this mode, a computer does not wait the response after sending data to the communication partner. To send data, use CaoController::Execute() command. Any responses from the communication partner will be ignored.

2.3. Method and Property

2.3.1. CaoWorkspace::AddController method

CoAP provider establishes communication by referring the communication connection parameters when AddController is executed.

At that time, specify option items (communication style, connection parameter, and timeout).

Syntax AddController(<bstrCtrlName:BSTR>,<bstrProvName:BSTR>, <bstrPcName:BSTR > [,<bstrOption:BSTR>])

bstrCtrlName	: [in] controller name
bstrProvName	: [in] provider name. Fixed value= ” CaoProv.IETF.CoAP”
bstrPcName	: [in] computer name where provider operates.
bstrOption	: [in] option character string

The following shows the list of option character string and description.

Table 2-2 Option character string of CaoWorkspace::AddController

Option	Description
Conn=<connection parameter>	Mandatory. Enter a connection parameter. For details, see 2.3.1.1.
Timeout=<timeout >	Send/Receive timeout (default: 5000 msec.)
CoAPClient [=TRUE / FALSE]	Enter an operation mode. TRUE represents client mode, FALSE represents server mode. (default: FALSE)
Sync [=TRUE / FALSE]	Set the communication mode. TRUE represents synchronous mode, FALSE represents asynchronous mode. If the operation mode is server mode, this entry is fixed to asynchronous mode. (default: FALSE)
PacketOpt [=<packet parameter>]	Set the communication packet (default: “0:0:0”) (See 2.3.1.2)
SendRetry=<Retry count>	Retry count at data sending. 0 to 7 (default: 5) If the value is 0 or less, it is treated as 0. If the value is 7 or more, it is treated as 7.
RecvRetry=<Retry count>	Retry count at data reception. 0 to 7(default: 5) If the value is 0 or less, it is treated as 0.

	If the value is 7 or more, it is treated as 7.

2.3.1.1. Conn option

The following shows the connection parameter character strings of Conn option. Items enclosed by square brackets are omittable.

・**Server mode**

"udp[:<Source Address>[:<Source Port>]]"

<Source Address>	:	When two or more NICs are used, specify an IP address of the server mode with this option. If this entry is omitted, an IP address will be selected automatically. If an IP address that is not assigned to a local machine is selected, an error occurs. (default: 255.255.255.255)
<Source Port>	:	Server port number (default: 5683)

・**Client mode**

"udp[:<Destination Address>[:<Destination Port>[:<Source Address>[:<Source Port>]]]]"

<Destination Address>	:	Server IP address (default: 127.0.0.1)
<Destination Port>	:	Server port number (default: 5683)
<Source Address>	:	When two or more NICs are used, specify an IP address of the client mode with this option. If this entry is omitted, an IP address will be selected automatically. If an IP address that is not assigned to a local machine is selected, an error occurs. (default: 255.255.255.255)
<Source Port>	:	Client port number (default: 0)

2.3.1.2. PacketOpt option

The following shows the parameter character strings of PacketOpt option. Items enclosed by square brackets are omittable. The underlined value in each parameter description represents the default value which will be used when any option is not specified.

"PacketOpt =[<Convert>[:<URIEncode>[:<Space>]]]"

<Convert>	:	Payload data conversion 0:b-CAP mode

<URIEncode>	:	Replace all blank characters in URI in a character specified by <Space>. The first bit : Replace blank characters of URI path The second bit :Replace blank characters of URI query The third bit : Replace blank character of URI host (default: 0)
<Space>	:	Specify a character which is replaced with all blank characters in URI. 0: Keep a blank as is. 1:Replace a blank with ”+”.

2.3.2. CaoController::AddCommand method

Once AddCommand method is invoked, CaoCommand object is obtained. CaoCommand object can be obtained under the synchronous mode only. For arguments in AddCommand method under CaoController class, specifies a command name (BSTR type). Command name specified here is an arbitrary character string and there is no restriction. For instance, you can specify AddCommand ("Cmd1").

Syntax AddCommand(<bstrName:BSTR> [,<bstrOption:BSTR>])

bstrName	:	[in]	Command name
bstrOption	:	[in]	Option character string

2.3.3. CaoController::AddVariable method

Once AddVariable method is invoked, CaoVariable object is obtained. CaoVariable object can be obtained under the synchronous mode only. For arguments in AddVariable method under CaoController class, specify a variable name (BSTR type). “Variable name” specified here will be used as a URI path of CoAP. For example, if you specify AddVariable ("Var1"), “/Var1” will be informed to the communication partner as a URI path.

Syntax AddVariable(<bstrName:BSTR> [,<bstrOption:BSTR>])

bstrName	:	[in]	Variable name
bstrOption	:	[in]	Option character string

2.3.4. CaoController::Execute method

Execute a command. CaoController::Execute method is available only in the asynchronous mode. For arguments in Execute method, specify a command with BSTR type and a parameter with VARIANT array.

Syntax [<vntRet:VARIANT> =] Execute(<bstrCmd:BSTR> [,<vntParam:VARIANT>])

bstrCmd	:	[in]	Command
vntParam	:	[in]	Parameter

vntRet	[out]	Return value

For about parameters required by command execution and the obtainment results, refer to Chapter 3.

2.3.5. CaoController::OnMessage event

This event does not occur under the synchronous mode.

Once a CoAP provider under the server mode receives data from a communication partner, OnMessage event in CaoController class will occur.

2.3.6. CaoCommand::get_Parameters property

Obtain the parameter that is currently set in the CaoCommand object.

2.3.7. CaoCommand::put_Parameters property

Set the parameter to CaoCommand object.

Parameters correspond with Option and Payload of CoAP.

For how to specify parameters, refer to "Appendix A Specifying Option and Payload.

2.3.8. CaoCommand::get_Result property

Obtain the execution result of the latest CaoCommand::Execute method.

2.3.9. CaoCommand::Execute method

Execute a command.

For arguments in Execute method, specify a command with LONG type. This command corresponds with Code of CoAP. For about the transmittable Code list of CoAP, refer to Table 3-2.

Syntax Execute(<lMode:LONG>)

lMode	:	[in]	command

2.3.10. CaoVariable::get_Value property

Issue a request of Code ("Get") of CoAP to a communication partner.

2.3.11. CaoVariable::put_Value property

Issue a request of Code ("Put") of CoAP to a communication partner.

2.3.12. CaoMessage::get_Number property

This corresponds with Type and Code of CoAP that is received from a communication partner. The upper four bytes represent Type, and the lower four bytes represent Code.

2.3.13. CaoMessage::get_Value property

This corresponds with Payload of CoAP that is received from a communication partner.

2.3.14. CaoMessage::get_Destination property

This corresponds with Option of CoAP that is received from a communication partner.

2.3.15. CaoMessage::Reply method

Inform a communication partner of the processing result of OnMessage event.

Syntax Reply(< vntData:VARIANT>)

vntData	:	[in]	Response data

Specify a response data with the following format.

vntData	:	Response data (VT_VARIANT | VT_ARRAY: 1 or 2 elements) [0] Execution result (VT_I4) Specify 0 if an event completes normally. Specify a corresponding HRESULT if an event completes abnormally. [1] Return value (VT_VARIANT) Specify this option when you need to send a result data to a communication partner after an event processing. Enter this option only when necessary.

3. Command reference

3.1. Command list

Table 3-1 Command list of CaoController::Execute method

Category	Command	Description	
	Get	Request to obtain a specified resource	P. 11
	Post	Request to process a specified resource	P. 11
	Put	Request to set a specified resource	P. 12
	Delete	Request to delete a specified resource	P. 12

3.2. Details of Commands

Get

Syntax `object.Get <Data>`

Argument `<Data>` = VT_VARIANT

This corresponds with Option and Payload of CoAP. For information on how to specify parameter, refer to "Appendix. A Specifying Option and Payload.

Return value None

Description Issue a request to obtain a specified resource.

Post

Syntax `object.Post <Data>`

Argument `<Data>` = VT_VARIANT

This corresponds with Option and Payload of CoAP. For information on how to specify parameter, refer to "Appendix. A Specifying Option and Payload".

Return value None

Description Issue a request to process a specified resource.

Put

Syntax `*object*.Put <Data>`

Argument `<Data>` = VT_VARIANT

This corresponds with Option and Payload of CoAP. For information on how to specify parameter, refer to “Appendix. A　Specifying Option and Payload.

Return value None

Description Issue a request to set a specified resource.

Delete

Syntax `*object*.Delete <Data>`

Argument `<Data>` = VT_VARIANT

This corresponds with Option and Payload of CoAP. For information on how to specify parameter, refer to “Appendix. A　Specifying Option and Payload”.

Return value None

Description Issue a request to delete a specified resource.

Appendix A. Specifying Option and Payload

CoAP provider enables to define Option and Payload of CoAP with one VARIANT type data as the table below shows.

Data(VARIANT)

1. Specify one Option	(VT_VARIANT | VT_ARRAY:2 elements)
	[0] Value of Option
	[1] Option data
	(Refer to Table 3-2)
2. Specify one Option	(VT_VARIANT | VT_ARRAY:3 elements)
＋Payload	[0] Value of Option
	[1] Option data
	[2] Payload(VT_VARIANT)
3. Specify multiple Options	(VT_VARIANT | VT_ARRAY: n elements)
	[0..n] Format 1.(VT_VARIANT ｜ VT_ARRAY)
4. Specify multiple Options	(VT_VARIANT | VT_ARRAY:2 elements)
＋Payload	[0] Format 3.(VT_VARIANT | VT_ARRAY)
	[1] Payload(VT_VARIANT)

Appendix B. CoAP parameter list

The following shows frequently used parameters in CoAP. For details, refer to http://coap.technology/.

Table 3-2 CoAP parameter list

Parameter	Name	Value	Description
Code	GET	1	Request to obtain the specified resource
	POST	2	Request to process the specified resource
	PUT	3	Request to set the specified resource
	DELETE	4	Request to delete the specified resource
Option	Uri-Host	3	Specify the host of URI. (VT_BSTR)
	Uri-Port	7	Specify the port number of URI. (VT_BSTR)
	Uri-Path	11	Specify the path of URI. (VT_BSTR)
	Content-Format	12	Specify the format of content. (VT_UI4)
	Uri-Query	15	Specify URI's Query. (VT_BSTR)