IoTDS provider

IoT Data Share / IoT Data Server

Version 1.3.0

User's guide

October 28, 2024

[remarks]

This document is translated from Japanese into English by the machine translation.

[version up history]

Version	Date	Contents
1.0.0	2017-10-31	A first edition.
1.1.0	2018-02-01	I revise a problem at the time of the local connection.
1.2.0	2019-09-24	Project file database support. Added project search order for $TREE_LIST$. Bug fixes.
1.2.1	2020-11-01	Fixed $TREE_LIST$ processing that would result in incorrect retrieval if the project name contained non-alphanumeric characters. Memory-related processing fixes. Overall process corrected.
1.2.2	2021-03-26	Fixed to be output as an error when writing to a read-only item.
1.2.3	2021-04-01	Fixed the spelling of error messages when local connection fails.
1.2.4	2021-08-19	Fixed processing at startup and termination. Fixed reconnection processing. Fixed memory related processing. Fixed exclusive processing. Support for asynchronous message processing.
	2021-12-20	Corrected typo. OpenSSL version upgrade.
	2023-10-09	Added how to set IoTDS. Added explanation of CaoVariable::get_ID and CaoVariable::put_ID. Added default port number at AddController. Added sample program.
1.3.0	2024-10-28	Added the UseCB option and CBPort option.

[Operation check model]

Model	Version	Instructions

Table of contents

1. Preface

IoTDS provider is ORiN2 CAO provider performing the note / reading of data for IoT Data Share, IoT Data Server.

I illustrate by this document about the summary of the IoTDS provider and implemented CAO interface (function specifications).

2. Summary of the provider

2.1. Summary

The IoTDS provider is a CAO provider that writes and reads data to IoT Data Share or IoT Data Server.

It is possible to select IoT Data Share on the local machine or IoT Data Share / IoT Data Server on the network as the connection destination.

When connecting to IoT Data Share / IoT Data Server on the network, b-CAP is used and the IoTDS provider operates as a b - CAP client. For this reason, the b - CAP server must be active at the connection destination.

If connecting to IoT Data Share on the local machine, make a direct connection.The file format is a DLL (Dynamic Link Library), which is dynamically loaded from the CAO engine when used. To use the IoTDS provider, you need to install ORiN 2 SDK or manually register registry by referring to the table below.

Table 2-1 IoTDS provider

File name	CaoProvDENSOIoTDS.dll
ProgID	CaoProv.DENSO.IoTDS
Registry registration	regsvr32 CaoProvDENSOIoTDS.dll
Erasion of the registry registration	regsvr32 /u CaoProvDENSOIoTDS.dll

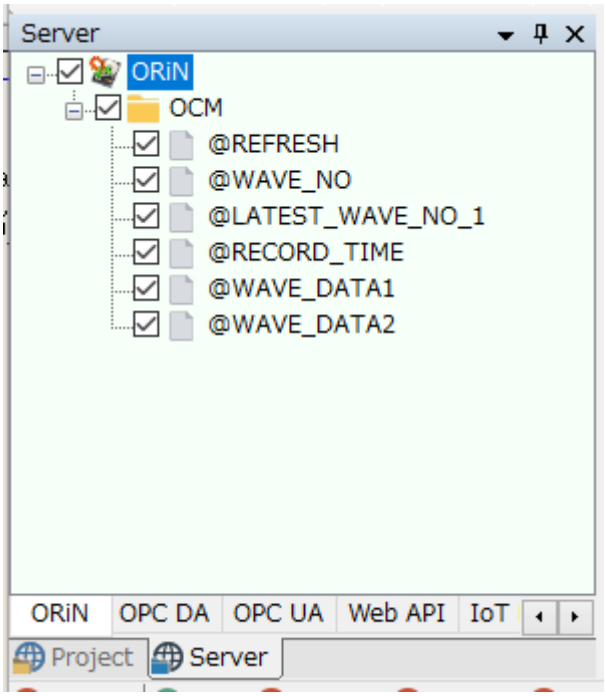
The IoTDS provider provides the following functions for controllers set in the project of IoT Data Share or IoT Data Server to be connected.

- Command execution to the controller
- Obtaining information on items
- Acquisition / setting of values for items

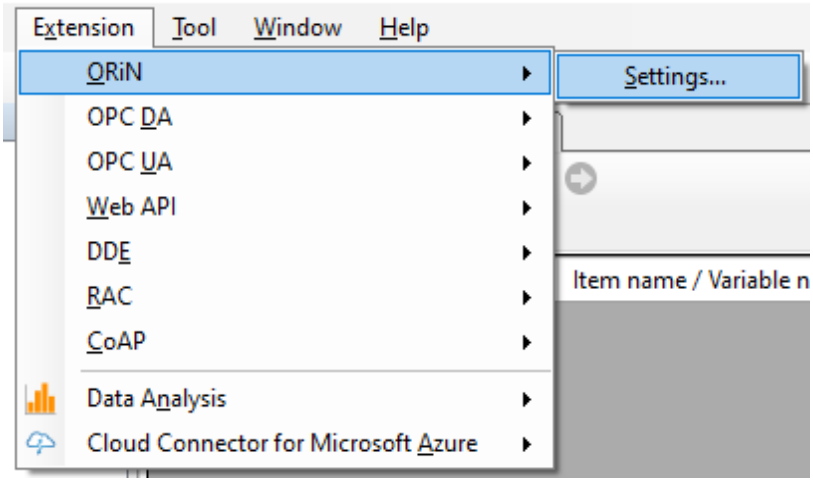
2.2. Publishing settings in IoT Data Share/IoT Data Server

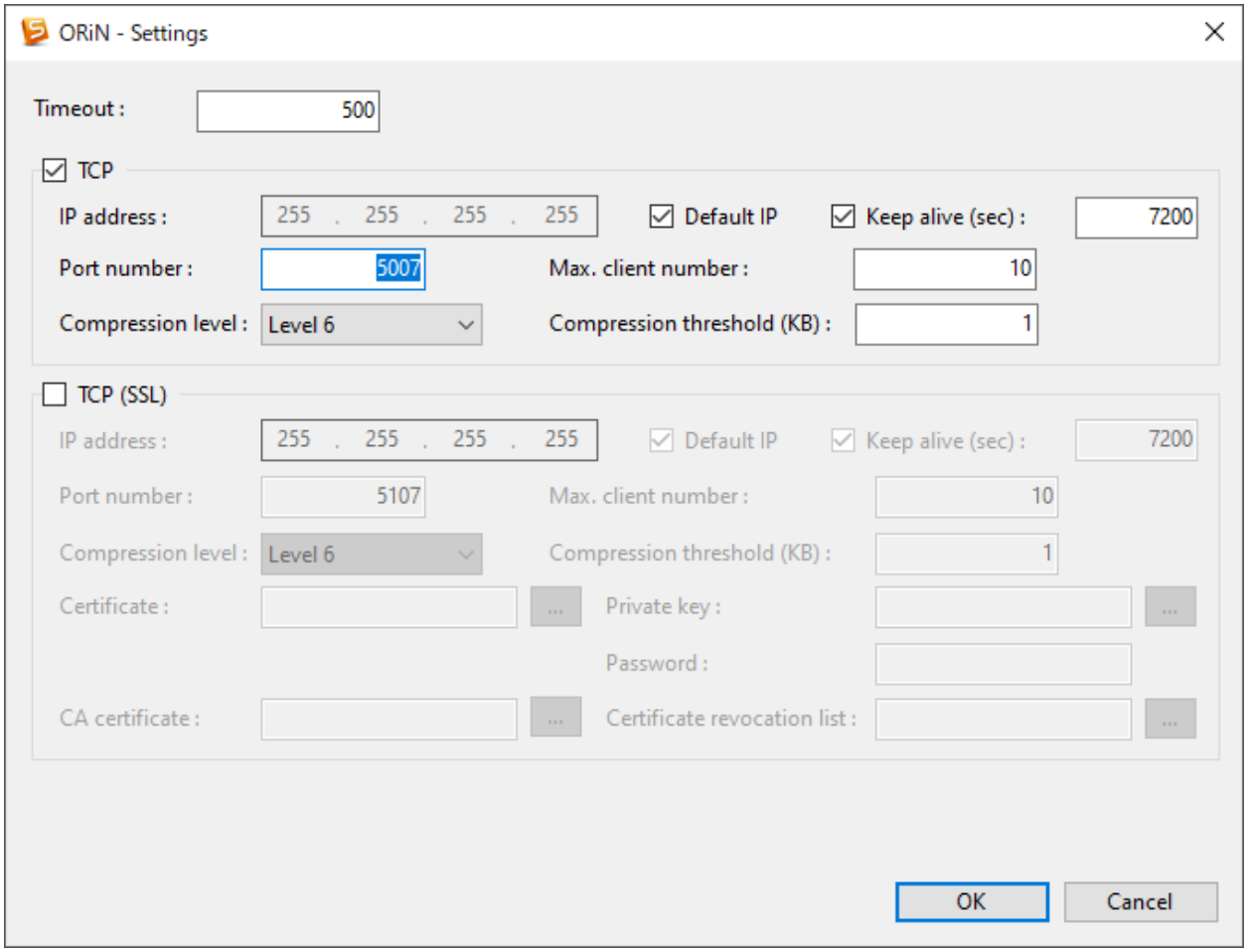
In order for IoTDS providers to write/read data to/from IoT Data Share, or IoT Data Server, they must be configured to expose it in IoT Data Share, or IoT Data Server.

By selecting Server/ORiN tab in IoT Data Share/IoT Data Server tree view and checking the controllers and items for which data is to be written/read, data can be written/read from IoTDS providers.



You can also change the port number on which IoT Data Share/IoT Data Server listens for connections in the configuration dialogs that appear when you select the menu's Extension-ORiN-settings.




ORiN - Settings
Timeout :
500
TCP
IP address :
255 . 255 . 255 . 255
Default IP
Keep alive (sec) :
7200
Port number :
5007
Max. client number :
10
Compression level :
Level 6
Compression threshold (KB) :
1
TCP (SSL)
IP address :
255 . 255 . 255 . 255
Default IP
Keep alive (sec) :
7200
Port number :
5107
Max. client number :
10
Compression level :
Level 6
Compression threshold (KB) :
1
Certificate :
Private key :
Password :
CA certificate :
Certificate revocation list :
OK
Cancel

2.3. Method property

2.3.1. CaoWorkspace::AddController method

The IoTDS provider connects to IoT Data Share and IoT Data Server when generating a Controller object.

In addition, the generated Controller object corresponds to the controller of the project.

Format AddController(<bstrCtrlName:BSTR>,<bstrProvName:BSTR>,
<bstrPCName:BSTR>,<bstrOption:BSTR>))

bstrCtrlName : [in] controller name
bstrProvName : [in] provider name. Fixation level = "CaoProv.DENSO.IoTDS"
bstrPcName : [in] Name of the provider's execution machine
bstrOption : [in] option string

I show a list to appoint below for optional character string.

Table 2-2 CaoWorkspace :: AddController option string

<table>
<tr><th>Option [1)]</th><th>Explanation</th></tr>
<tr><td>Controller
[= <Controller name of project>]</td><td>Name of the controller registered in the project of connection destination IoT Data Share and IoT Data Server
(Default: the controller name specified by the first argument of AddController)
The following reservation names can also be used as controller names.
<table>
<tr><th>Controller name</th><th>Explanation</th></tr>
<tr><td>$ENGINE$</td><td>With CaoController::get_VariableNames, you can obtain a list of controller names registered in IoT Data Share and IoT Data Server.
If you create a CaoController object with this controller name, you can not execute except get_VariableNames, Execute.</td></tr>
<tr><td>$TREE_LIST$</td><td>With CaoController::get_VariableNames, you can acquire structure information of controllers and items registered in IoT Data Share and IoT</td></tr>
</table>
</td></tr>
</table>

[1] The inside of square brackets ("[]") indicates that it can be omitted. The underlined part in each parameter's explanation is the default value when no option is specified.

	Data Server. If you create a CaoController object with this controller name, you can not execute except get_VariableNames, Execute.
Server[=<IP address> [:<port number>]]	Specify the IP address and port number of IoT Data Share or IoT Data Server to be connected remotely. The port number is optional; if it is omitted, 5007 is used as the port number. If you omit this option, you will connect to the local IoT Data Share.
MyIP[=< local machine IP address>]	The IP address on the local machine. (Multiple NIC applications) (Default: Unspecified)
Timeout[=< time-out time>]	Timeout time for transmission / reception. (millisecond) (Default: 5000)
UseCB[=TRUE/FALSE]	Specifies whether or not to use a callback port for receiving events. If used, the port number is specified with the CBPort option. TRUE: Use the callback port. FALSE: Do not use the callback port. (Default: TRUE)
CBPort[=call back port number]	Specifies whether to use a callback port for receiving events. If used, the port number is specified with the CBPort option. TRUE: Use the callback port. FALSE: Do not use the callback port. (Default: TRUE) Specifies the port number to be used in the goalback for receiving events. If 0 is specified, the port number will be dynamically assigned. This option is used when the UseCB option is specified as TRUE. (Default: 0)

```
AddController
(
    "<Controller name>",            // The controller name of the provider.
    "CaoProv.DENSO.IoTDS",          // Provider name. (Fixed)
    "",
    "[[Controller=<Controller name>][, Server=<IP address of connected IoT Data Share]"
        // Name of registered Controller, IP address of connection destination IoT Data Share.
)
```

(example 1)

```
AddController
(
    "IoTDSCtrl1",                   // Use IoTDSCtrl1for provider's controller name
   "CaoProv.DENSO.IoTDS",           // Provider name. (Fixed)
    "",
    "Controller=RC1"
      // Use the registered controller RC1, connect to the local machine's IoT Data Share
)
```

(example 2)

```
AddController
(
    "IoTDSCtrl1",                   // Use IoTDSCtrl1for provider's controller name
   "CaoProv.DENSO.IoTDS",           // Provider name. (Fixed)
    "",
    "Controller=RC1,Server=192.168.0.1"
      // Use registered controller RC 1, connect to IoT Data Share on the network (192.168.0.1)
)
```

2.3.2. CaoController::AddVariable method

In the IoTDS provider, it corresponds to the items registered in the controller of IoT Data Share and IoT Data Server when generating the Variable object. Therefore, for the variable name specified in AddVariable, enter the item name of the specified controller.

```
AddVariable
(
   "<Item name of controller>" // variable name of provider = item name of designated controller
)
```

2.3.3. CaoController::Execute

The contents executed by this method depend on the controller name specified at AddController. Detailed contents of each controller name are shown below.

(1) In the case of $ENGINE$

I can execute the following commands.

Command name	Argument	Return value	Summary
$CSQP_GET_COUNT$	None	Number of controllers registered in the project	Get the number of controllers registered in the project of IoT Data Share and IoT Data Sever

(2) For other than $ENGINE$

You can execute the following command.

Command name	Argument	Return value	Summary
$CSQP_GET_COUNT$	None	Number of items registered in controller	IoT Data Share and IoT Data Sever acquire the number of items registered in the specified controller
$CSQP_GET_STATE$	None	Controller status status 0: Uninit 1: Active 2: Inactive 3: Error 4: Terminated	Get status status of controller specified by IoT Data Share and IoT Data Sever

2.3.4. CaoController::get_Help property

Get the Description property of the controller.

2.3.5. CaoController::get_ID property

Gets the Index property of the controller.

2.3.6. CaoController::get_Tag property

Get the Tag property of the controller.

2.3.7. CaoController::put_Tag property

Set the Tag property of the controller.

2.3.8. CaoController::get_VariableNames property

If the controller name is "$ TREE_LIST $", you can get the controller structure information and items registered in IoT Data Share and IoT Data Server. At this time, the acquired IoT Data Share projects are detected in the following order.

- Running project
- Default project registered by IoT Data Server or IoT Data Share Manager
- Last executed project

When the controller name is "$ ENGINE $", we obtain a list of controllers registered in IoT Data Share and IoT Data Server.

In the case of other controller names, we obtain a list of items registered in the controller specified by AddController.

2.3.9. CaoVariable::get_Attribute property

Gets the attribute value of the item. The attribute values to be acquired are as follows.

The table 2-3 Available attribute values

Attribute level	Meaning
0	Read/Write
1	ReadOnly
2	WriteOnly

2.3.10. CaoVariable::get_DateTime property

Retrieve the item update date and time.

2.3.11. CaoVariable::get_Help property

Gets the Description property of the item.

2.3.12. CaoVariable::get_ID property

Gets ID property-value of the item.

The item's ID property accesses CAO's Variable property of CAO that the item holds (or ID property of the parent item's CAO that it holds for subitems).

Generally, for providers that do not implement ID property of Variable of CAO, accessing ID property results in an unimplemented error (0x80004001).

If ID property of an item (virtual item, aliased item, or array type item) that does not have a Variable of CAO is accessed, an 0x80004005 failure occurs.

2.3.13. CaoVariable::put_ID property

Set ID property-value of the item.

2.3.14. CaoVariable::get_Microsecond property

Get microseconds of item update date / time.

2.3.15. CaoVariable::get_Tag property

Gets the Tag property of the item.

2.3.16. CaoVariable::put_Tag property

Sets the Tag property of the item.

2.3.17. CaoVariable::get_Value property

Gets the value of the item.

2.3.18. CaoVariable::put_Value property

Set a value for the item.

2.4. Error code

The IoTDS provider defines the following unique error codes. For the ORiN 2 common error, refer to the error code chapter of "ORiN 2 Programming Guide".

List of table 2-4 original error codes

Error name	Error number	Explanation
E_CAOP_NO_CAOSQL	0x80100000	Project is not started.

3. Sample program

Here is an example that retrieves the valuefrom a SampleItem item below SampleController controllers of a IoT Data Share running on a localhost.

List 3-1 Form1.vb

```
Imports ORiN2.ManagedCAO

Public Class Form1
    Private eng As CCaoEngine = Nothing
    Private ctrl As CCaoController = Nothing
    Private variable As CCaoVariable = Nothing

    Private Sub DisposeEngine()
        If Not IsNothing(eng) Then
            eng.Dispose()
            eng = Nothing
        End If
    End Sub

    Private Sub BtnConnect_Click(sender As Object, e As EventArgs) Handles BtnConnect.Click
        If Not IsNothing(eng) Then Return
        Try
            eng = New CCaoEngine()
            ctrl = eng.Workspaces(0).AddController( _
                "SampleController", _
                "CaoProv.DENSO.IoTDS", _
                "", _
                "Server=127.0.0.1")
            variable = ctrl.AddVariable("SampleItem", "")
        Catch ex As Exception
            MessageBox.Show(Me, ex.ToString(), Text, MessageBoxButtons.OK, MessageBoxIcon.Error)
            DisposeEngine()
        End Try
    End Sub

    Private Sub BtnGetValue_Click(sender As Object, e As EventArgs) Handles BtnGetValue.Click
```

```
        If IsNothing(variable) Then Return
        Dim value As Object = variable.Value
        MessageBox.Show(Me, String.Format("{0}({1})", value, _
                                If(IsNothing(value), "Nothing", value.GetType().Name)))
    End Sub

    Private Sub Form1_FormClosed(sender As Object, e As FormClosedEventArgs) _
        Handles MyBase.FormClosed
        DisposeEngine()
    End Sub
End Class
```