# OPC UA Provider
# Gateway for OPC UA

Version 1.5.0

## User's guide

July 1, 2024

Note:

## [Revision history]

| Version | Date | Contents |
|---|---|---|
| 1.0.0 | 2014-02-10 | First edition |
| 1.0.1 | 2018-11-01 | Memory leak was corrected. |
| 1.0.2 | 2018-11-14 | Bug in reconnection was corrected. |
| 1.0.2 | 2018-12-21 | Added about AE function. |
| 1.0.2 | 2020-07-08 | Corrected the description of VARIANT type data. |
| 1.0.3 | 2020-10-06 | Fixed a bug in type conversion. |
| 1.1.0 | 2021-07-20 | Update of internally used library. |
| 1.2.0 | 2021-08-23 | Supports node ID. |
| 1.2.1 | 2021-08-26 | Fixed a bug in connection settings. |
| 1.2.2 | 2021-09-03 | Obfuscation of source code. |
| 1.3.0 | 2021-11-04 | Added security policy. |
| 1.3.1 | 2022-01-24 | Fixed a bug when specifying a certificate. |
| 1.3.2 | 2022-02-16 | Error output added when writing values with anonymous authentication. |
| 1.3.3 | 2022-03-31 | Fixed a bug when there is no certificate. |
| 1.4.0 | 2022-06-06 | General-purpose event acquisition support for AC function. |
| | 2022-07-07 | Minor revisions to User’s Guide. |
| | 2022-07-29 | Update of internally used library. |
| 1.4.1 | 2022-09-06 | Event data mapping with a structure array. |
| 1.4.2 | 2022-10-18 | Bug fixes for termination processing. |
| 1.4.3 | 2022-10-27 | Bug fixes for event data including NULL. |
| | 2023-11-30 | Error Correction. |
| 1.5.0 | 2024-07-01 | Update of internally used library. |

## [Hardware]

| Model | Version | Note |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

## Contents

1. Introduction ..... 5
2. Outline of the provider ..... 6
2.1. Outline ..... 6
2.2. Method and Property ..... 8
2.2.1. CaoWorkspace::AddController method ..... 8
2.2.2. CaoController::AddVariable method ..... 11
2.2.3. CaoController::get_VariableNames property ..... 14
2.2.4. CaoController::OnMessageEvent ..... 14
2.2.5. CaoVariable::get_DateTime property ..... 17
2.2.6. CaoVariable::get_Value property ..... 17
2.2.7. CaoVariable::put_Value property ..... 17
2.2.8. CaoVariable::get_ID property ..... 17
2.2.9. CaoMessage::Reply method ..... 17
2.3. Error code ..... 18
3. Command Reference ..... 20
3.1. Controller classes ..... 20
3.1.1. CaoController::Execute("GetTypeInfo") Command ..... 20
4. Certificate files ..... 21
4.1. Application certificate ..... 21
4.2. Certificate for the authentication with the certificate ..... 21
4.3. Creating a Certificate ..... 22
5. Sample program ..... 23
5.1. UA Demo Server (manufactured by Softing) ..... 23
5.2. CaoOPCUA ..... 24
5.2.1. Use access path ..... 24
5.2.2. Use node ID ..... 25

# 1. Introduction

This is a user's guide for “CAO provider for OPC UA Connection (hereafter, OPCUA provider)", which establishes a connection between CAO with PLC (Programmable Logic Controller) through the OPC UA (OLE for Process Control Unified Architecture) server.

With this provider, CAO-compatible tools can access a PLC and a display where OPCUA server is equipped, as well as a robot.

This OPCUA provider provides the security function to establish safe connection with the OPCUA server, and the access function to access the information stored in the connected device.

# 2. Outline of the provider

## 2.1. Outline

OPC UA provider is a genuine ORiN provider that connects with CAO of ORiN, and an OPC UA client that supports the OPC UA specification as well (Figure 2-1). This provider establishes connection between CAO with OPCUA-compatible server, providing access not only with robots and PLCs but also OPCUA-compatible various devices.

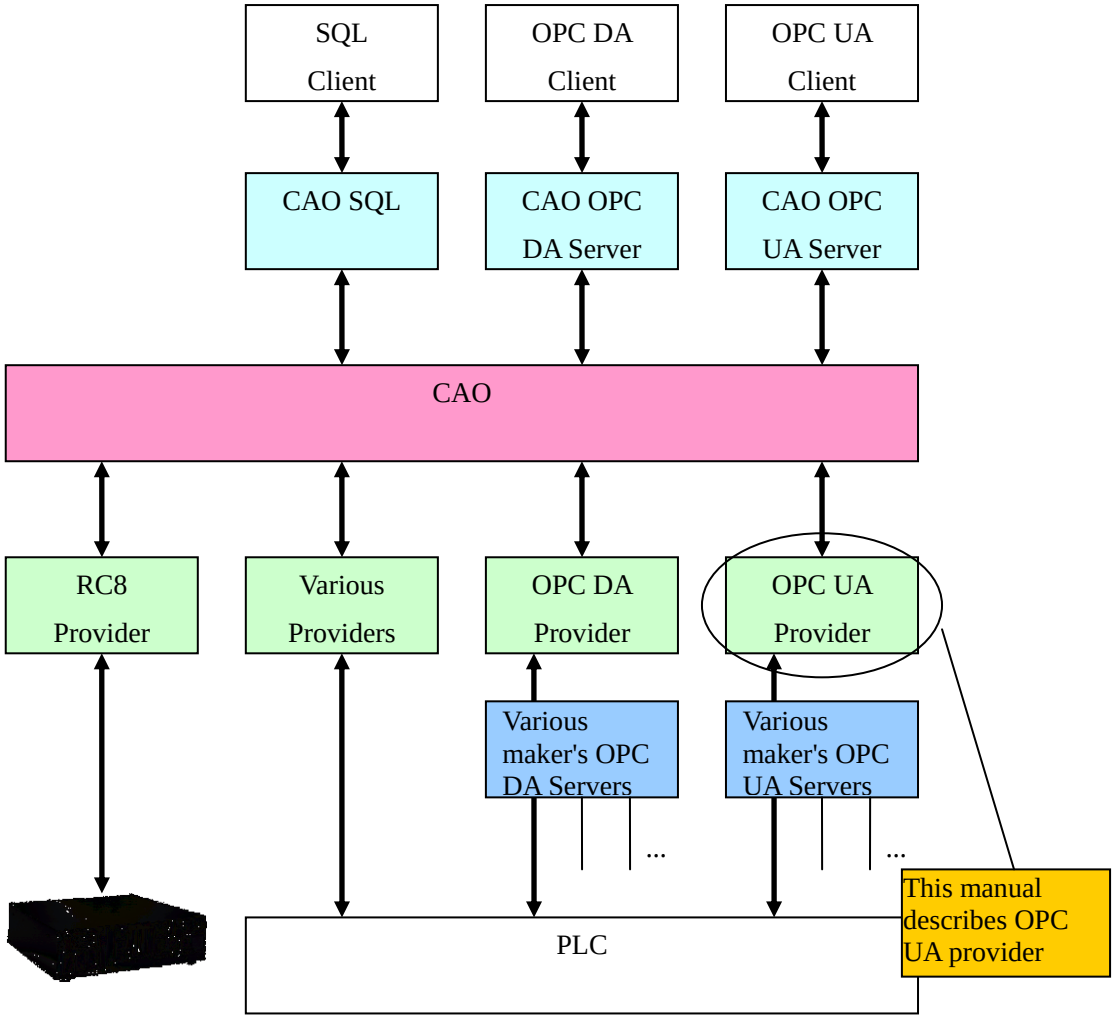



**Figure 2-1 CAO provider for OPC UA connection**

This OPC UA provider allows CaoVariable objects to refer to values of Item retained by OPCUA server. The purpose of OPC DA provider and OPC UA provider are the same but these have no compatibility.

Moreover, the alarm state of the Item value can be observed as a function of AE, and the event be received.

- It corresponds to DA(DataAccess) of OPC UA and AE(Alarm&Event) in this provider.

**Table 2-1 OPC UA provider**

| | |
|---|---|
| File name | CaoProvOPCUA.dll |
| ProgID | CaoProv.OPCUA |
| Registry registration[1] | regsvr32 CaoProvOPCUA.dll |
| Remove registry registration | regsvr32 /u CaoProvOPCUA.dll |

[1] If the registry is installed by ORiN SDK, you do not need to register/delete it manually

## 2.2. Method and Property

### 2.2.1. CaoWorkspace::AddController method

The following shows AddController method for CaoWorkspace class of the CAO

```
AddController
(
 "<Controller name>",                    // controller name
 "CaoProv.OPCUA",                        // Provider name. Fixed.
 "<Machine name>,                        // Machine name that the provider runs.
 "<Option>"                        // Option character string(OPC UA option)
)
```

In this example, <Provider name> is fixed; <Machine name> means other provider.

The following shows the format of <Option>.

**<Option> ::= Server=<OPC UA server URL>[, AccessPath=<Setting value>][, Security=<Setting value>:<Setting value>][, Der=<Setting value>][, Pem=<Setting value>][, Password=<Setting value>][, User=<Setting value>:<Setting value>][, Certificate=<Setting value>][, PrivateKey=<Setting value>]**

Enter the URL of OPC UA server in <OPC UA server URL>. This is prerequisite. Parameters enclosed by "[]" are not prerequisite items, and default values will be used if not specified. The specified value is interpreted based on the OPC UA server being called. For other options, refer to Table 2-2.

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

| | | |
|---|---|---|
| <bstrCtrlName> | : | [in] Controller name(VT_BSTR)<br>Specify an arbitrary character strings |
| <bstrProvName> | : | [in] Provider ProgID(VT_BSTR)<br>Specify a fixed character string of "CaoProv.OPCUA" |
| <bstrPCName> | : | [in] PC name (VT_BSTR)<br>Specify a PC name that is used for the remote connection.<br>Specify null string ("") for the local connection. |
| <bstrOption> | : | [in] Option character string (VT_BSTR)<br>Specify option character strings to ensure the connection.<br>To specify each option, use comma-delimited following format..<br><OptionName>=<Value>,<OptionName>=<Value>,..<br>Example: "Server=opc.tcp://192.168.100.100:4890/CaoOPCUA,<br>AccessPath=OPCUA.CAO/RC8" |

The following table lists options that are specified in the <bstrOption> option character strings.

**Table 2-2 Option character strings of CaoWorkspace::AddController**

<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td>Server=<OPC UA server URL></td><td>URL of OPC UA server [prerequisite]<br>Example: “Server=opc.tcp://192.168.100.100:4890/CaoOPCUA"</td></tr>
<tr><td>AccessPath=[Access path]</td><td>Default access path<br>You can set desired access path individually by AccessPath option of AddVariable. If not, the value specified here will be used.</td></tr>
<tr><td>Security=[security policy [:security mode]]</td><td>The following table lists available security settings. (Default:0:0)<br>Security policy:
<table>
<tr><th>Setting</th><th>value</th><th>Description</th></tr>
<tr><td>None</td><td>0</td><td>No security.</td></tr>
<tr><td>Basic128Rsa15</td><td>1</td><td>Deprecated. This feature is reserved for compatibility.[2]</td></tr>
<tr><td>Basic256</td><td>2</td><td>Deprecated. This feature is reserved for compatibility.[2]</td></tr>
<tr><td>Basic256Sha256</td><td>3</td><td>Average security.</td></tr>
<tr><td>Aes128Sha256RsaOaep</td><td>4</td><td>High security.</td></tr>
<tr><td>Aes256Sha256RsaPss</td><td>5</td><td>Ultra high security.</td></tr>
</table>
Security mode:
<table>
<tr><th>Setting</th><th>value</th><th>Description</th></tr>
<tr><td>None</td><td>0</td><td>No security is set</td></tr>
<tr><td>Sign</td><td>1</td><td>Message has a sign but is not encrypted</td></tr>
<tr><td>SignAndEncrypt</td><td>2</td><td>Message has a sigh and is encrypted</td></tr>
</table>
The table below shows the available combinations between the security policy and the security mode.
<table>
<tr><th>Security policy</th><th>Security mode</th></tr>
<tr><td>None</td><td>None</td></tr>
<tr><td>Basic128Rsa15</td><td>Sign</td></tr>
<tr><td>Basic128Rsa15</td><td>SignAndEncrypt</td></tr>
<tr><td>Basic256</td><td>Sign</td></tr>
<tr><td>Basic256</td><td>SignAndEncrypt</td></tr>
<tr><td>Basic256Sha256</td><td>Sign</td></tr>
<tr><td>Basic256Sha256</td><td>SignAndEncrypt</td></tr>
<tr><td>Aes128Sha256RsaOaep</td><td>Sign</td></tr>
</table></td></tr>
</table>

[2] Encryption algorithm is known to be broken. This policy shall not be used any more.

<table>
<tr><td></td><td>
<table>
<tr><td>Aes128Sha256RsaOaep</td><td>SignAndEncrypt</td></tr>
<tr><td>Aes256Sha256RsaPss</td><td>Sign</td></tr>
<tr><td>Aes256Sha256RsaPss</td><td>SignAndEncrypt</td></tr>
</table>
If the security policy is set to “0", the security mode can be omitted. In this case, the security mode is treated as “0".
</td></tr>
<tr><td>Der=[Security certificate file name]</td><td>Specify a file name for the security certificate file (Default: Null)</td></tr>
<tr><td>Pem=[Security private key file name]</td><td>Specify a file name for the security private key (Default: Null)</td></tr>
<tr><td>Password=[Security password]</td><td>Specify a security password (Default: Null)</td></tr>
<tr><td>User=[Username : Password]</td><td>Specify a username and password for the user authentication. (Default: Null)</td></tr>
<tr><td>Certificate=[User authenticate certificate file name]</td><td>Specify a file name of certificate that is used at the certificate authenticate (Default: Null)</td></tr>
<tr><td>PrivateKey=[User authenticate private key file name]</td><td>Specify a file name of private key that is used at the certificate authenticate (Default: Null)</td></tr>
<tr><td>TrustServer=[<True|False>]</td><td>Specify the behavior when the server certificate does not exist in the certificate trust list. (Default: True)<br>True: Trust and connect<br>False: Not trust this server certificate. Not connect</td></tr>
<tr><td>EventAutoACK=[<True|False>]</td><td>Specify the operation when the condition event of OPCUA is received. (default: True)<br>True: Return the ACK response automatically.<br>False: Do not return the ACK response automatically. (Respond by the Reply method of OnMessage )</td></tr>
<tr><td>EventNodeIds=[((Node ID)[,(Node ID)[…]])]</td><td>Specifies the nodeID of the event whose properties are to be retrieved when CaoMessage is received.<br>Node ID = namespace index, identifier type, identifier<br>Namespace indexes: namespace indexes for node IDs defined on the server.<br>Identifier type: Identifier data type.
<table>
<tr><td>Setting</td><td>value</td><td>Description</td></tr>
<tr><td>Numeric</td><td>0</td><td>Specify the identifier numerically.</td></tr>
<tr><td>String</td><td>1</td><td>Specify the identifier as a character string.</td></tr>
</table>
Identifier: Identifier of the node.<br>You can specify more than one. (Default: Unspecified)</td></tr>
</table>

| | |
|---|---|
| | (See 2.2.2.1) |

Example

```
Dim oCtrl As Object
Set oCtrl = caoWorkspace.AddController("OpcUa","CaoProv.CaoOPCUA","","Server=opc.tcp://192.168.100.100:4890/CaoOPCUA, AccessPath=OPCUA.CAO/RC8")
```

```
Dim oCtrl As Object
Set oCtrl = caoWorkspace.AddController("OpcUa","CaoProv.CaoOPCUA","","Server=opc.tcp://192.168.100.100:4890/CaoOPCUA, AccessPath=OPCUA.CAO/RC8,EventNodeIds=((2,0,1011),(2,0,1010))")
```

### 2.2.2. CaoController::AddVariable method

The following shows the CaoController::AddVariable method of CAO.

```
CaoController::AddVariable
(
 "<Variable name>",       // Variable name
 "<Option>"               // Option character string
)
```

To use the OPC UA provider, set the arguments as follows.

**<Variable name> ::=<Item ID>**

**<Option> ::=[[AccessPath=<Access path>] [,RequestType=<Variable type to receive data>][,NamespaceIndex=< Namespace indexes>,IdentifierType=< Identifier type>,Identifier=< Identifier>]]**

Parameters enclosed by "[]" are not prerequisite items, and default values will be used if not specified. The specified value is interpreted based on the OPC UA server being called. For the data type to receive data, use VT_TYPE. Refer to Table 2-4 for information about VT_TYPE and its value. Refer toTable 2-3 for other options.

For about available system variables, refer to 2.2.3

Format AddVariable(<bstrVariableName:VT_BSTR>[,<vntOption:VT_BSTR>])

**Table 2-3 Option character strings of CaoController::AddVariable**

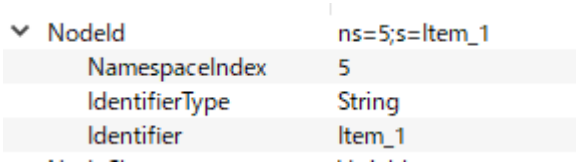
<table>
<tr><th>OPC option name</th><th>Description</th></tr>
<tr><td>AccessPath [=<Access path>]</td><td>Set an access path. (Default: AccessPath option value of the master controller)<br>This overwrites the value of the AccessPath option of AddController.</td></tr>
<tr><td>RequestType [=<Variable type to receive data>]</td><td>Set the variable type to receive data (Default: VT_EMPTY)</td></tr>
<tr><td>NamespaceIndex [=<Namespace indexes>]</td><td>Specify the namespace indexes of the node ID defined on the server. (Default: Unspecified)<br>(See 2.2.2.1)</td></tr>
<tr><td>IdentifierType [=<Identifier type>]</td><td>Specifies the data type of the identifier. (Default: Unspecified)
<table>
<tr><td>Setting</td><td>value</td><td>Description</td></tr>
<tr><td>Numeric</td><td>0</td><td>Specify the identifier numerically.</td></tr>
<tr><td>String</td><td>1</td><td>Specify the identifier as a character string.</td></tr>
</table>
(See 2.2.2.1)</td></tr>
<tr><td>Identifier [=<Identifier>]</td><td>Specify the node identifier. (Default: Unspecified)<br>(See 2.2.2.1)</td></tr>
</table>

#### 2.2.2.1. Node ID

If you specify the namespace indexes, identifier type, or identifier, specify it by node ID. When specifying by node ID, an error will occur if all three types are not specified.

When specifying by node ID, search by item name and access path is not performed.

Example)




In the case of the item with the above settings on the OPC UA server, it will be

NamespaceInex = 5, IdentifierType = 1, Identifier = Item1

.

**Table 2-4 VARIANT type data**

| Data type | Value | Size(byte) | Description |
|---|---|---|---|
| VT_I2 | 2 | 2 | Single-precision (16-bit) integer |
| VT_I4 | 3 | 4 | Double-precision (32-bit) integer |
| VT_R4 | 4 | 4 | Single-precision (32-bit) floating-point |
| VT_R8 | 5 | 8 | Double-precision (64-bit) floating-point |
| VT_CY | 6 | 8 | Same as VT_UI. Currency type |
| VT_DATE | 7 | 8 | Same as VT_R8. Aggregate days from 1899/12/30. |
| VT_BSTR | 8 | variable | Character string<br>Unicode character and null terminator |
| VT_BOOL | 11 | 2 | Same as VT_I2. 0: FALSE -1: TRUE |
| VT_VARIANT | 12 | variable | VARIANT type (Available only for the variant type array) |
| VT_UI1 | 17 | 1 | Unsigned character |
| VT_ARRAY | 8192 | variable | One-dimensional array which data type is mentioned above |

Example

```
Dim oVar As Object
Set oVar = caoCtrl.AddVariable("@CURRENT_ANGLE", "")
Debug.Print oVar.Value
```

### 2.2.3. CaoController::get_VariableNames property

Obtain the system variable list.

### 2.2.4. CaoController::OnMessageEvent

"OPCUA" In the provider, the following OnMessage events are generated.

| Number | Explanation |
|---|---|
| 1 | Simple event |
| 2 | Tracking event |
| 3 | Condition event |

・Simple event

| | | |
|---|---|---|
| Occurrence condition | When you fill the occurrence condition of the simple event set with the server by the change of the value | |
| Number | 1 | |
| Value | Array index:0<br>Type:VT_DATE | Generation date |
| | Array index:1<br>Type:VT_I4 | Severity (1-1000) |
| | Array index:2<br>Type:VT_BSTR | Message |
| | Array index:3<br>Type:VT_ARRAY \| VT_VARIANT | (only if specified in EventNodeIds)<br>Array of added property values |
| Explanation | Generation date:Date when event was generated<br>Severity:Severity of event set with server<br>Message:Message of event set with server<br>Array of added property values: List of property values for the event specified by EventNodeIds during AddController. | |

| ・Tracking event | | |
|---|---|---|
| Occurrence condition | When you fill the occurrence condition of the tracking event set with the server by the change of the value | |
| Number | 2 | |
| Value | Array index:0<br>Type:VT_DATE | Generation date |
| | Array index:1<br>Type:VT_I4 | Severity (1-1000) |
| | Array index:2<br>Type:VT_BSTR | Message |
| | Array index:3<br>Type:VT_BSTR | Generation source |
| | Array index:3<br>Type:VT_ARRAY \| VT_VARIANT | (only if specified in EventNodeIds)<br>Array of added property values |

| | |
|---|---|
| Explanation | Generation date:Date when event was generated<br>Severity:Severity of event set with server<br>Message:Message of event set with server<br>Generation source:Variable where event is generated<br>Array of added property values: List of property values for the event specified by EventNodeIds during AddController. |

| ・Condition event | | |
|---|---|---|
| Occurrence condition | When you fill the occurrence condition of the condition event set with the server by the change of the value | |
| Number | 3 | |
| Value | Array index:0<br>Type:VT_DATE | Generation date |
| | Array index:1<br>Type:VT_I4 | Severity (1-1000) |
| | Array index:2<br>Type:VT_BSTR | Message |
| | Array index:3<br>Type:VT_BSTR | Generation source |
| | Array index:4<br>Type:VT_BSTR | Condition name |
| | Array index:5<br>Type:VT_I4 or<br>VT_BSTR or<br>VT_UI4, VT_UI2, VT_UI2, VT_UI1[8] or<br>VT_UI1[] | Condition identifier |
| | Array index:6<br>Type:VT_I4 | Condition name space index |
| | Array index:7<br>Type:VT_BOOL | Hold flag |
| | Array index:8<br>Type:VT_ARRAY \| VT_VARIANT | (only if specified in EventNodeIds)<br>Array of added property values |

| | |
|---|---|
| Explanation | Generation date:Date when event was generated<br>Severity:Severity of event set with server<br>Message:Message of event set with server<br>Generation source:Variable where event is generated<br>Condition name:Condition name of generated condition event<br>Condition identifier:Value in which generated condition event is identified<br>Condition name space index:Name space index of condition identifier<br>- Identify the condition event by condition identifier + condition name space.<br>Hold flag:It is shown that the object condition was deleted for FALSE.<br>Array of added property values: List of property values for the event specified by EventNodeIds during AddController. |

### 2.2.5. CaoVariable::get_DateTime property

Obtain the time stamp of Item ID.

Note that the value is the time stamp at the time of get_Value property execution.

### 2.2.6. CaoVariable::get_Value property

Obtain the value of Item ID.

### 2.2.7. CaoVariable::put_Value property

Set the value of Item ID.

### 2.2.8. CaoVariable::get_ID property

This get_ID property gets the ID (node ID) of the item ID.

| VT_ARRAY \| VT_VARIANT | |
|---|---|
| Array index:0<br>Type:VT_UI2 | Namespace indexes |
| Array index:1<br>Type:VT_UI2 | Identifier type<br>0: Numeric(VT_UI4)<br>1:String(VT_BSTR) |
| Array index:2<br>Type:VT_UI4 or VT_BSTR | Identifier |

### 2.2.9. CaoMessage::Reply method

The ACK response can be returned to the generated alarm by using the Reply method of the CaoMessage

class.

## 2.3. Error code

In OPC UA provider, the following original error codes are defined. For about ORiN2 common errors, refer to the error code section of ORiN 2 programming guide

**Table 2-5 Original error code list**

| Error name | Error number | Description |
|---|---|---|
| S_OPCUA_TRUST | 0x00100800 | Trusted with the server certificate |
| E_OPCUA_NOT_TRUST | 0x80100800 | Not trusted because no server certificate was confirmed |
| E_OPCUA_PKI_NOTFOUND | 0x80100801 | PKI (Public Key Infrastructure) was not found. |
| E_OPCUA_FAILED_USERCERTIFICATION | 0x80100802 | It failed in the user attestation. |
| E_OPCUA_FAILED_START | 0x80100803 | It failed in beginning the client application. |
| E_OPCUA_FAILED_CONNECTSESSION | 0x80100804 | It failed in the connection of the session. |
| E_OPCUA_FAILED_GETENDPOINT | 0x80100805 | It failed in the acquisition of the end point. |
| E_OPCUA_FAILED_OPENUSERCERTIFICATE | 0x80100806 | The user certificate was not opened. |
| E_OPCUA_FAILED_LOADUSERPRIVATEKEY | 0x80100807 | It failed in loading the user private key. |
| E_OPCUA_ENDPOINT_NOTFOUND | 0x80100808 | The specified end point did not exist. |
| E_OPCUA_FAILED_ADDSESSION | 0x80100809 | It failed in the addition of the session. |
| E_OPCUA_CERTIFICATE_SETTINGNOW | 0x8010080a | It started opening other instances with an individual certificate setting it. |
| E_OPCUA_OTHERINSTANCE_EXIST | 0x8010080b | It tried to set an individual certificate with other instances existed. |

| | | |
|---|---|---|
| E_OPCUA_INVALID_ACCESSPATH | 0x8010080c | The access passing is invalid. |
| E_OPCUA_VARIABLENAME_NOTEXIST | 0x8010080d | It is a variable identifier that doesn't exist. |
| E_OPCUA_TYPEMISMATCH | 0x8010080e | The type is not corresponding. |
| E_OPCUA_BADVARTYPE | 0x8010080f | It is an illegal type. |
| E_OPCUAAE_ACK_NOTCOMPATIBLE_EVENT | 0x80100810 | It is not possible to respond. |
| E_OPCUAAE_ACK_EVENTAUTOACK_ENABLED | 0x80100811 | An automatic response is set. |
| E_OPCUA_SECURITY_CHECKS_FAILED | 0x80100812 | An error occurred verifying security. |
| E_OPCUA_NODEID_NOTEXIST | 0x80100813 | A NodeId that does not exist. |
| E_OPCUA_NODEID_NOTENOUGHOPTIONS | 0x80100814 | There are not enough option strings for NodeId. |
| E_OPCUA_ACCESSDENIED | 0x80100815 | Unable to write because of anonymous authentication. |
| E_OPCUA_NO_TYPE_INFORMATION | 0x80100816 | The specified item has no type information. |
| E_OPCUA_NOT_STRUCTURE_TYPE | 0x80100817 | The specified item is not a structure type. |
| E_OPCUA_UNSUPPORTED_TYPE | 0x80100818 | This type is not supported. |

# 3. Command Reference

## 3.1. Controller classes

| Command | Function | Page |
|---|---|---|
| GetTypeInfo | Gets the type information of the structure. | 20 |

### 3.1.1. CaoController::Execute("GetTypeInfo") Command

Gets the type information (field name list) of the structure.

Format GetTypeInfo()

Argument : VT_ARRAY | VT_VARIANT
[0]: Namespace indexes (VT_UI2)
[1]: Identifier type (VT_UI2) 0: Numeric/1:String
[2]: Identifier (VT_UI4 or VT_BSTR)

Return value : VT_ARRAY | VT_VARIANT
Mixed sequence of the following.
VT_BSTR: Field Name
VT_ARRAY | VT_VARIANT: Child structure information
---VT_BSTR: Field name of the parent structure
---VT_ARRAY | VT_VARIANT: Structure type information

Usage

example

```
Dim result As Variant
result = caoCtrl.Execute("GetTypeInfo", Array(8, 1, "TestValue"))
```

# 4. Certificate files

In OPCUA provider, following two types of certificates are available.

- Application certificate
- Certificate for the certificate authentication

This section describes these certificates and the way of creation.

## 4.1. Application certificate

To start OPCUA provider, an application certificate and the private key file are required. The certificate used here is the same one that is used in the security connection.

The application certificate file and the private key file will be created in the “certs" folder and the “private" folder, respectively. These folders are stored in the “¥PKI¥store" under application's install folder. For the way of creation, see Section 3.3.

Certificate file is “cert_server_self_signed.der" under the [certs] folder

Private Key file is “private_key_server_self_signed.pem" under the [private] folder

You can use [Der], [Pem] or [Password] option to specify them as well.

*If these files are specified, multiple controllers cannot be handled.

Example)

・If any controller exists, a controller that designates an application certificate cannot be added

・A new controller cannot be added while a controller that designates an application certificate is used.

Provider startup will be failed if these files do not exist or do not work properly.

## 4.2. Certificate for the authentication with the certificate

The following authentications are available

- Anonymous authentication
- User name authentication
- Certificate authentication

This certificate is required for the “Certificate authentication"

The application certificate file and the private key file will be created in the “certs" folder and the “private" folder, respectively. These folders are stored in the “¥UserCertificate" under application's install folder. For the way of creation, see Section 3.3.

Certificate file is “cert_client_user.der" under the [certs] folder

Private Key file is “private_key_client_user.pem" under the [private] folder

Provider startup will be failed if these files do not exist or do not work properly

## 4.3. Creating a Certificate

This subsection describes how to create a certificate.

To create a certificate, do as the following steps.

① Start “create_store.bat"

② A certificate is automatically created and will be stored in the setting destination.

# 5. Sample program

## 5.1. UA Demo Server (manufactured by Softing)

The following sample program shows the way to; connect to the UA Demo Server, manufactured by Softing ; with the Put button, set the character string in the textbox to the value of ItemID:StringValue ; with the Get button, get the value of ItemID:StringValue, and then display the value.

**List 5-1 Sample.frm**

```
Option Explicit

Dim Eng As CaoEngine
Dim WS As CaoWorkspace
Dim Ctrl As CaoController
Dim Item As CaoVariable

Private Sub cmdGet_Click()

 'Set values
 txtValue.Text = Item.Value

End Sub

Private Sub cmdPut_Click()

 'Set values
 Item.Value = txtValue.Text
 txtValue.Text = ""

End Sub

Private Sub Form_Load()

 Set Eng = New CaoEngine
 Set WS = Eng.Workspaces(0)
 'Obtain Variable objects
 Set Ctrl = WS.AddController("OpcUa", "CaoProv.OPCUA", "", _
 "Server=opc.tcp://192.168.100.100:51510/UA/DemoServer, AccessPath=Data/Static/Scalar")
 Set Item = Ctrl.AddVariable("StringValue", "")

End Sub

Private Sub Form_Unload(Cancel As Integer)

 Set Item = Nothing
 Set Ctrl = Nothing
 Set WS = Nothing
 Set Eng = Nothing

End Sub
```

## 5.2. CaoOPCUA

### 5.2.1. Use access path

The following sample program shows the way to; connect to the CaoOPCUA; with the Put button, set the character string in the textbox to the value of ItemID:S0 ; with the Get button, get the value of ItemID:S0, and then display the value.

**List 5-2 Sample.frm**

```
Option Explicit

Dim Eng As CaoEngine
Dim WS As CaoWorkspace
Dim Ctrl As CaoController
Dim Item As CaoVariable

Private Sub cmdGet_Click()

 'Obtain values
 txtValue.Text = Item.Value

End Sub

Private Sub cmdPut_Click()

 'Set values
 Item.Value = txtValue.Text
 txtValue.Text = ""

End Sub

Private Sub Form_Load()

 Set Eng = New CaoEngine
 Set WS = Eng.Workspaces(0)
 'Obtain Variable object「
 Set Ctrl = WS.AddController("OpcUa", "CaoProv.OPCUA", "", _
 "Server=opc.tcp://192.168.100.100:4890/CaoOPCUA, AccessPath=OPCUA.CAO/RC8,
User=**********:*******")
 Set Item = Ctrl.AddVariable("S0", "")

End Sub

Private Sub Form_Unload(Cancel As Integer)

 Set Item = Nothing
 Set Ctrl = Nothing
 Set WS = Nothing
 Set Eng = Nothing

End Sub
```

### 5.2.2. Use node ID

The following sample program shows the way to; connect to the CaoOPCUA; with the Put button, set the character string in the textbox to the value of Node ID (namespace index: 5, identifier: Item_1) ; with the Get button, get the value of Node ID (namespace index: 5, identifier: Item_1), and then display the value.

**List 5-3 Sample.frm**

```
Option Explicit

Dim Eng As CaoEngine
Dim WS As CaoWorkspace
Dim Ctrl As CaoController
Dim Item As CaoVariable

Private Sub cmdGet_Click()

 'Obtain values
 txtValue.Text = Item.Value

End Sub

Private Sub cmdPut_Click()

 'Set values
 Item.Value = txtValue.Text
 txtValue.Text = ""

End Sub

Private Sub Form_Load()

 Set Eng = New CaoEngine
 Set WS = Eng.Workspaces(0)
 'Obtain Variable object「
 Set Ctrl = WS.AddController("OpcUa", "CaoProv.OPCUA", "", _
 "Server=opc.tcp://192.168.100.100:4890/CaoOPCUA, User=**********:*******")
 Set Item = Ctrl.AddVariable("TEST", " NamespaceIndex=5,IdentifierType=8,Identifier=Item_1")

End Sub

Private Sub Form_Unload(Cancel As Integer)

 Set Item = Nothing
 Set Ctrl = Nothing
 Set WS = Nothing
 Set Eng = Nothing

End Sub
```