OPC プロバイダ

OPC 用ゲートウェイ

Version 1.2.0

ユーザーズ ガイド

October 18, 2022

【備考】

【改版履歴】

バージョン	日付	内容
1.0.0.0	2006-02-24	初版.
1.0.0.1	2010-02-12	エラーコード追加
1.0.0	2012-07-17	ドキュメントのバージョンルールを変更
1.0.1	2018-10-30	メモリリーク バグ修正
1.0.1	2018-12-21	AE 機能について追記
1.0.2	2021-03-12	メモリリーク バグ修正.
1.0.3	2021-08-31	内部コード最適化.
1.1.0	2021-12-06	名前指定オプション追加.
1.1.1	2022-01-24	リモート接続のバグ修正.
1.2.0	2022-10-18	OPCEnum 対応.
		EventAutoACK オプションの記載場所修正.

【対応機器】

機種	バージョン	注意事項

目次

1. はじめに

本書は CAO から OPC(OLE for Process Control)サーバを介して，PLC (Programmable Logic Controller) にアクセスする手段を与える「OPC 接続用の CAO プロバイダ」(以下，単に OPC プロバイダと呼ぶ) のユーザーズガイドです.

このプロバイダにより，CAO 対応ツールはロボットのみならず OPC サーバを持つ PLC や表示盤に対してもロボットと同様にアクセスすることができます.

本書で解説する OPC クライアント(これが CAO プロバイダになる)では CAO のインタフェースを基本とし，OPC 固有の機能のサポートは目的としません. これらについて将来的に必要な機能であるか否かは，CAO のインタフェースの見直しに同期させて今後進める予定です.

2. プロバイダの概要

2.1. 概要

OPC プロバイダは, CAO を介して PLC と接続するための CAO プロバイダです(図 2-1). このプロバイダにより, CAO 対応ツールはロボットのみならず PLC に対しても同様にアクセスすることができます.

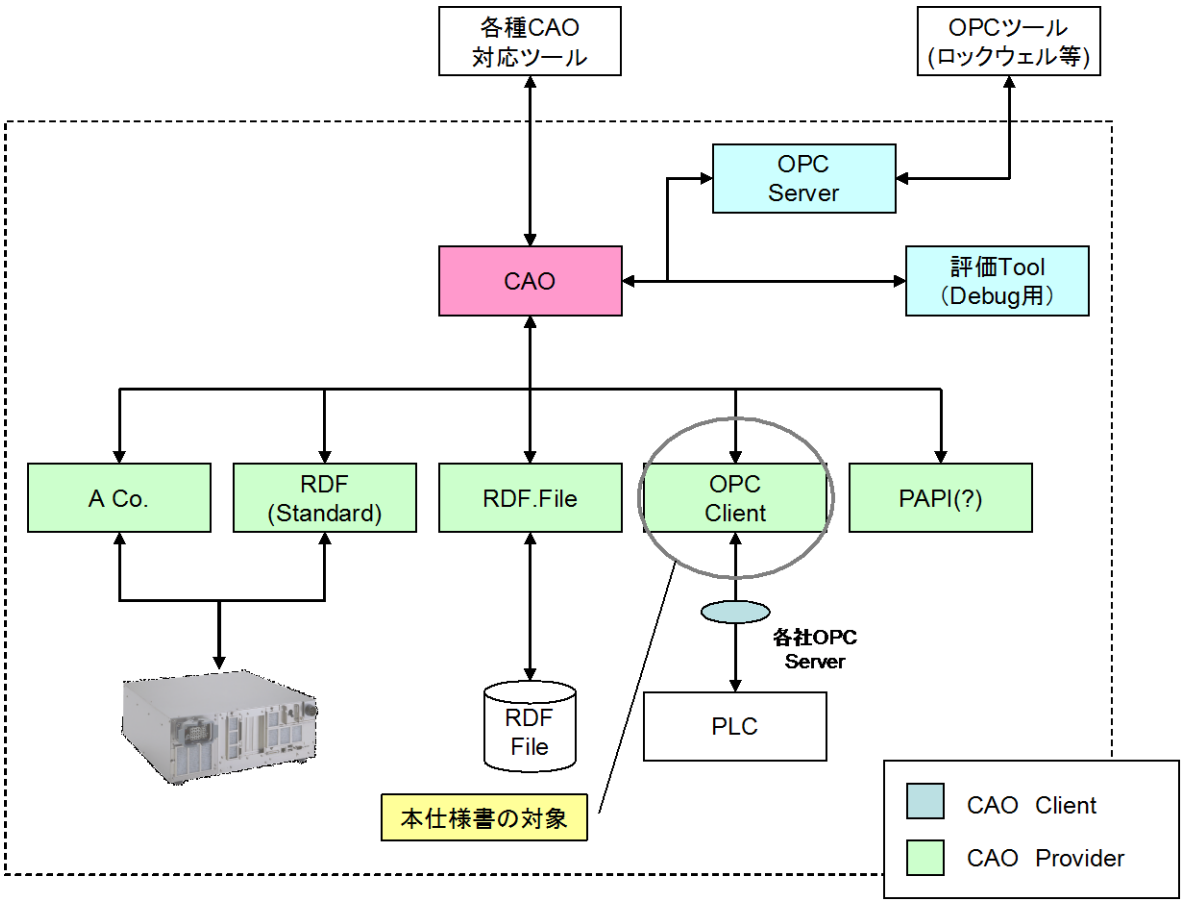


図 2-1 OPC 接続用の CAO プロバイダ

この OPC プロバイダは, OPC サーバが保持する Item の値を CaoVariable オブジェクトから参照することを可能としています(図 2-2). 但し, OPC サーバと CAO とではインタフェースやオブジェクト構造が異なるため, 現段階では同期型の値の読み書きのみの機能を使っています. 非同期型の読み書きは CAO エンジンの機能で実現されているので特には必要ないと思われます.

また, AE の機能として Item 値のアラーム状態を監視したり, イベントの受信を行うことができます.

※本プロバイダでは OPC の DA(DataAccess)と AE(Alarm&Event)に対応しています.

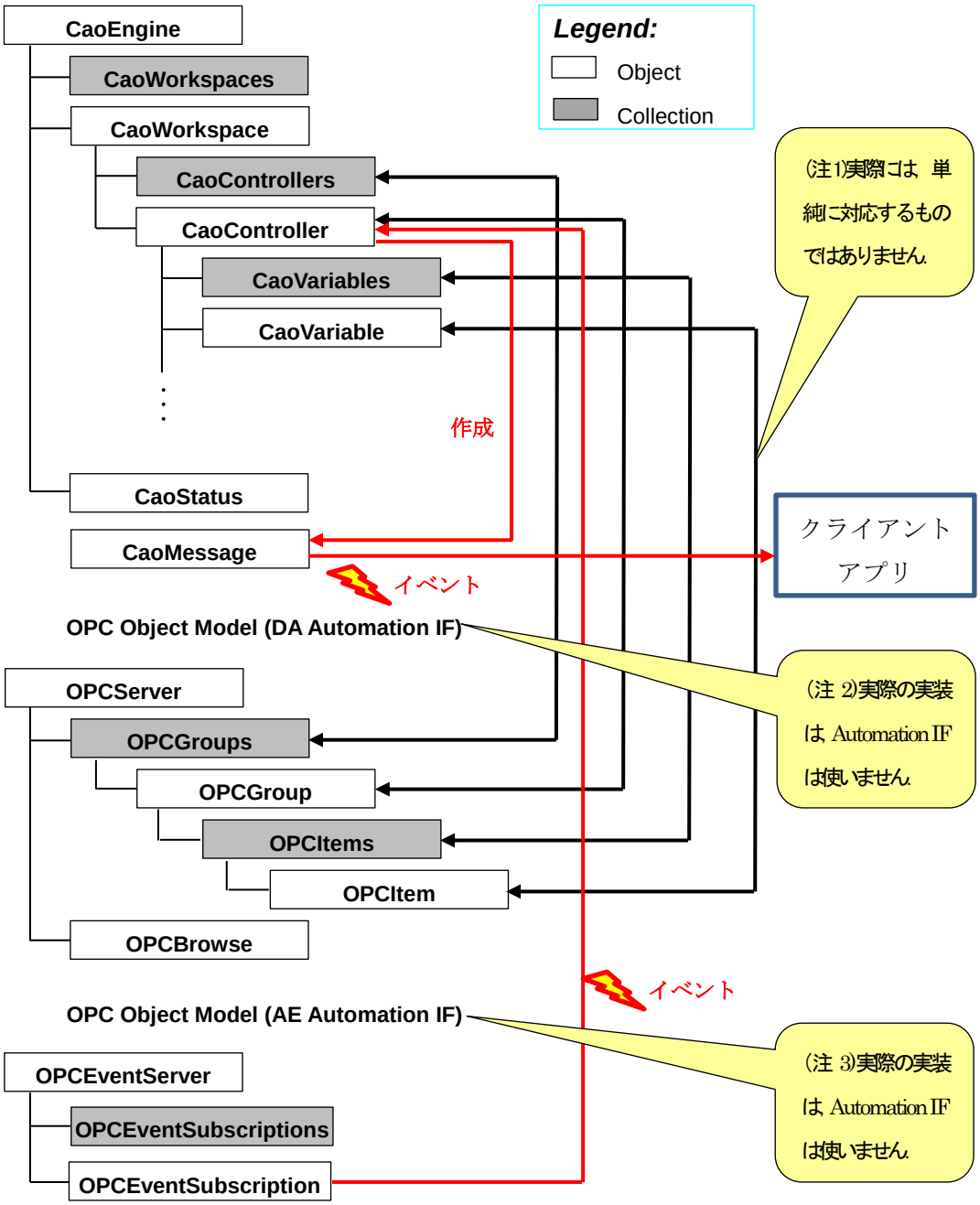
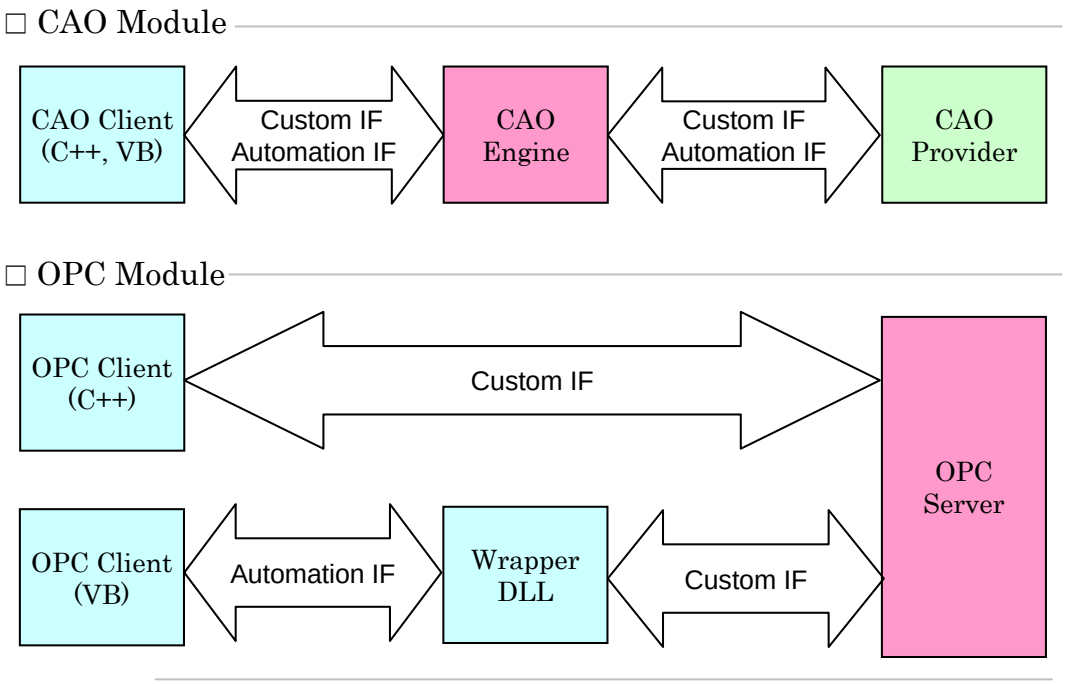


図 2-2 オブジェクトマッピング

また, OPC プロバイダは DA カスタムインタフェースを使い, DA オートメーションインタフェースは使いません. OPC の場合, この 2 つのインタフェースはそのオブジェクトモデルから違うので, 注意する必要があります (図 2-3).



■OPC構成上の問題点

1. カスタムIFとオートメーションIFを使う場合，そのオブジェクトモデルまで違うため，クライアントアプリを開発する言語によって違う操作が必要になります.
2. CAOのように，エンジン部とプロバイダ部に分かれていないため，サーバを提供するベンダーは同じような機能さえも個々に実装しなくてはなりません.

図 2-3：CAO と OPC のモジュール構成比較

表 2-1 OPC プロバイダ

ファイル名	CaoProvOPC.dll
ProgID	CaoProv.OPC
レジストリ登録[1]	regsvr32 CaoProvOPC.dll
レジストリ登録の抹消	regsvr32 /u CaoProvOPC.dll

[1] ORiN SDK でインストールした場合は手動で登録／抹消する必要はありません.

2.2. メソッド・プロパティ

2.2.1. CaoWorkspace::AddController メソッド

CAO の CaoWorkspace クラスの AddController メソッドの仕様を下記に示します.

```
AddController
(
    "<Controller 名>",                  // コントローラ名
    "CaoProv.OPC",                      // プロバイダ名. 固定.
    "<マシン名>",                       // プロバイダの実行マシン名.
    "<オプション>"                      // オプション文字列(OPC オプション)
)
```

ここで, <プロバイダ名>は固定, <実行マシン名>は他のプロバイダと同じ意味です.
<オプション>の書式は次のとおりです.

<オプション> ::= OPCServer=<OPC サーバ名> [, DeadBand =<設定値> [, UpdateRate=<設定値>] [, AccessPath=<設定値>] [, ActiveState=<True|False>] [,TimeBias=<設定値>][,LocaleID=<設定値>][, Machine=<OPC サーバ実行マシン名>]]

OPC グループは“CaoProv_<コントローラ名>”という名前で作成されます.
<OPC サーバ>の指定は必須で, OPC サーバの名前(ProgID)を入力します. これらの中で“[]”で囲まれたパラメータは必須ではなく, 指定しなかった場合はデフォルト値が設定されます. 設定した値の解釈は呼び出している OPC サーバに依存します. その他のオプションについては以下の表 2-2 を参照して下さい.

表 2-2: CaoWorkspace::AddController のオプション文字列

OPC オプション名	説明
OPCServer=< OPC サーバ>	OPC サーバの ProgID.【必須】
DeadBand[=<不感帯>]	不感帯(百分率). OPC の Group オブジェクトを作成するときに使用. (デフォルト:0.0)
UpdateRate[=<更新期間>]	Group の更新期間(ms). OPC の Group オブジェクトを作成するときに使用. (デフォルト:1000)
AccessPath[=<アクセスパス>]	デフォルトのアクセスパス. AddVariable の AccessPath オプションで個別に設定することができます. 個別に設定しなかった場合はここで設定した値が使われます. (デフォルト:空文字列)
ActiveState[=<True|False>]	ActiveState の設定. True:追加するアイテムを Active に設定. False:追加するアイテムを Inactive に設定.

	AddVariable の ActiveState オプションで個別に設定することができます. 個別に設定しなかった場合はここで設定した値が使われます.（デフォルト:True）
TimeBias[=<差分調節時間>]	デバイスの時刻との差分調整時間(min).（デフォルト:0）
LocaleID[=<ロケール ID>]	ロケール ID の設定.（デフォルト:CaoConfig で設定したロケール ID）
Machine [=<OPC サーバ実行マシン名>]	OPC サーバ実行マシン名（IP アドレス指定可能）. 指定したマシン名がプロバイダを実行するマシンであればローカルサーバとして, 違うマシンであればリモートサーバとして起動します.（デフォルト:“localhost”） リモートサーバへの接続については, 2.2.1.1 を参照ください.
EventAutoACK[=<True|False>]	OPC のコンディションイベントを受信した時の動作を指定します.（デフォルト:True） True:自動的に ACK 応答を返す. False:自動的に ACK 応答を返さない.（OnMessage の Reply メソッドで応答） ※OPC AE サーバへの接続時のみ

以下に AddController メソッドを実行するときの例を示します.

```
AddController
(
     “OPC1”,
     “CaoProv.OPC”,                                    // プロバイダ名は固定.
     “”,                                               // インプロセスで起動
     “OPCServer= OPC.DAServer, UpdateRate=1000”        // 更新期間が 1 秒
);
```

2.2.1.1. リモートサーバへの接続について

リモートサーバに接続する為には以下の何れかの条件を満たす必要があります.

1. OPC サーバに OPCEnum サービス[2]が登録されている（優先）
2. プロバイダを実行する PC に OPC サーバのクラス ID が登録されている

満たしていない場合, 接続は失敗します.

[2] OPC サーバのアーキテクチャ（32/64bit）に対応している必要があります.

2.2.2. CaoController::AddVariable メソッド

CAO の CaoController::AddVariable メソッドの仕様を下記に示します.

```
CaoController::AddVariable
(
    "<変数名>",                    // 変数名
    "<オプション>"                 // オプション文字列
)
```

OPC プロバイダを使う場合は, この引数を次のように設定します.

<変数名> ::=<アイテム ID>

<オプション> ::=[[AccessPath=<アクセスパス>] [, ActiveState=<True|False>] [,RequestType=<受け取り変数型>] [,Source=<データソース>]]

これらの中で"[]"で囲まれたパラメータは必須ではなく, 指定しなかった場合はデフォルト値が設定されます. 設定した値の解釈は呼び出している OPC サーバに依存します. 受け取り変数型は VT_TYPE を使用します. VT_TYPE とその数値については表 2-4 を参照して下さい. その他のオプションについては以下の表 2-3 を参照して下さい.

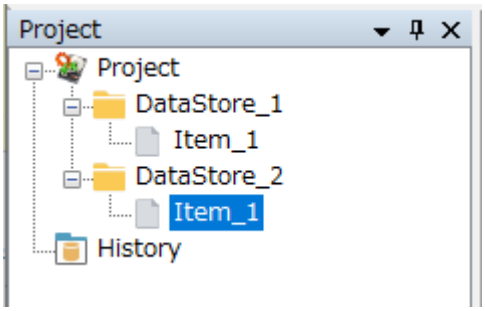
使用可能なシステム変数については, 2.3.1 を参照してください.

表 2-3: CaoController::AddVariable のオプション文字列

OPC オプション名	説明
Name [=<タグ名>]	対象のタグ名を指定します. このオプションを省略した場合は変数名がタグ名として使用されます. 使用方法については, 2.2.2.1 を参照して下さい.
AccessPath [=<アクセスパス>]	アクセスパスの設定. (デフォルト: 親コントローラの AccessPath オプション値) AddController の AccessPath オプションで設定した値を上書きします.
ActiveState[=<True|False>]	ActiveState の設定. (デフォルト: 親コントローラの ActiveState オプション値) AddController の ActiveState オプションで設定した値を上書きします. True: 追加するアイテムを Active に設定. False: 追加するアイテムを Inactive に設定.
RequestType [=<受け取り変数型>]	受け取り変数型の設定. (デフォルト: VT_EMPTY)

Source [=<データソース>]	データソースの設定.（デフォルト:ActiveState が Active の場合は OPC_DS_CACHE，Inactive の場合は OPC_DS_DEVICE） 1:OPC_DS_CACHE 2:OPC_DS_DEVICE

2.2.2.1. Name オプションの使用方法



上記のような CaoOPC サーバがあった場合に一つのコントローラで

- DataStore_1 の Item_1
- DataStore_2 の Item_1

を同時に使用するには Name オプションを使用して，以下の様に変数を追加します.

```
Dim eng As CaoEngine
Dim ws As CaoWorkspace
Dim ctrl As CaoController
Dim var1 As CaoVariable
Dim var2 As CaoVariable

Set eng = new CaoEngine
Set ws = eng.Workspaces(0)
Set ctrl = ws.AddController(“”, “CaoProv.OPC”, “”, “OPCServer=OPC.CAO”)
Set var1 = ctrl.AddVariable(“ITEM-1”, “AccessPath=DataStore_1, Name=Item_1”)
Set var2 = ctrl.AddVariable(“ITEM-2”, “AccessPath=DataStore_2, Name=Item_1”)
・・・
```

表 2-4: VARIANT 型データ(一部)

データ型	指定値	バイト数	説明
VT_I2	2	2	単精度(16 ﾋﾞｯﾄ)整数
VT_I4	3	4	倍精度(32 ﾋﾞｯﾄ)整数

VT_R4	4	4	単精度(32 ﾋﾞｯﾄ)浮動小数
VT_R8	5	8	倍精度(64 ﾋﾞｯﾄ)浮動小数
VT_CY	6	8	VT_UI8 と同じ　通貨.
VT_DATE	7	8	VT_R8 と同じ　 1899/12/30 からの通算日時.
VT_BSTR	8	可変	文字列 VT_UI4 で示された文字数の後に UNICODE 文字と NULL ターミネータを付加
VT_BOOL	11	1	VT_UI1 と同じ　0:FALSE,　0以外::TRUE
VT_VARIANT	12	可変	バリアント型 (バリアント型配列にのみ使用)
VT_UI1	17	1	符号なし文字
VT_ARRAY	8192	可変	上記データタイプの一次元配列

2.2.3. CaoController::get_VariableNames プロパティ

この get_VariableNames プロパティは, システム変数の一覧を取得します.

2.2.4. CaoController::OnMessage イベント

“OPC”プロバイダでは, 以下の OnMessage イベントが発生します.

※OPC AE サーバへの接続時のみ

Number	説明
1	シンプルイベント
2	トラッキングイベント
3	コンディションイベント

・シンプルイベント		
発生条件	値の変化によりサーバで設定されたシンプルイベントの発生条件を満たしたとき	
Number	1	
Value	配列インデックス:0 型:VT_DATE	発生日時
	配列インデックス:1 型:VT_I4	重大度(1～1000)
	配列インデックス:2 型:VT_BSTR	メッセージ
説明	発生日時:イベントが発生した日時 重大度:サーバで設定されたイベントの重大度	

	メッセージ:サーバで設定されたイベントのメッセージ

・トラッキングイベント		
発生条件	値の変化によりサーバで設定されたトラッキングイベントの発生条件を満たしたとき	
Number	2	
Value	配列インデックス:0 型:VT_DATE	発生日時
	配列インデックス:1 型:VT_I4	重大度(1～1000)
	配列インデックス:2 型:VT_BSTR	メッセージ
	配列インデックス:3 型:VT_BSTR	発生ソース
説明	発生日時:イベントが発生した日時 重大度:サーバで設定されたイベントの重大度 メッセージ:サーバで設定されたイベントのメッセージ 発生ソース:イベントが発生した変数	

・コンディションイベント		
発生条件	値の変化によりサーバで設定されたコンディションイベントの発生条件を満たしたとき	
Number	3	
Value	配列インデックス:0 型:VT_DATE	発生日時
	配列インデックス:1 型:VT_I4	重大度(1～1000)
	配列インデックス:2 型:VT_BSTR	メッセージ
	配列インデックス:3 型:VT_BSTR	発生ソース
	配列インデックス:4 型:VT_BSTR	コンディション名
	配列インデックス:5 型:VT_BSTR	サブコンディション名

	配列インデックス:6 型:VT_I4	コンディション識別子
	配列インデックス:7 型:VT_BOOL	保持フラグ
説明	発生日時:イベントが発生した日時 重大度:サーバで設定されたイベントの重大度 メッセージ:サーバで設定されたイベントのメッセージ 発生ソース:イベントが発生した変数 コンディション名:発生したコンディションイベントのコンディション名 サブコンディション名:発生したコンディションイベントのサブコンディション名 コンディション識別子:発生したコンディションイベントを識別する値 保持フラグ:FALSE の場合，対象コンディションが消去されたことを示します	

2.2.5. CaoVariable::get_DateTime プロパティ

この get_DateTime プロパティは，アイテム ID のタイムスタンプを取得します．

get_Value プロパティを実行した際に取得したタイムスタンプであることに注意して下さい．

2.2.6. CaoVariable::get_Value プロパティ

この get_Value プロパティは，アイテム ID の値を取得します．

2.2.7. CaoVariable::put_Value プロパティ

この put_Value プロパティは，アイテム ID の値を設定します．

2.2.8. CaoMessage::Reply メソッド

CaoMessage クラスの Reply メソッドを使用して，発生済みのアラームに ACK 応答を返すことができます．

※OPC AE サーバへの接続時のみ

2.3. 変数一覧

2.3.1. コントローラクラス

表 2-5 コントローラクラス システム変数一覧

変数名	データ型	説明	属性	
			get	put
@SERVER_STATE	VT_I4	OPC サーバの現在の状態を取得します. 1:OPC_STATUS_RUNNING 2:OPC_STATUS_FAILED 3:OPC_STATUS_NOCONFIG 4:OPC_STATUS_SUSPENDED 5:OPC_STATUS_TEST	○	-

2.4. エラーコード

OPC プロバイダでは，以下の固有エラーコードが定義されています．ORiN2 共通エラーについては，「ORiN2 プログラミングガイド」のエラーコードの章を参照してください．

表 2-6 独自エラーコード一覧

エラー名	エラー番号	説明
E_OPC_QUALITY_NOT_GOOD	0x80100800	アイテムの状態が GOOD になっていません.
E_OPCAE_ACK_NOTCOMPATIBLE_EVENT	0x80100801	応答することができません.
E_OPCAE_ACK_EVENTAUTOACK_ENABLED	0x80100802	自動応答が設定されています.

3. サンプルプログラム

以下に CaoOPC に接続し, Put ボタンでテキストボックスの文字列をアイテム ID:item1 の値に設定し, Get ボタンでアイテム ID:item1 の値を取得し, 表示するサンプルを示します.

List 3-1 Sample.frm

```
Option Explicit
Dim eng As CaoEngine
Dim ws As CaoWorkspace
Dim ctrl As CaoController
Dim item As CaoVariable


Private Sub Form_Load()
    Set eng = New CaoEngine
    Set ws = eng.Workspaces(0)

    ‘変数オブジェクトを取得します
    Set ctrl = ws.AddController("", _
                                "CaoProv.OPC", _
                                "", _
                                "OPCServer=OPC.CAO, UpdateRate=500")
    Set item = ctrl.AddVariable("item1", _
                                "AccessPath=OPCC2")

End Sub

Private Sub cmdGet_Click()
    ‘値を取得します
    txtValue.Text = item.Value

End Sub

Private Sub cmdPut_Click()
    ‘値を設定します
    item.Value = txtValue.Text
    txtValue.Text = ""

End Sub

Private Sub Form_Unload(Cancel As Integer)
    Set item = Nothing
    Set ctrl = Nothing
    Set ws = Nothing
    Set eng = Nothing

End Sub
```

付録A. OPC サーバの設定

付録A.1. デジタル製 OPC サーバの設定

デジタルの OPC サーバは，インプロセス用，ローカル/リモートプロセス用があるようですが，同時にインストールできないようなので，ここではインプロセス用のみ説明します．CAO 自体がローカルプロセス／リモートプロセスが可能なので，そもそもインプロセス以外は特に必要はありません．この場合，CAO のプロセスと PRO-SERVER の2つのプロセスができます．

AddController メソッドと AddVariable メソッドの引数は次のように指定します．

AddController の OPCServer オプション：：＝ ”DIGITAL.OPCPRO.1”

AddVariable の変数名：：＝ ”<局名>:<アドレス>”

例えば，”GP1:LS0022:2” は，GP1 のアドレス LS0022 のワードデータを指定したことになります．アドレスには，表示器に接続された PLC のアドレスを直接指定することもできます．例えば，AB 社製の SLC5 が接続されている場合，”GP1:N027007:2”とすると，SLC5 のアドレス N027007 を指定したことになります．

OPC サーバの詳細な仕様については，ヘルプ(opcpro.hlp) を参照してください．

□ OPCPRO.ini ファイルの内容

```
[Server]
Server_Max=10              <-（注）インスタンス数の制限がかかっている！
Group_Max=10
Item_Max=100
InproSvr_ScanRate=250
LocalSvr_ScanRate=500
[Client]
CharacterStringCode=0
ItemID_Separator=. :!
```

付録A.2. Rockwell 製 RS-Linx の設定

Rockwell 製の OPC サーバを使うには，まず RS-Linx でいくつかの設定を先にする必要があります（下記参照）．それが完了した後，AddController メソッドと AddVariable メソッドの引数は次のように指定します．

AddController の OPCServer オプション：：＝ ”RSLinx OPC Server”[3]

AddVariable の変数名：：＝ ”<トピック名>!<アドレス>”

例えば，デフォルトトピック名を topic1 とする場合は AddController メソッドのオプション文字列は "OPCServer=RSLinx OPC Server, AccessPath=topic1" と指定します．また，AddVariable メソッドの変数名を "topic1!N027:041"，そのオプション文字列を "RequestType=2" と指定すると，topic1 のアドレス N027:041 のワードデータを指定したことになります．（この例の場合はデフォルトのトピック名が指定してあるので変数名の topic1 は省略することもできます．）

□ RS-Linx の設定手順を示します[4]．

1. [Topic Configuration…]で新しいトピックを作成する[5]．トピック名は任意の文字列．
2. 1で作成したトピックを選択し，[Data Collection]タブでプロセッサタイプを指定する．SLC5/04 プロセッサの場合は「SLC503+」を選択する．
3. [Advanced Communication] タブで Communication Driver を指定する．
4. [Advanced Communication] タブで Station に接続先 PLC の Node アドレスを指定する．

[3] RSLinx の ProgID には空白が入っているので丸括弧で囲んでください．OPCServer=(RSLinx OPC Server)
[4] RS-Linx のパッケージによっては OPC の機能が使えないらしいので注意すること．また，バージョンによっては若干設定手順が違う可能性がある．
[5] DDE プロジェクトを Open した状態でないとこのメニューは有効にならない．

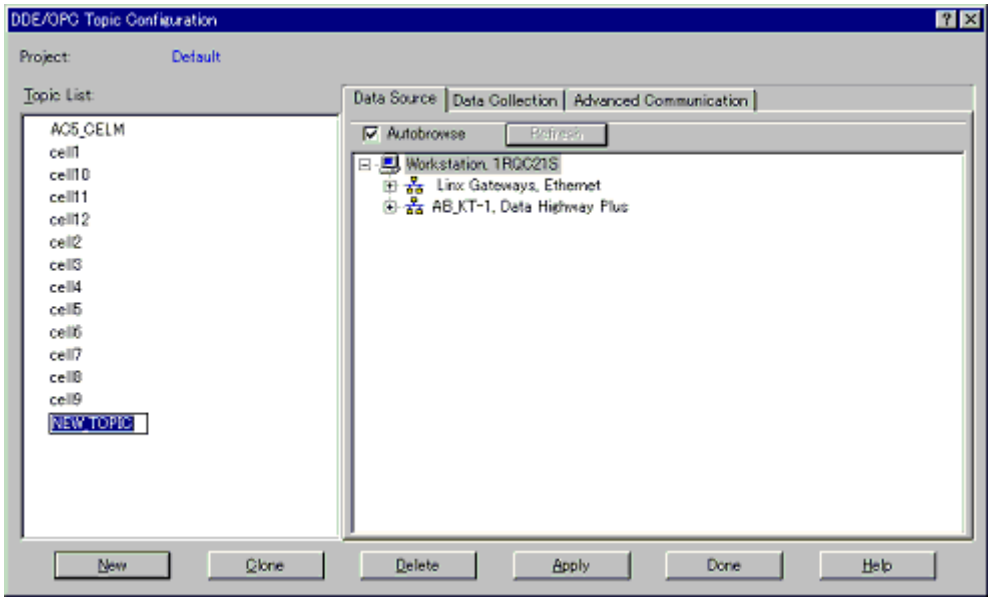


図 3-1：[Topic Configuration]ダイアログ

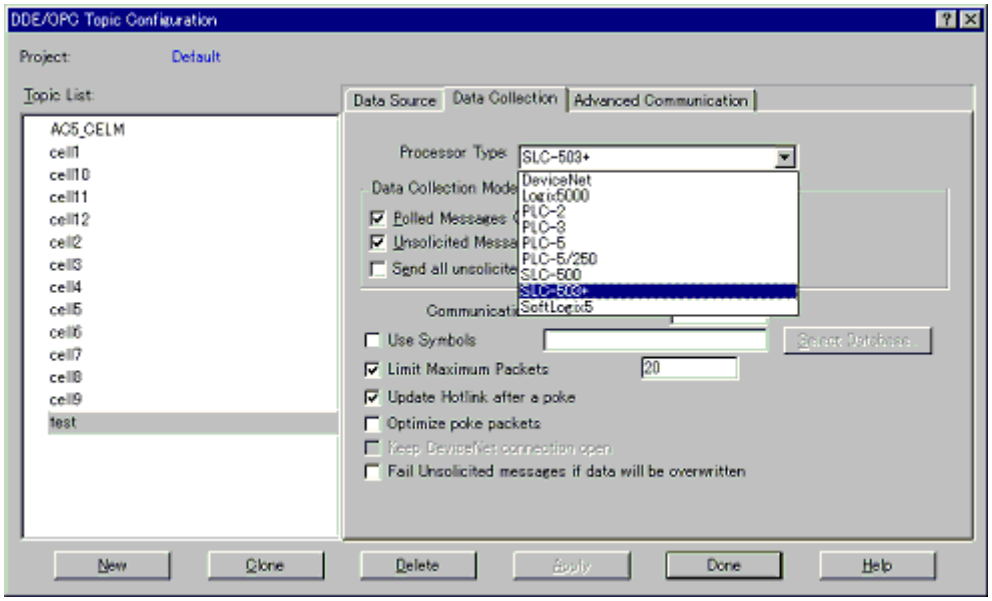


図 3-2：[Data Collection]タブ

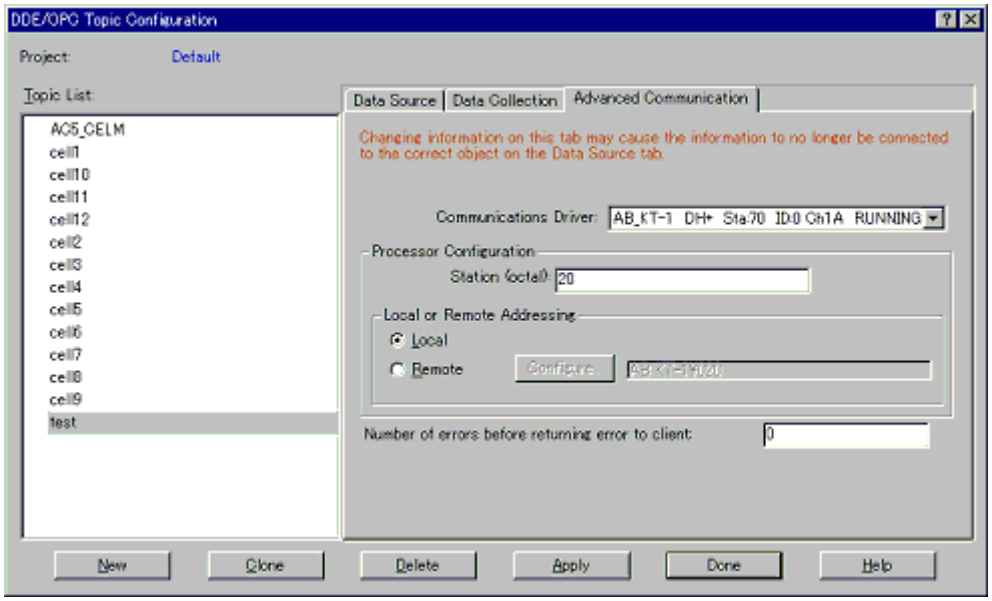


図 3-3: [Advanced Communication]タブ

付録A.3. ICS Triplex 製 ISaGRAF の設定

ICS Triplex 製の OPC サーバを使うには，まず ISaGRAF でいくつかの設定を先にする必要があります（下記参照）．それが完了した後，AddController メソッドと AddVariable メソッドの引数は次のように指定します．

AddController の OPCServer オプション：：＝ ”ICSTriplex.IsagrafOPCDA20.1”

AddVariable の変数名：：＝ ”<コンフィグ名>.<リソース名>.<変数名>”

例えば，”config1.resource1.var1” は，config1 の中にある resource1 の変数 var1 を指定したことになります．

□ ISaGRAF の設定手順を示します．

1. ISaGRAF の Workbench.exe を起動する．
2. ファイルを開くからプロジェクトファイルを開く．
3. メニューの[Debug]→[Simulation]を実行する．

付録A.4. TAKEBISHI 製 DeviceXPlorer の設定

TAKEBISHI 製の DeviceXPlorer を OPC サーバとして接続するには，まず DeviceXPlorer でデバイスの設定をする必要があります．それが完了した後，AddController メソッドと AddVariable メソッドの引数は次のように指定する．

AddController の OPCServer オプション：：＝ ”Takebishi.DXP”

AddVariable の変数名：：＝ ”<パス>.<タグ名>”

例えば，”Device1.LoopFlag” は，Device1 の中にある LoopFlag を指定したことになる．