RobCom プロバイダ
Matrox 視覚システム用

Version 1.0.1

ユーザーズ ガイド

December 16, 2015

【備考】

【改版履歴】

バージョン	日付	内容
1.0.0	2014-09-10	初版.
1.0.1	2014-11-12	コマンド追加 SendCommand, ReceiveCommand, SendPositionRequest, WaitPositionReply
	2015-12-16	プロバイダの dll 名等追記

目次

1. はじめに

本書は Matrox 社製の Matrox Design Assistant(以下 DA)で設定されたビジョンシステムと通信する為の CAO プロバイダである，RobCom プロバイダのユーザーズガイドです．

RobCom プロバイダは Ethernet 接続された DA のプロジェクトの Communication:Robot(図 1-1)とコマンドの送受信を行います．通信は Ethernet(TPC/IP)接続をサポートしています． Matrox IrisGT スマートカメラの DA プロジェクトとの通信概要を図 1-2 に示します．

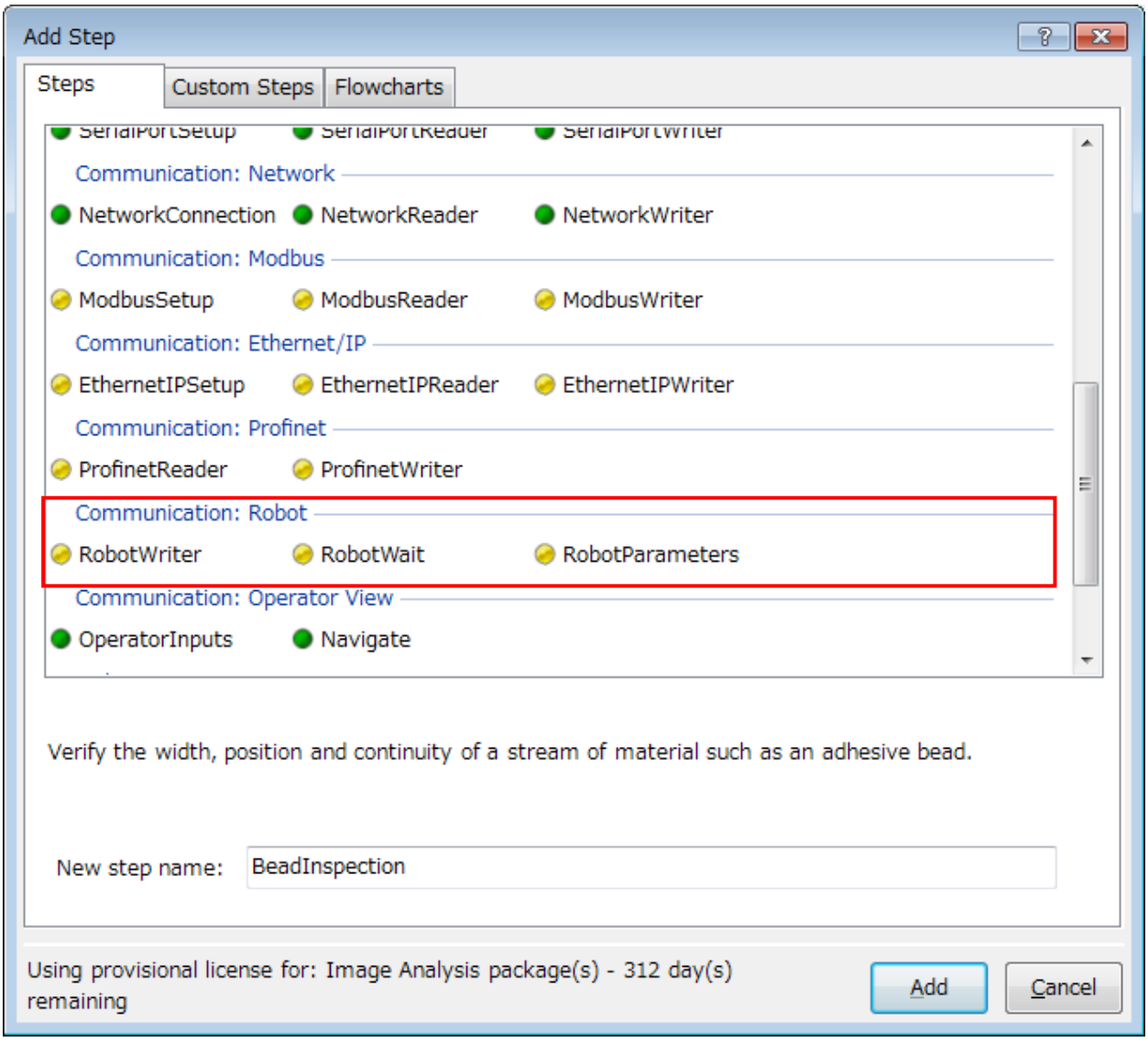

図 1-1 Communication:Robot

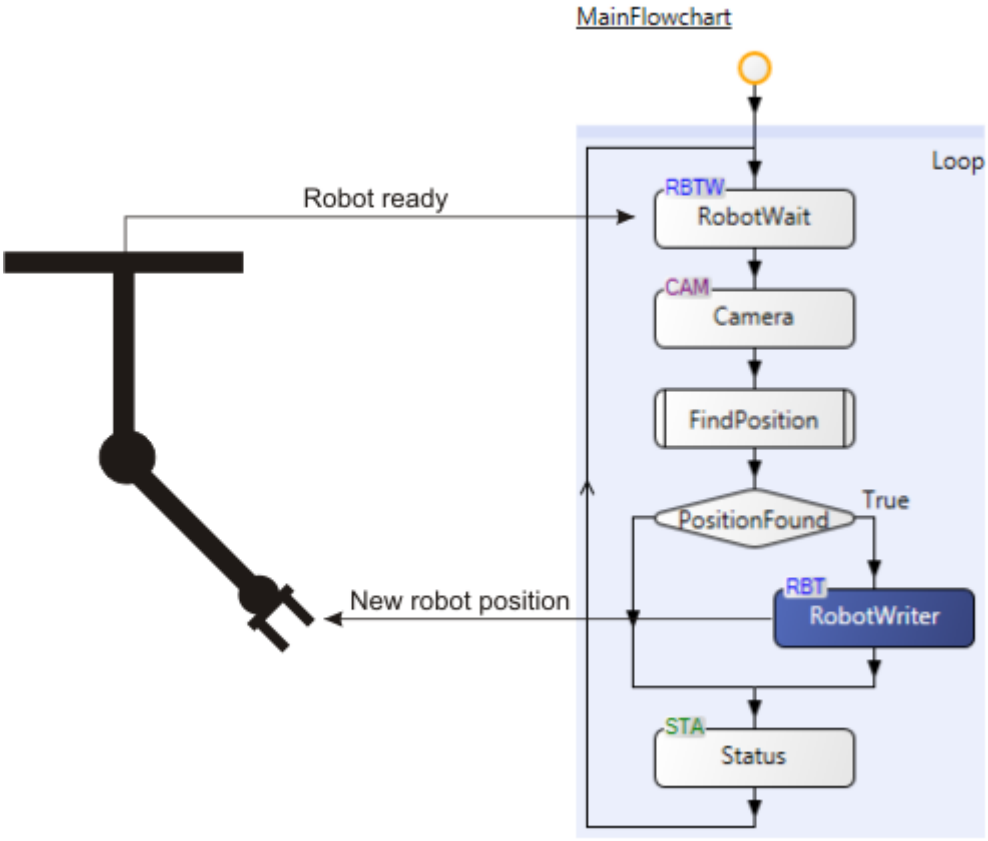
MainFlowchart
Loop
Robot ready
RBTW
RobotWait
CAM
Camera
FindPosition
PositionFound
True
New robot position
RBT
RobotWriter
STA
Status

図 1-2 通信概要

2. プロバイダの概要

2.1. 概要

RobCom プロバイダは, Addcontroller 時にサーバーが起動します. DA のプロジェクト側の設定をするときは, AddController で指定されたポートに対して接続するよう, プロジェクトの設定をしてください(図 2-1). コマンドの実行方法として CaoController::Execute を提供しています. 各コマンドの詳細は 2.5 章を参照してください.

RobCom プロバイダのファイル形式は DLL(Dynamic Link Library)であり, CAO エンジンから使用時に動的にロードされます. RobCom プロバイダを使用するにあたっては ORiN2SDK をインストールするか, 下表を参照して手作業でレジストリ登録を行う必要があります.

表 2-1 RobCom プロバイダ

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

2.2. メソッド・プロパティ

2.2.1. CaoWorkspace::AddController メソッド

RobCom プロバイダでは AddController 時に, 通信用の接続パラメータを参照し, 通信の接続を行います. このときオプションでタイムアウト, 待ち受けポート番号を指定します. DA のプロジェクトでは「Configure Robots」で指定したコントローラのポート番号を設定してください. (図 2-1 参照)

書式 AddController(<bstrCtrlName:VT_BSTR>,<bstrProvName:VT_BSTR>,
<bstrPcName:VT_BSTR > [,<bstrOption:VT_BSTR>])

bstrCtrlName	: [in] コントローラ名 任意
bstrProvName	: [in] プロバイダ名 固定値 =”CaoProv.Matrox.Robcom”
bstrPcName	: [in] プロバイダの実行マシン名
bstrOption	: [in] オプション文字列

以下にオプション文字列に指定するリストを示します.

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

[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.

オプション	意味
Timeout[=<タイムアウト時間>]	送受信時のタイムアウト時間(ミリ秒)を指定します．（デフォルト：500）
Port[=<ポート番号>]	指定した TPC ポートを待ちうけ用に開きます．

使用例

```
Dim caoCtrl as Object
caoCtrl = cao.AddController("RobCom", "caoProv.Matrox.RobCom", "", "timeout=1000, port=2001")
```

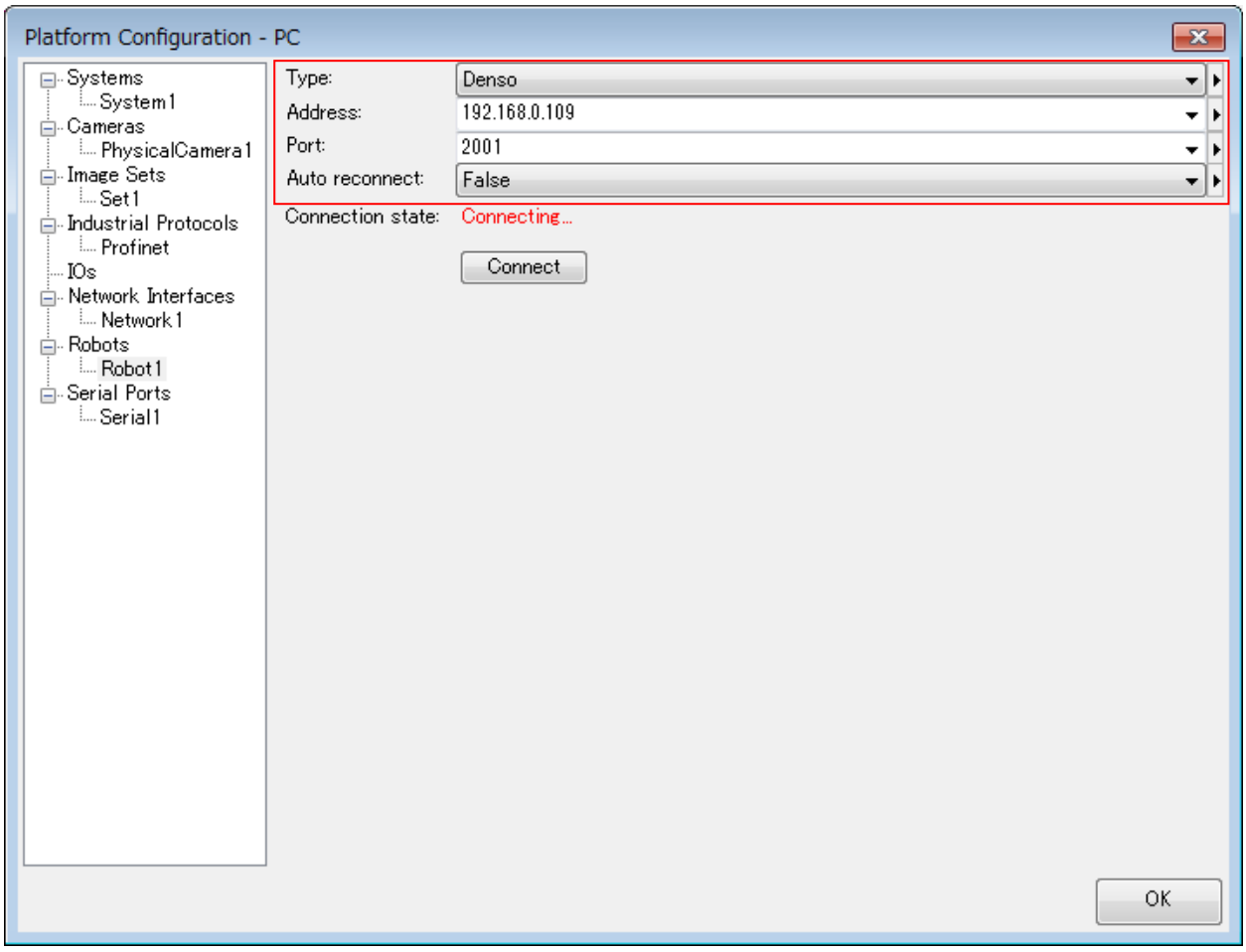

図 2-1 DA での設定画面

2.2.2. CaoController::Execute メソッド

ビジョンシステムとコマンドの送受信を行います．第 1 引数にコマンド名，第 2 引数にコマンドのパラメータを指定します．各コマンドの詳細は 2.5 章コマンドリファレンスを参照してください．

書式 Execute (<bstrCommandName:VT_BSTR>,[<vntParam : VT_VARIANT>])

bstrCommandName: [in] コマンド名

vntParam : [in] パラメータ

2.2.3. CaoController::AddVariable メソッド

システムの情報を取得する為の変数を作成します. 使用できる変数については表 2-3 を参照してください.

書式 AddVariable(<bstrVariableName:VT_BSTR>,[< bstrOption: VT_BSTR >])

bstrVariableName : [in] 変数名

bstrOption : [in] オプション文字列 (未使用)

使用例

```
Dim bstrVal as String
bstrVal = caoCtrl.AddVariable("@VERSION", "")

bstrVal : "1.0.0"
```

2.2.4. CaoVariable::put_value プロパティ

現在変数クラスでは put_value プロパティをサポートしていません.

2.2.5. CaoVariable::get_value プロパティ

表 2-3 コントローラクラス システム変数一覧のフォーマットで取得できます.

2.3. 変数一覧

2.3.1. コントローラクラス

表 2-3 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
@MAKER_NAME	VT_BSTR	プロバイダを作成した会社名を取得します.	○	-
@VERSION	VT_BSTR	プロバイダのバージョン情報を取得します.	○	-
@STATUS	VT_BOOL	ビジョンシステムの接続状態を取得します.	○	-

		True : 接続あり False : 接続無し		

2.4. エラーコード

RobCom 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 2-4 独自エラーコード一覧

エラー名	エラー番号	説明
E_NOCLIENT	0x80100001	ビジョンシステムが接続されていない.
E_CONECTED	0x80100002	ビジョンシステムが既に接続されている.
E_ROBCOMSTATUS	0x80100003	ビジョンシステムがエラーを返している.
E_WINDOWS_MASK	0x8091xxxx	winsock の標準エラーコード(16bit)を xxxx(下位 2 バイト)に格納します. 例) 10053 (WSAECONNABORTED) ネットワークが破棄された. → 0x80912745

2.5. コマンドリファレンス

本章では CaoController::Execute メソッドの各コマンドについて解説します.

表 2-5 CaoController::Execute コマンド一覧

RobCom コマンド	コマンド	機能	
通信コマンド			
	SendPos	ID, ポジションデータをビジョンシステムへ送り, ステップを進めます.	P11
	ReceivePos	ビジョンシステムからポジションデータを受信します.	P11
	SendPositionRequest	ID, ポジションデータ, ステータスをビジョンシステムへ送り, ステップを進めます.	P12
	WaitPositionReply	ビジョンシステムから ID, ポジションデータ, ステータスを受信します.	P12

	SendCommand	コマンドをビジョンシステムに送ります.	P13
	ReceiveCommand	コマンド結果をビジョンシステムから受信します.	P14
補助コマンド			
-	DisconnectClient	接続されたビジョンシステムを切断します.	P14
-	IsConnected	ビジョンシステムの接続状態を確認します.	P15
-	Clear	受信したデータを破棄します.	P15
-	SetTimeout	受信時のタイムアウト時間を設定します.	P15

2.5.1. 通信コマンド

2.5.1.1. CaoController::Execute(“SendPos”) コマンド

ID, ポジションデータをビジョンシステムへ送り, ステップを進めます. ビジョンシステムからのデータを受信するには ReceivePos コマンドを使用してください.

書式 SendPos(<ID >, <Potision>)

<ID > : [in] オブジェクト ID (VT_I4)

<Potision> : [in] ポジションデータ (VT_R4 | VT_ARRAY)
X, Y, Z, Rx, Ry, Rz

戻り値 : [out] なし

接続が確立されていない場合, E_NOCLIENT が返ります.

使用例

```
Dim lID as Long
Dim vntPos as Variant

lID = 1
vntPos = Array(100, 100, 200, 10, 10, 10)
caoCtrl.Execute "SendPos", Array(lID, vntPos)
```

2.5.1.2. CaoController::Execute (“ReceivePos”) コマンド

ビジョンシステムから送られてきたポジションデータを受信します.

書式 ReceivePos ()

引数 : なし

戻り値 : [out] ポジションデータ (VT_R4 | VT_ARRAY)
X, Y, Z, Rx, Ry, Rz

Timeout で設定された時間以内に受信できない場合, 0x80000900 を返します. RobotWriter の Status を Error にした場合, E_ROBCOMSTATUS を返します.

使用例

```
Dim vntPos as Variant

vntPos = caoCtrl.Execute("ReceivePos")
```

2.5.1.3. CaoController::Execute(“SendPositionRequest”) コマンド

ID, ポジションデータ, ステータスをビジョンシステムへ送り, ステップを進めます. ビジョンシステムからのデータを受信するには WaitPositionReply コマンドを使用してください.

書式 SendPositionRequest (<Status>, <ID >, <Potision>)

<Status>	:	[in] ステータスコード (VT_I4)
		0 : M_COM_SUCCESS
		1 : M_COM_ERROR
<ID >	:	[in] オブジェクト ID (VT_I4)
<Potision>	:	[in] ポジションデータ (VT_R4 | VT_ARRAY)
		X, Y, Z, Rx, Ry, Rz
戻り値	:	[out] なし

接続が確立されていない場合, E_NOCLIENT が返ります.

使用例

```
Dim lStatus as Long
Dim lID as Long
Dim vntPos as Variant

lStatus = 0
lID = 1
vntPos = Array(100, 100, 200, 10, 10, 10)
caoCtrl.Execute "SendPositionRequest", Array(lStatus , lID, vntPos)
```

2.5.1.4. CaoController::Execute (“WaitPositionReply”) コマンド

ビジョンシステムから送られてきたデータを受信します.

書式 WaitPositionReply ()

引数	:	なし
戻り値	:	[out] <Status>, <ID>, <Potision> (VT_ARRAY | VT_VARIANT)
		<Status> : ステータス (VT_I4)
		<ID> : オブジェクト ID (VT_I4)
		<Potision> : ポジションデータ (VT_R4 | VT_ARRAY)
		X, Y, Z, Rx, Ry, Rz

Timeout で設定された時間を超えると, 0x80000900 を返します.

使用例

```
Dim vntRet as Variant
Dim lStatus as Long
Dim lID as Long
Dim vntPos as Variant

vntRet = caoCtrl.Execute("WaitPositionReply")
lStatus = vntRet(0)
lID = vntRet(1)
vntPos = vntRet(2)
```

2.5.1.5. CaoController::Execute("SendCommand") コマンド

ビジョンシステムへコマンドの送信を行います.

書式 SendCommand (< OperaCode>, <Status>, <ID>, <Potision>)

< OperaCode>	:	[in] オペレーションコード(VT_I4)
		1 : M_COM_ROBOT_FIND_POSITION
< Status>	:	[in] ステータス(VT_I4)
		0 : M_COM_SUCCESS
		1 : M_COM_ERROR
<ID>	:	[in] ID (VT_I4)
<Potision>	:	[in] ポジションデータ (VT_R4 | VT_ARRAY)
		X, Y, Z, Rx, Ry, Rz
戻り値	:	[out] なし

コマンドのデータを作成し, ビジョンシステムに送信します. 応答を受信するには ReceiveCommand コマンドを使用してください.

接続が確立されていない場合, E_NOCLIENT が返ります.

使用例

```
Dim lOpCode as Long
Dim lStatus as Long
Dim lID as Long
Dim vntPos as Variant

lOpCode = 1
lStatus = 0
lID = 1
vntPos = Array(100, 100, 200, 10, 10, 10)
caoCtrl.Execute "SendCommand", Array(lOpCode, lStatus, lID, vntPos)
```

2.5.1.6. CaoController::Execute (“ReceiveCommand”) コマンド

ビジョンシステムからコマンドの応答を受け取ります.

書式 ReceiveCommand ()

引数 : なし

戻り値 : [out] < OperaCode>, <Status>, <ID>, <Potision>

(VT_ARRAY | VT_VARIANT)

< OperaCode> : オペレーションコード (VT_I4)

2 : M_COM_ROBOT_FIND_POSITION_RESULT

<Status> : ステータス (VT_I4)

0 : M_COM_SUCCESS

1 : M_COM_ERROR

<ID> : オブジェクト ID (VT_I4)

<Potision> : ポジションデータ (VT_R4 | VT_ARRAY)

X, Y, Z, Rx, Ry, Rz

Timeout で設定された時間を超えると, 0x80000900 を返します.

使用例

```
Dim vntRet as Variant
Dim lOperaCode as Long
Dim lStatus as Long
Dim lID as Long
Dim vntPos as Variant

vntRet = caoCtrl.Execute("ReceiveCommand")
lOperaCode = vntRet(0)
lStatus = vntRet(1)
lID = vntRet(2)
vntPos = vntRet(3)
```

2.5.2. 補助コマンド

2.5.2.1. CaoController::Execute (“DisconnectClient”) コマンド

ビジョンシステムとの接続を切断します.

書式 DisconnectClient()

引数 : なし

戻り値 : なし

使用例

```
caoCtrl.Execute "DisconnectClient"
```

2.5.2.2. CaoController::Execute (“IsConnected”) コマンド

接続されているビジョンシステムがあるか確認します.

書式 IsConnected ()

引数 : なし

戻り値 : [out] 接続状態 (VT_BOOL)

TRUE : 接続あり

FALSE : 接続無し

使用例

```
Dim lVal as Long
lVal = caoCtrl.Execute("IsConnected")
```

2.5.2.3. CaoController::Execute (“Clear”) コマンド

既に受信しているデータをクリアします.

書式 Clear

引数 : なし

戻り値 : なし

使用例

```
Call caoCtrl.Execute("Clear")
```

2.5.2.4. CaoController::Execute (“SetTimeout”) コマンド

タイムアウト時間を変更します

書式 SetTimeout (<Timeout>)

Timeout : [in] タイムアウト時間 msec (VT_I4)

戻り値 : なし

使用例

```
Call caoCtrl.Execute("SetTimeout", 1000)
```

3. サンプルプログラム

以下に RC8 Pacscript でのサンプルプログラムを示します,

```
'!TITLE "MatroxRobComSample"

#include <Variant.h>

Sub Main
    Dim caoCtrl as Object

    ' socket bind (Port=PortNumber)
    caoCtrl = cao.AddController("RobCom", "CaoProv.Matrox.RobCom", "", "Port=2001,
timeout=6000")

    Do
        ' Wait for Connection
        if caoCtrl.IsConnected Then
            Exit Do
        End if
        delay 100
    Loop

    ' Send Position and Request
    caoCtrl.SendPos 1, VarChangeType(CurPos, VT_R8 + VT_ARRAY)

    ' Get Position
    P11 = caoCtrl.ReceivePos

    ' Disconnect Client
    caoCtrl.DisconnectClient

    ' Close
    cao.Controllers.Remove caoCtrl.Index
    caoCtrl = Nothing
End Sub
```

4. 付録

4.1. 通信コマンドについて

RobCom プロバイダでは DA で作成されたビジョンシステムのプロジェクトとの通信コマンドを 3 種類準備しています. 以下に使用例を示します.

4.1.1. SendPos と ReceivePos

見つけたワークのポジションデータのみを取得したい場合に使用します. ワークが見つからない場合, RobotWriter の Status を Error に設定すると ReceivePos は E_ROBCOMSTATUS を返しますので, OnError ステートメント等でエラー処理を行ってください.

```
'!TITLE "MatroxRobComSample"

#include <Variant.h>

Sub Main
	On Error Goto ErrorProc
	Dim caoCtrl as Object

	' socket bind (Port=PortNumber)
	caoCtrl = cao.AddController("RobCom", "CaoProv.Matrox.RobCom", "", "Port=2001,
timeout=6000")

Connect:
	Do
		' Wait for Connection
		if caoCtrl.IsConnected Then
			Exit Do
		End if
		delay 100
	Loop

SendAndRecieve:
	Do
		' Send Position and Request
		caoCtrl.SendPos 1, VarChangeType(CurPos, VT_R8 + VT_ARRAY)
Recieve:
		' Get Position
		P11 = caoCtrl.ReceivePos
	Loop
	Exit Sub

ErrorProc:
	On Error Goto ErrorProc
	PrintDbg "Error:" & HEX(Err.OriginalNumber)

	' Work not found
	If (Err.OriginalNumber = &H80100003) Then
		Goto SendAndRecieve
	' Timeout
	ElseIf (Err.OriginalNumber = &H80000900) Then
		Goto Recieve
	' Error
	Else
		caoCtrl.DisconnectClient
		Goto Connect
	End If
```

```
End Sub
```

4.1.2. SendPositionRequest と WaitPositionReply

見つけたワークのポジションデータの他に，ID やステータスを取得したい場合に使用します．DA の RobotWriter で設定した Status は WaitPositionReply の戻り値に格納されますので，E_ROBCOMSTATUS は返りません．

```
'!TITLE "MatroxRobComSample"

#Include < Variant.h >

Sub Main
     On Error GoTo ErrorProc
     Dim caoCtrl As Object
     Dim vntRet as Variant

     ' socket bind (Port=PortNumber)
     caoCtrl = Cao.AddController( "RobCom", "CaoProv.Matrox.RobCom", "", "Port=2001,
timeout=6000" )

Connect:
     Do
          ' Wait for Connection
          If caoCtrl.IsConnected Then
               Exit Do
          End If
          Delay 100
     Loop

SendAndRecieve:
     Do
          ' Send Position and Request
          caoCtrl.SendPositionRequest 0, 1, VarChangeType( CurPos, VT_R8 + VT_ARRAY )
Recieve:
          ' Get Position
          vntRet = caoCtrl.WaitPositionReply
          I11 = vntRet(0)    ' Status
          I12 = vntRet(1)    ' ID
          P11 = vntRet(2)    ' Potision
     Loop
     Exit Sub

ErrorProc:
     On Error GoTo ErrorProc
     PrintDbg "Error:" & Hex( Err.OriginalNumber )

     ' Work not found
     If ( Err.OriginalNumber = &H80000900 ) Then
          GoTo Recieve
     ' Error
     Else
          caoCtrl.DisconnectClient
          GoTo Connect
     End If

End Sub
```

4.1.3. SendCommand と ReceiveCommand

通信コマンドを生成し，ビジョンシステムと通信する際に使用します．DA で新しいコマンドが規定された場合に使用してください．

```
'!TITLE "MatroxRobComSample"

#Include < Variant.h >

Sub Main
     On Error GoTo ErrorProc
     Dim caoCtrl As Object
     Dim vntRet as Variant

     ' socket bind (Port=PortNumber)
     caoCtrl = Cao.AddController( "RobCom", "CaoProv.Matrox.RobCom", "", "Port=2001,
timeout=6000" )

Connect:
     Do
               ' Wait for Connection
               If caoCtrl.IsConnected Then
                        Exit Do
               End If
               Delay 100
     Loop

SendAndRecieve:
     Do
               ' Send Command
               caoCtrl.SendCommand 1, 0, 1, VarChangeType( CurPos, VT_R8 + VT_ARRAY )
Recieve:
               ' Get Command reply
               vntRet = caoCtrl.ReceiveCommand
               I10 = vntRet(0)    ' OperaCode
               I11 = vntRet(1)    ' Status
               I12 = vntRet(2)    ' ID
               P11 = vntRet(3)    ' Potision
     Loop
     Exit Sub

ErrorProc:
     On Error GoTo ErrorProc
     PrintDbg "Error:" & Hex( Err.OriginalNumber )

     ' Work not found
     If ( Err.OriginalNumber = &H80000900 ) Then
               GoTo Recieve
     ' Error
     Else
               caoCtrl.DisconnectClient
               GoTo Connect
     End If

End Sub
```