VBP プロバイダ

VB6 プロバイダ用ゲートウェイ

Version 1.2.0

ユーザーズ ガイド

July 17, 2012

【備考】

【改版履歴】

バージョン	日付	内容
1.0.0.0	2007-04-06	初版.
1.0.1.0	2009-09-07	VB プロバイダ作成手順, プログラミングTips追記.
1.0.1.1	2010-02-12	エラーコード追加
1.1.1.0	2010-09-16	モーダレスダイアログの表示方法追記
1.2.0.0	2011-12-26	メッセージオブジェクトの初期化処理の改良
1.2.0	2012-07-17	メドキュメントのバージョンルールを変更

【対応機器】

機種	バージョン	注意事項

【ご注意】

本プロバイダを使用する場合は別途“**ORiN2 VBGateway Provider**”ライセンスが必要です.

目次

1. はじめに

このドキュメントは VB プロバイダ用 Gateway プロバイダ（以下 VB_Gateway プロバイダ）のユーザーズガイドです.

これまでのプロバイダ開発は C++, COM (Component Object Model)や多目的データ型に対する高度な知識が必要で, 簡単に開発ができませんでした. この問題の解決策の1つとして, より実装が簡便である Visual Basic 6.0 によるプロバイダ開発を提案します. これにより, 従来までのプロバイダよりも容易に開発することが可能になり, プロバイダの種類も一層の充実させることが可能になります.

VB_Gateway プロバイダは, VB Provider をラップして CAO Provider 仕様へ変換するゲートウェイ・プロバイダとしての役割を果たします.

2. プロバイダの概要

2.1. 概要

VB プロバイダのインターフェース仕様は，CAO プロバイダのインターフェース仕様に近い仕様になっている．しかし，Visual Basic の制約により，同一のインターフェースを再現することが困難である．そこで VB_Gateway プロバイダは，VB プロバイダをラップすることで，この差分を吸収し CAO プロバイダと同一の機能を提供する．

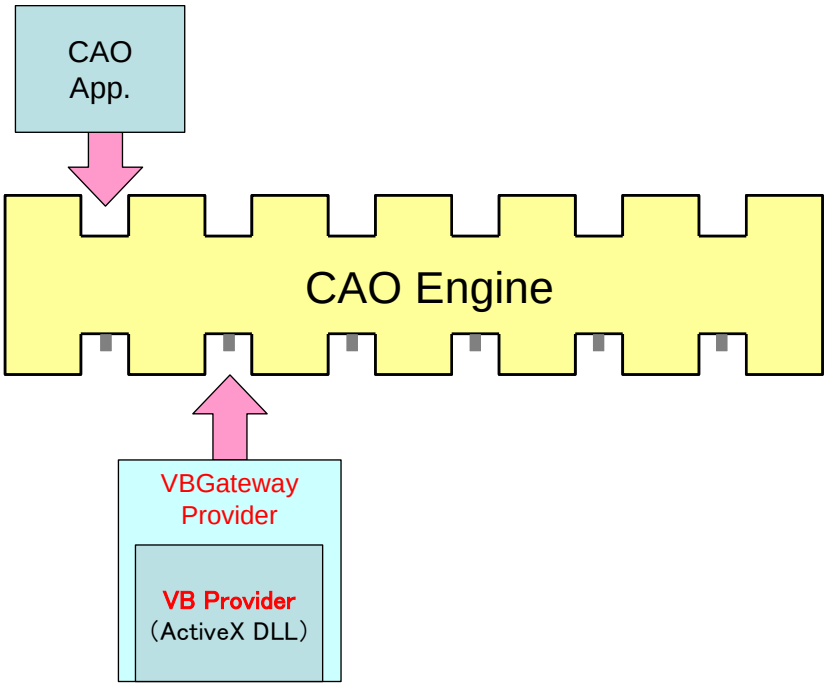


図 1-1 アーキテクチャ

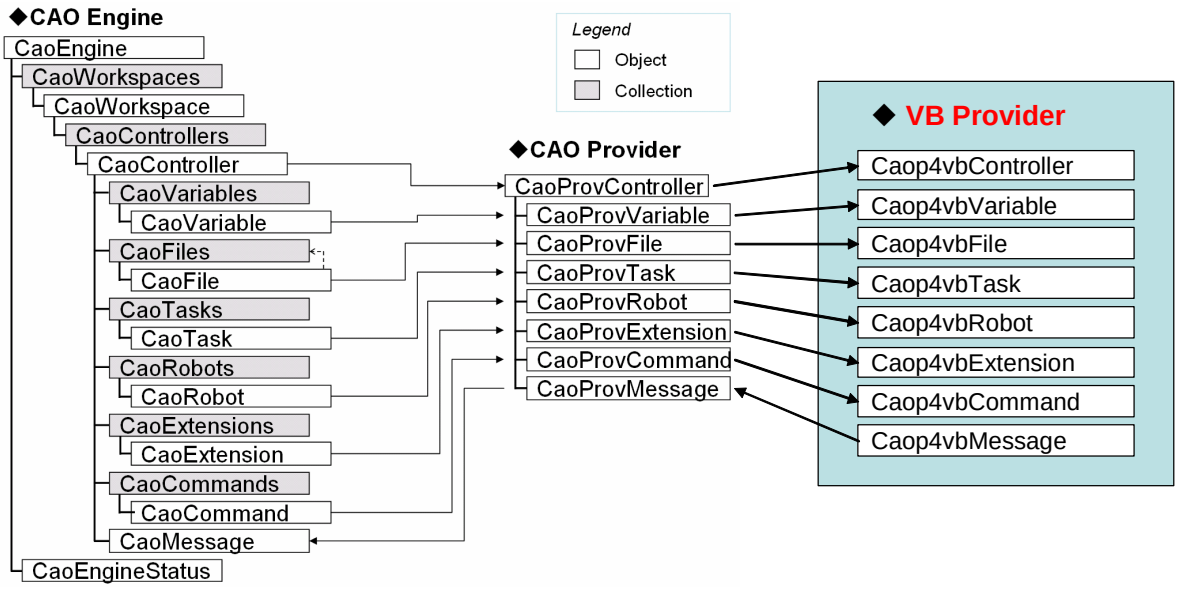


表 1-1 VB_Gateway プロバイダ

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

VB_Gateway プロバイダでは AddController 時に，VB プロバイダと接続します．
このときオプションで使用する VB プロバイダを ProgID で指定します．

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

bstrCtrlName	: [in] コントローラ名
bstrProvName	: [in] プロバイダ名．固定値 =" CaoProv.VBGateway".
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 2-1　CaoWorkspace::AddController のオプション文字列

オプション	説明
VBP=< ProgID>	VB プロバイダの ProgID VB プロバイダのプロジェクト名と一致する．

2.2.2. AddController 以外のメソッド・プロパティ

VB Gateway プロバイダは，コントローラ，ロボット，ファイル，タスク，変数，拡張ボードクラスのすべてのメソッド，プロパティが実装されており，それらはそのまま VB プロバイダの対応する関数をコールします．

2.3. エラーコード

VBGateway プロバイダでは，固有のエラーコードはありません．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．

3. サンプルプログラム

List 3-1 Sample.frm

```
Private Eng As CaoEngine
Private Ctrl As CaoController
Private Var As CaoVariable

Private Sub Form_Load()

    ' CAO エンジンの生成
    Set Eng = New CaoEngine

    ' コントローラの接続
    Set Ctrl = Eng.Workspaces(0).AddController("Sample", "CaoProv.VBGateway", "",
"VBP=CaoProv_Sample_Macro")

    ' VB プロバイダのメソッド呼び出し
    Set Var = Ctrl.AddVariable("Var", "")
    MsgBox Var.Value

End Sub
```

4. VB プロバイダ作成手順

4.1. スケルトンの作成

CAO プロバイダウィザードを使用して，Visual Basic6.0 のスケルトンプロジェクトを作成します．

(1) スタートメニューの「全てのプログラム」→「ORiN2」→「CAO」→「Provider」→「CaoProvWizard」を起動します．

(2) Output Folder にインストール先のパスを指定します．

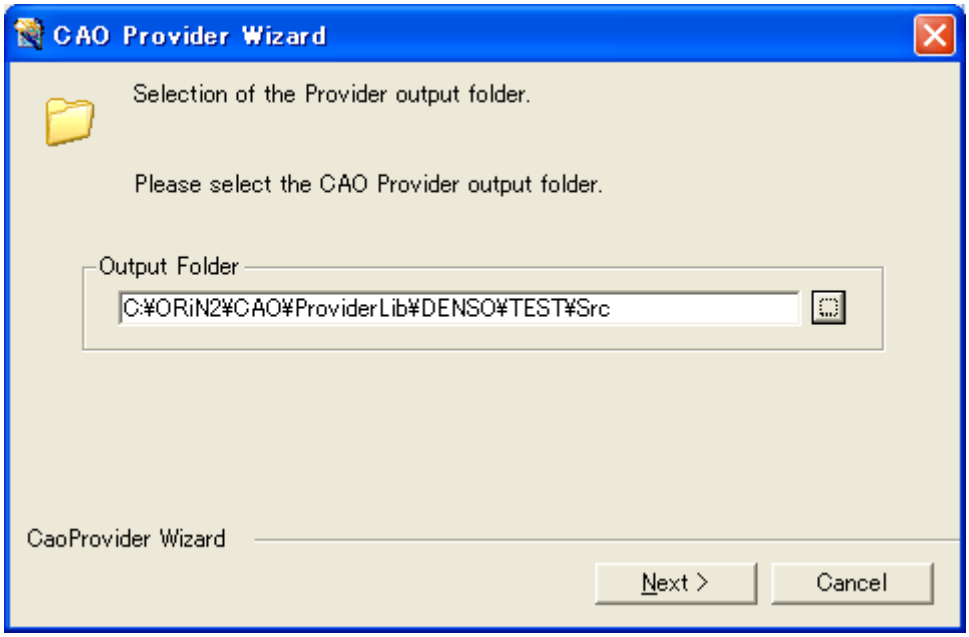


図 4-1 パスの指定

(3) Project Type に「Visual Basic 6.0 SP6」を選択し，DLL Name，Vendor Name，Module Name を入力します．

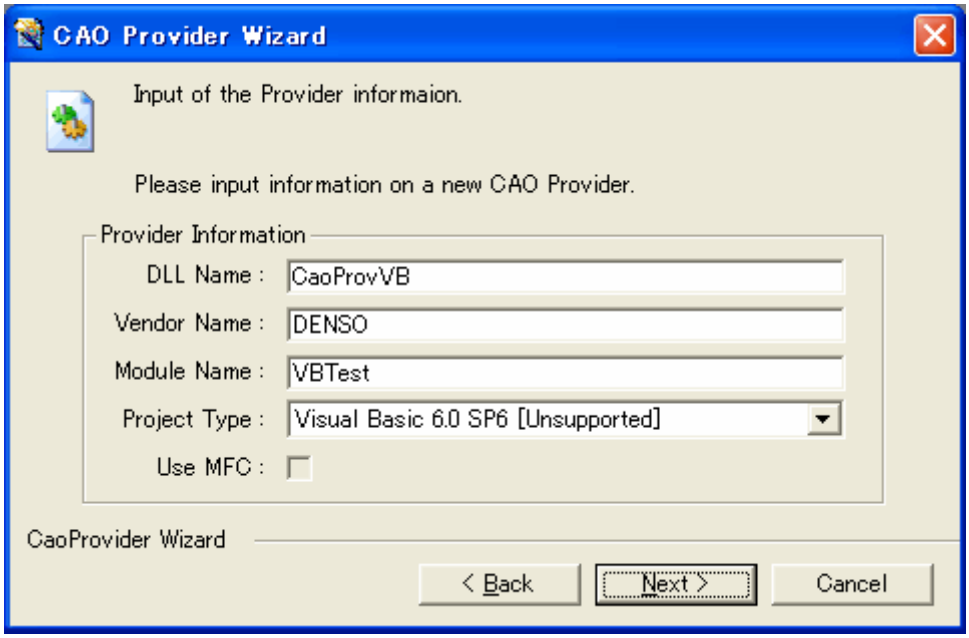


図 4-2 プロバイダの設定

(4) 確認画面がでますので, はい(Y)を押します.

図 4-3 プロバイダの設定確認

(5) 下記の画面が出れば, 作成完了です.

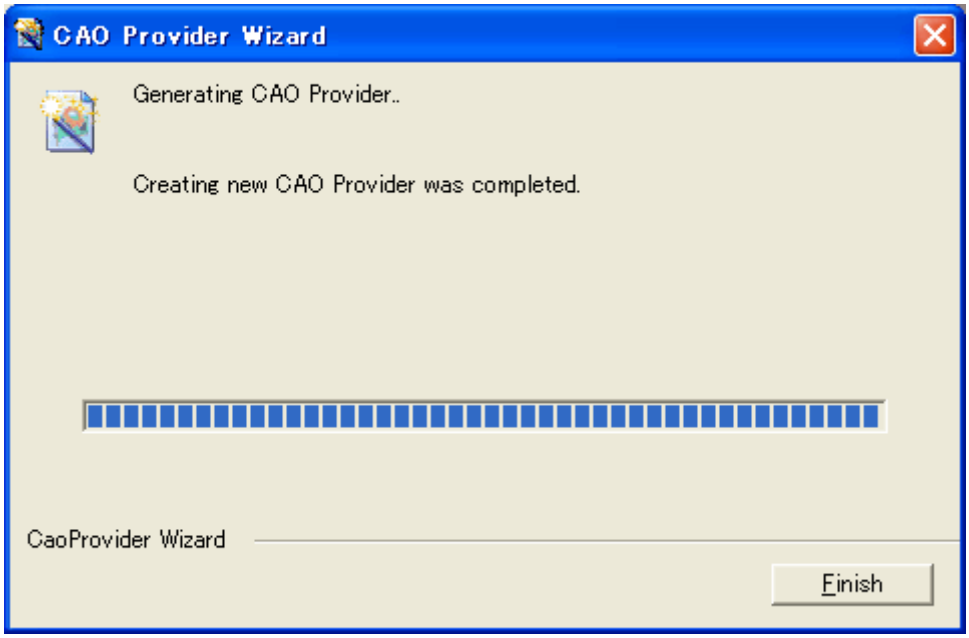


図 4-4 スケルトン作成完了

4.2. 関数の実装

先ほど指定したフォルダに CaoProv.vbp が作成されていますので，必要な関数を実装します．

4.3. デバッグ

(1) プロジェクト・プロパティのデバッグタブで CAO エンジン(Cao.exe)を指定します．

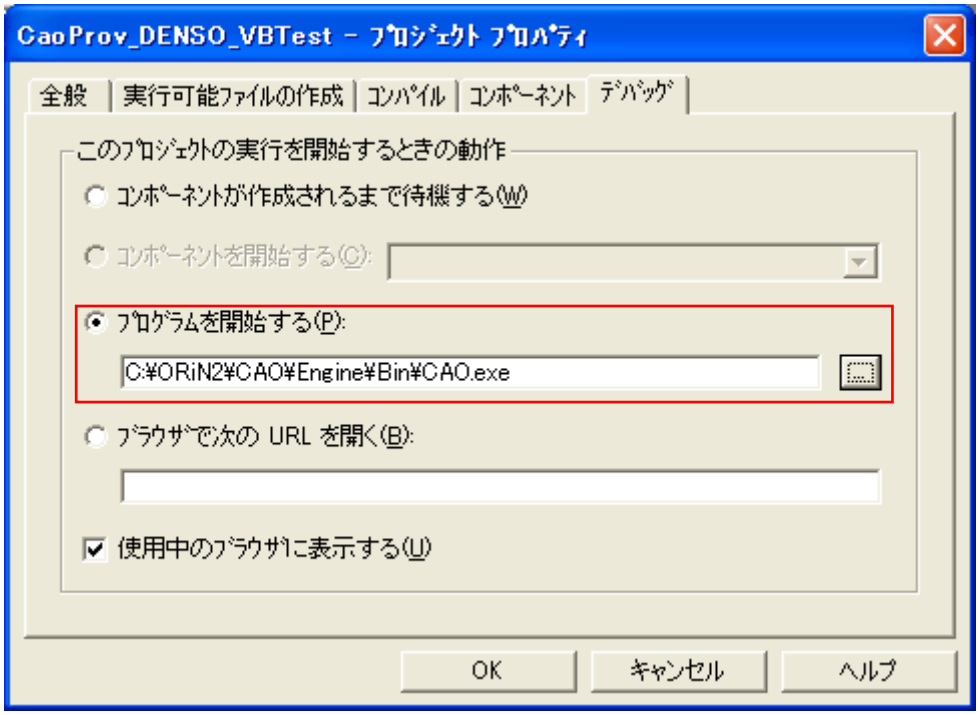


図 4-4 CAO エンジンの指定

(2)　デバッグを開始して，適当なクライアントプログラム(例：CaoTester)を起動します．

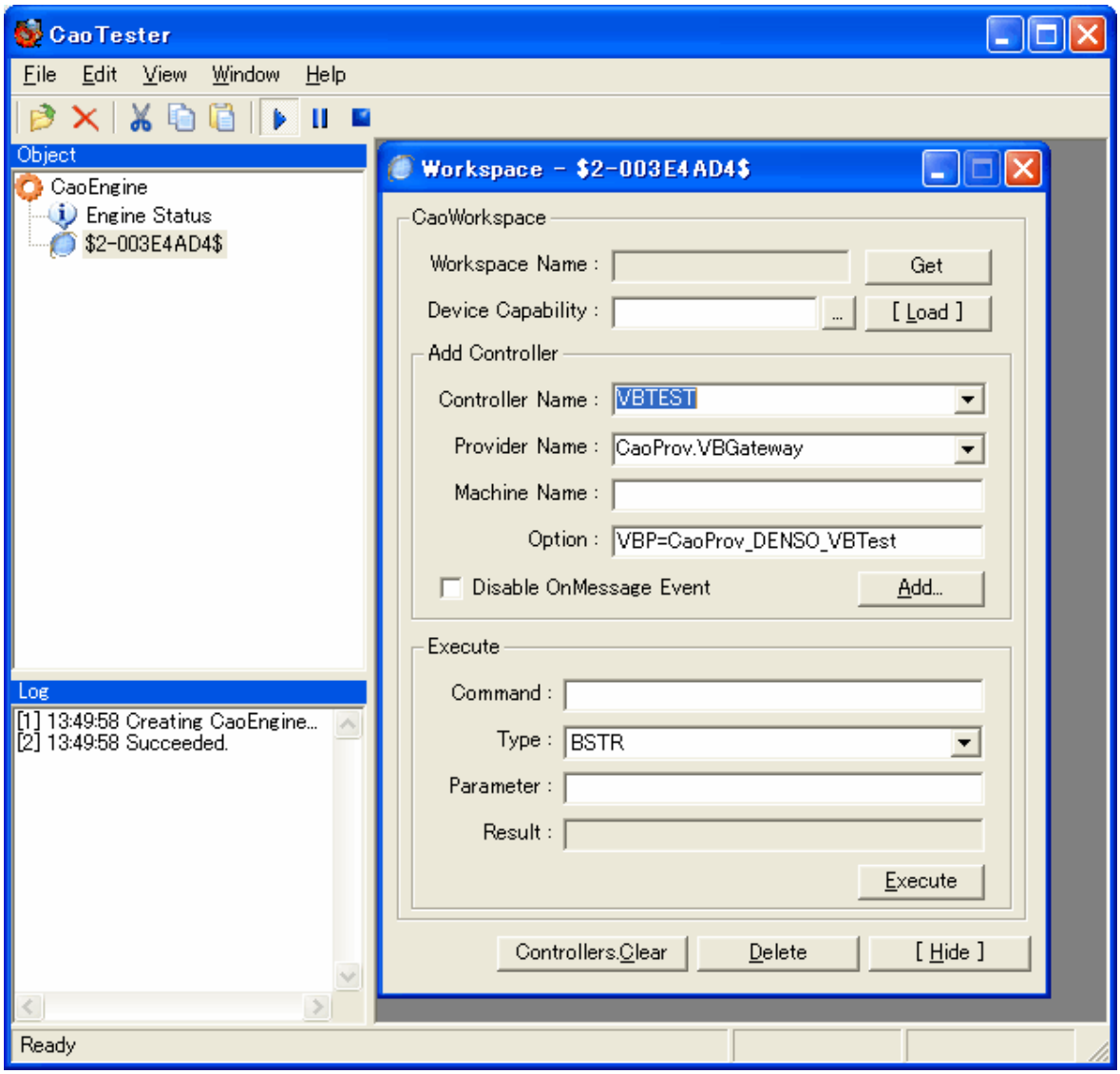


図 4-4　デバッグテスト例

5. プログラミング Tips

5.1. OnMessage イベント

5.1.1. 手順

OnMessage イベントを発生させるには，以下の手順を行います．

(1) Caop4vbController_Initialize()でタイマーフラグを ON にします．

(2) Caop4vbConttoller クラスの CreateMessage 関数でメッセージオブジェクト(Caop4vbMessage)を生成します．

(3) Caop4vbMessage クラスの実装を行います．初期化関数 FinalInitialize の引数 pVal が VT_EMPTY か否かで，呼び出した先が VBGateway プロバイダ(外部)か，Caop4vbController オブジェクト(内部)かを判定しているところに注意してください．

5.1.2. サンプル

List 5-1-1 Caop4vbController

```
Private Sub ICaop4vbController_FinalConnect(pVal As Variant)

    ' タイマーフラグを ON にする．引数の 3 番目 (pVal(2)) がタイマーフラグ．
    pVal(2) = True              ‘ タイマーON

    ' Caop4vbMessage オブジェクトを生成します．
    CreateMessage “TestMsg”    ‘ メッセージ生成
End Sub
```

List 5-1-2 Caop4vbMessage

```
Private m_vntData As Variant

' メッセージの初期化処理．引数”pVal”は CreateMessage の引数となります．
Private Sub ICaop4vbMessage_FinalInitialize(pVal As Variant)
    If IsEmpty(pVal) Then
        ‘ Called from VBGateway Provider
    Else
        ‘ Called from Caop4vbController
        m_vntData = pVal
    End If
End Sub

' メッセージの実装．
Private Function ICaop4vbMessage_FinalGetValue() As Variant
    ICaop4vbMessage_FinalGetValue = m_vntData
End Function
```

5.2. 親オブジェクト情報

5.2.1. 手順

FinalInitialize で親オブジェクトの情報を取得するには, 以下の手順を行います.

(1) 親オブジェクトにパブリックメンバを追加します.

(2) FinalInitialize()の引数にある親オブジェクトからパブリックメンバを呼び出します.

5.2.2. サンプル

List 5-2-1 Caop4vbController (親オブジェクト)

```
Option Explicit
Implements CAOPROV4VB.ICaop4vbController

' 親オブジェクトでパブリックメンバの宣言.
Public TempData As String

' パブリックメンバの初期化.
Private Sub ICaop4vbController_FinalConnect(pVal As Variant)
    TempData = "This is VB Test!!"
End Sub
```

List 5-2-2 Caop4vbVariable (子オブジェクト)

```
Private Sub ICaop4vbVariable_FinalInitialize(pVal As Variant)

    ' 親オブジェクト（引数の２番目 pval(1)）を取得.
    Dim Ctrl As Caop4vbController
    Set Ctrl = pVal(1)

    ' 親オブジェクトのパブリックメンバから値を取得
    OutputDebugString Ctrl.TempData

End Sub
```

5.3. マイクロプロバイダ

5.3.1. 手順

マイクロプロバイダの作り方を以下に示します.

(1) 使用する CAO プロバイダをプロジェクトの参照設定から追加します.

(2) CAO プロバイダのオブジェクトを宣言します.

(3) 各メソッドを実装します.

5.3.2. サンプル

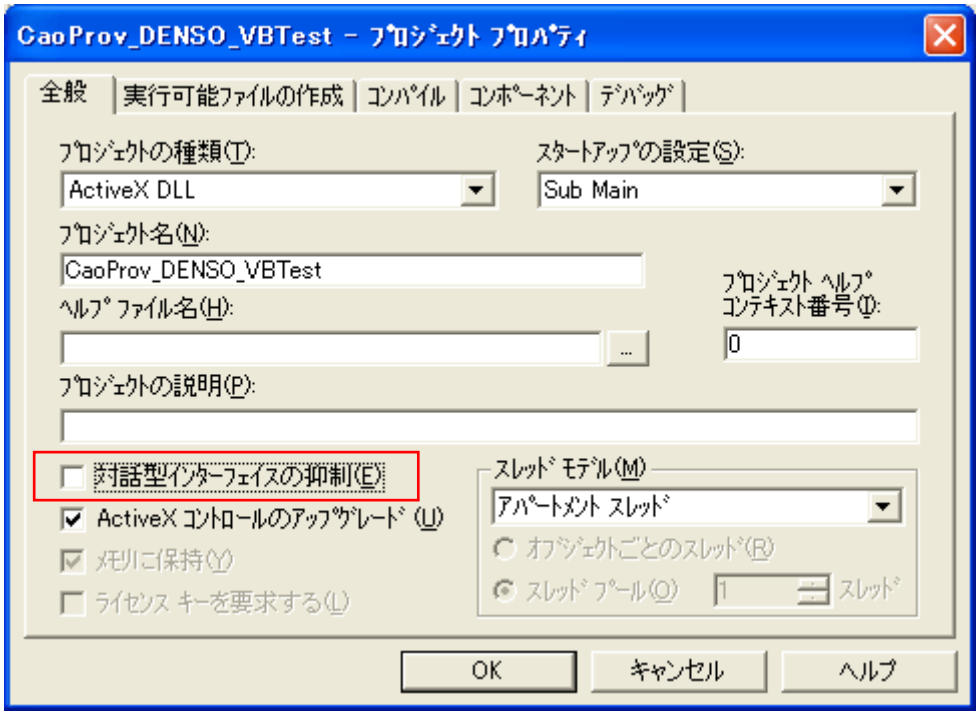
List 5-3

```
Implements CaoProv4VB.ICaop4vbController

' CAO プロバイダオブジェクトの宣言.
Private m_queMsg As New Collection
Public WithEvents m_ProvCtrl As CaoProvController

' CAO プロバイダオブジェクトの生成・初期化.
Private Sub ICaop4vbController_FinalConnect(pVal As Variant)
    ' Connect Controller
    Set m_ProvCtrl = New CaoProvController
    m_ProvCtrl.Connect "Test", "Option=Sample"

    ' Timer on
    pVal(2) = True
End Sub

' CAO プロバイダオブジェクトの切断処理.
Private Sub ICaop4vbController_FinalDisconnect()
    m_ProvCtrl.Disconnect
End Sub

' CAO プロバイダのメソッド呼び出し.
Private Function ICaop4vbController_FinalGetHelp() As String
    ICaop4vbController_FinalGetHelp = m_ProvCtrl.Help
End Function

' CAO プロバイダからのメッセージ処理.
' VB プロバイダのメッセージオブジェクトを作成し, CAO プロバイダのメッセージを格納します.
Private Sub m_ProvCtrl_OnMessage(ByVal pCaoProvMessage As _
                                 CAOPROVLib.ICaoProvMessage, _
                                 Optional ByVal lOption As Long = 0&)
    CreateMessage pCaoProvMessage
End Sub
```

5.4. モーダルダイアログの表示

5.4.1. 手順

(1) プロジェクトプロパティの全般タブで「対話型インターフェイスの抑制」をオフにします.



(2) フォーム(ダイアログ)を作成します.

(3) Form.show vbModal でモーダル表示します.

5.4.2. サンプル

List 5-4

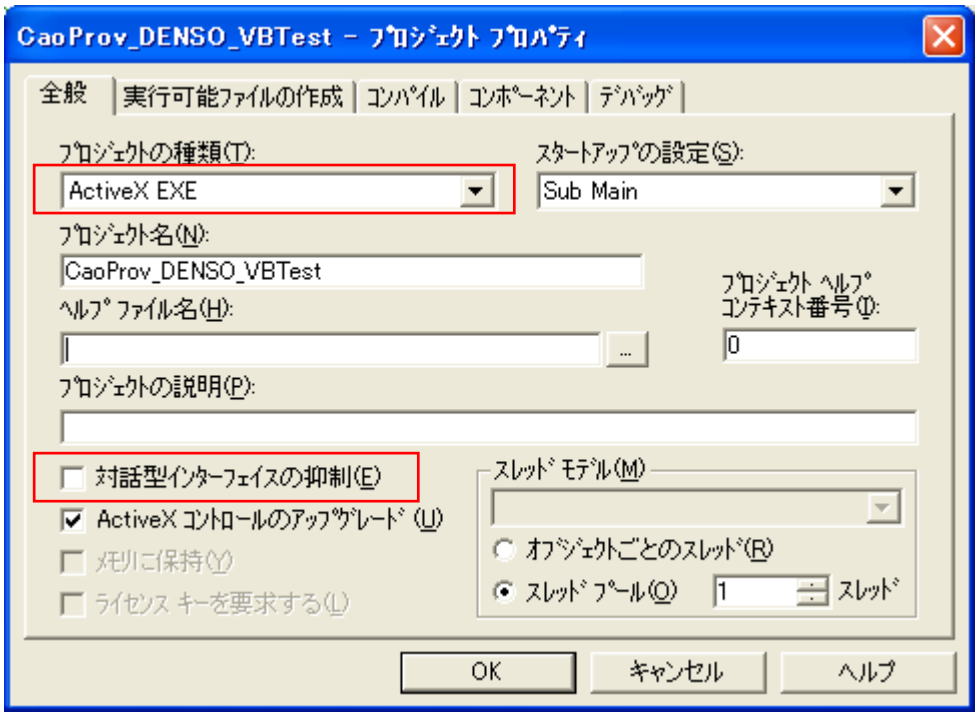
```
Private Sub ICaop4vbController_FinalConnect(pVal As Variant)

     ‘ Set initial parameters
    Load frmConnect
    frmConnect.InitParam pVal

    ’ Show a connection dialogue
    frmConnect.Show vbModal

    If (frmConnect.Connected = False) Then
        Err.Raise vbObjectError + 1, , “Connection Error!”
    End If

End Sub
```

5.5. モーダレスダイアログの表示

5.5.1. 手順

(1) プロジェクトプロパティの全般タブでプロジェクトの種類を「ActiveX　EXE」に変更します.

(2) 「対話型インターフェイスの抑制」をオフにします.



(3) フォーム(ダイアログ)を作成します.

(4) Form.Show でモーダレス表示する.

5.5.2. サンプル

List 5-5

```
Private Sub ICaop4vbController_FinalConnect(pVal As Variant)

     ‘ Set initial parameters
    Load frmConnect
    frmConnect.InitParam pVal

    ’ Show a connection dialogue
    frmConnect.Show vbModal

    If (frmConnect.Connected = False) Then
        Err.Raise vbObjectError + 1, , “Connection Error!”
    End If

End Sub
```