unitec BACnet プロバイダ
BACnet/IP 通信
Version 1.1.0

ユーザーズ ガイド

March 22, 2023

備考：本プロバイダを使用するためには，ユニテック社製の Windows 対応 BACnet ドライバ DLL が必要となります.

【改版履歴】

バージョン	日付	内容
1.0.0	2019-06-12	初版.
	2021-12-08	動作確認機種を追記 ドライバ名の記述を変更
1.0.1	2023-01-17	複数台接続時の不具合修正 ユーザーズガイドのエラーコード修正
1.1.0	2023-03-22	初期化時に時刻同期をしないように修正 時刻同期が通知されたときに PC の時刻を同期しないように修正 SendTimeSync, SendTimeSyncUnicast, SendUTCTimeSync コマンド追加

【動作確認機種】

機種	バージョン	注意事項
ICONTSimulator2004	1.0.1.0	ユニテック社製の BACnet 開発支援ツールにて動作を確認.

目次

1. はじめに

本書は，ユニテック社製 BACnet 通信 API を使用し，BACnet に接続している BACnet コントローラに対してアクセスをするプロバイダのユーザーズガ
イドです．図 1-1 が本プロバイダとデバイスの全体構成図になります．以降本プロバイダを BACnet プロバイダと呼称します．また BACnet では中央監視である「BACnet オペレータワークステーション」を「B-OWS」と呼称し，BACnet コントローラを「B-BC」と呼称します．

BACnet プロバイダは BACnet と呼ばれる，ビルディングネットワークのための通信規格をベースにユニテック社製 BACnet 通信 API を使用して，BACnet コントローラと接続します．また今回プロバイダを作成するにあたって，アズビル株式会社のホームページにて提供されている BACnet 公開仕様書[1]に記載されている動作手順を参考に作成しております．以降本書では，アズビル株式会社のホームページにて提供されている BACnet 公開仕様書を BACnet 公開仕様書と呼称いたします．本プロバイダを使用する際には，BACnet についての知識を事前に修得したうえで使用することをお勧めいたします．

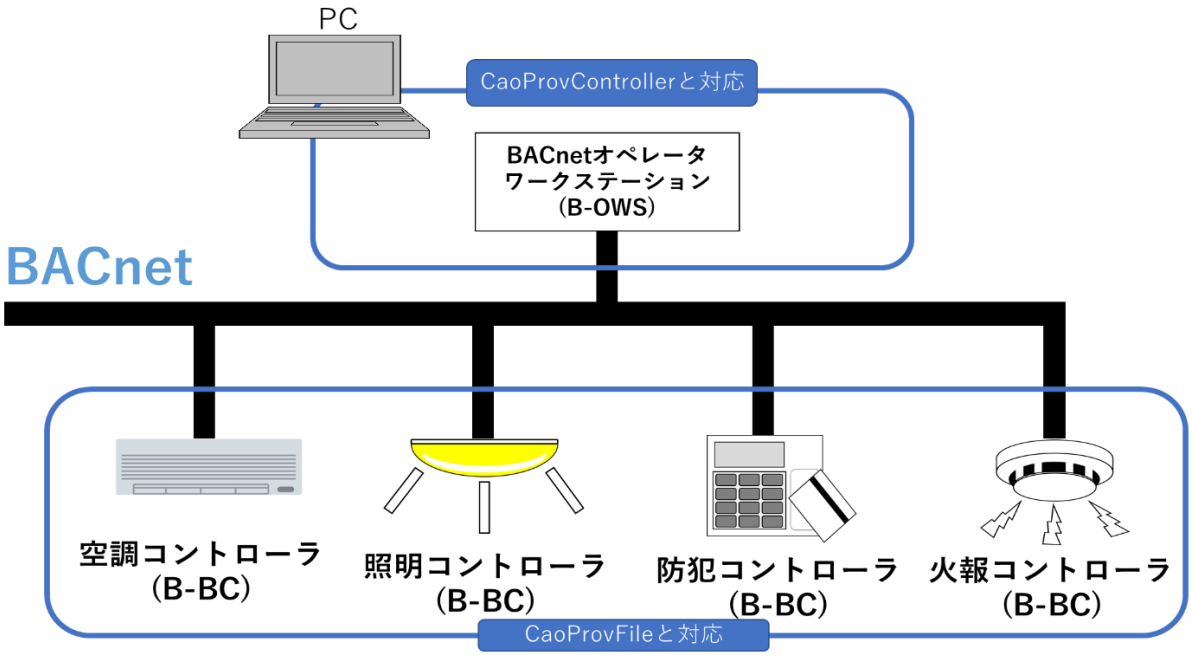


図 1-1 構成図

また，本プロバイダ及びデバイスそれぞれの対応例を図 1-2 に示します．

[1] https://www.azbil.com/jp/product/building/system/BACnet.html

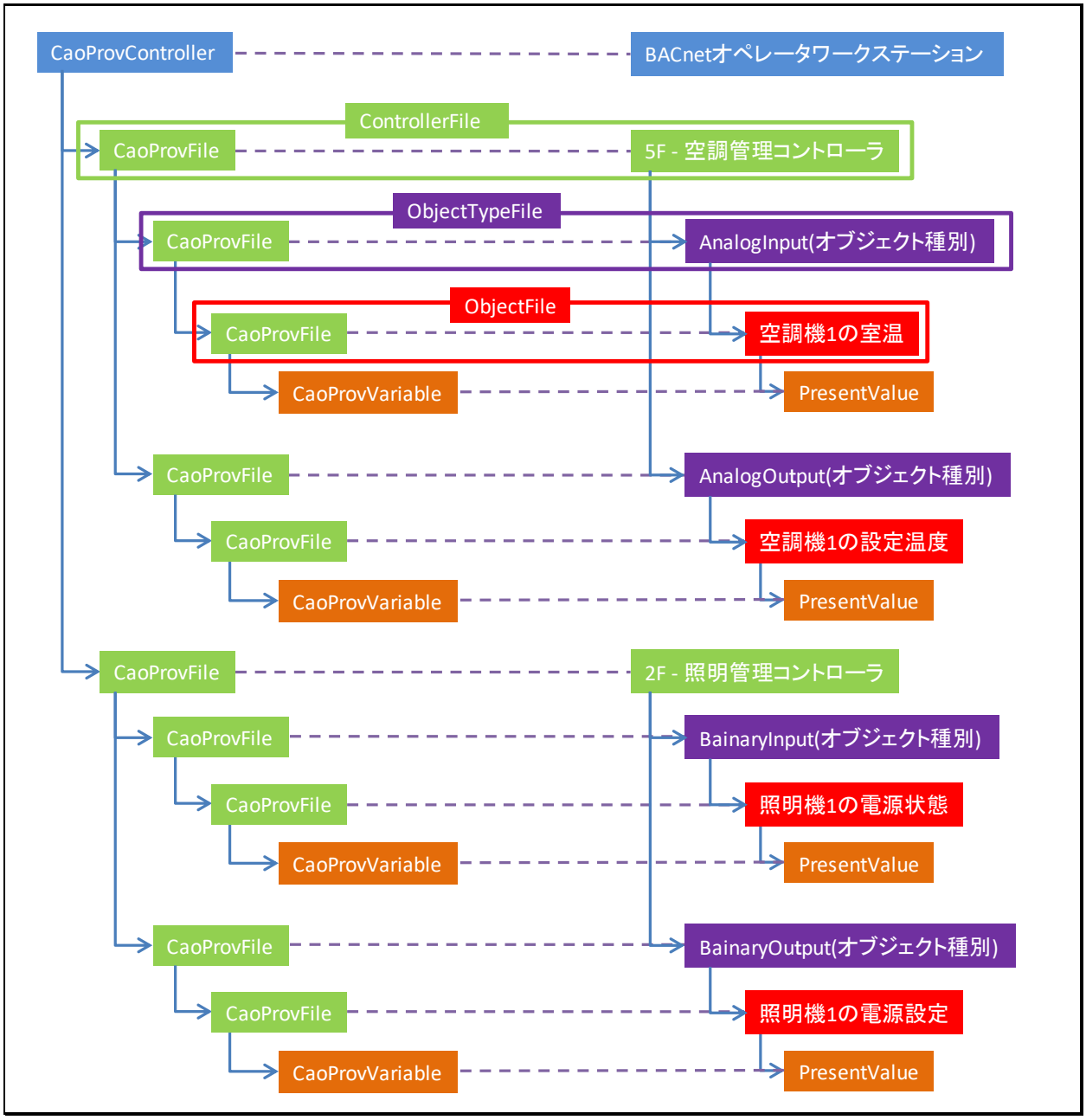


図 1-2 プロバイダの構成とデバイス情報との対応図

CaoProvController:

BACnet オペレータワークステーションに対応します.

※ ユニテック社製 BACnet 通信 API の制約上, 複数の BACnet オペレータワークステーションの追加は出来ません. ですので, 追加可能な CaoProvController は 1 つのみとなります.

CaoProvFile:

BACnet では, データを識別するための BACnet コントローラ, オブジェクトの種別, オブジェクトの 3 階層に分かれています. プロバイダでは, 各階層に対応した CaoProvFile を以下のように作成する必要があります.

ControllerFile:

BACnet でのコントローラ階層に対応する CaoProvFile です. ControllerFile は CaoProvController の直下に作成されます.

ObjectTypeFile:

BACnet で定義されるオブジェクトの種別に対応する CaoProvFile です. ObjectTypeFile は ControllerFile の直下に作成されます. ObjectTypeFile は現在以下の表 1-1 のオブジェクトを作成することが可能です. オブジェクト種別の詳細につきましては BACnet 公開仕様書を参照してください.

表 1-1 作成可能オブジェクト一覧表

オブジェクトタイプ番号	オブジェクト種別名
0	AnalogInput オブジェクト
1	AnalogOutput オブジェクト
2	AnalogValue オブジェクト
3	BinaryInput オブジェクト
4	BinaryOutput オブジェクト
5	BinaryValue オブジェクト
13	MultistateInput オブジェクト
14	MultistateOutput オブジェクト
19	MultistateValue オブジェクト
23	Accumulator オブジェクト
128	計量オブジェクト
130	電力デマンド監視オブジェクト
131	電力デマンド制御オブジェクト
132	発電機負荷制御オブジェクト

ObjectFile:

BACnet でのオブジェクト階層に対応する CaoProvFile です. ObjectFile は ObjectTypeFile の直下に作成されます.

2. アプリケーション開発のための環境セットアップ

2.1. B-BC とクライアント PC との接続

BACnet プロバイダはユニテック社製 BACnet 通信 API を使用して BACnet に参入し，接続されている B-BC と接続を行い，ユニテック社製 BACnet 通信 API は BACnet と接続する際は UDP 通信で接続を行います．6.付録 B に BACnet と通信する際の手順を記載します.

2.1.1. B-BC 監視

本プロバイダでは，BACnet のネットワークに存在している B-BC の状態を監視しています．下記に本プロバイダが行っている監視方法を記載します.

2.1.1.1. ヘルシーチェック

Who-Is/I-Am による各 B-BC の生存監視を行います．本プロバイダでは，60 秒ごとに Who-Is を送信し，各 B-BC が送信する I-Am にて生存確認を行います 150 秒以上 I-Am の送信がない B-BC に関しては非運用状態として離脱していると判断します.

2.2. PC 開発環境のセットアップ

2.2.1. 通信ライブラリの準備

BACnet プロバイダを開発 PC で使用するために，ユニテック社から提供されている Windows 対応 BACnet ドライバ DLL を事前に開発 PC に用意する必要があります．下記ファイルを，32bitOS の場合「C:¥Windows¥System32」に，64bitOS の場合「C:¥Windows¥SysWow64」に配置してください.

表 2-1 必要な通信ライブラリ

DLL	説明
BACnetDriver.dll	ユニテック社製の BACnet 通信ライブラリ

2.2.2. BACnet プロバイダの手動インストール

BACnet プロバイダを使用するためには手作業で下記レジストリ登録を行う必要があります．レジストリ登録を行う場合は，管理者権限でコマンドプロンプトを起動し，regsvr32 コマンドを実行してください．実行する際にはファイルのあるパスまで移動するか，ファイルパスを指定して実行してください.

表 2-2 BACnet プロバイダ

ファイル名	CaoProvunitecBACnet.dll
ProgID	CaoProv.unitec.BACnet
レジストリ登録	regsvr32 CaoProvunitecBACnet.dll
レジストリ登録の抹消	regsvr32 /u CaoProvunitecBACnet.dll

3. プロバイダの概要

3.1. メソッド/プロパティ一覧

表 3-1 メソッド/プロパティ一覧

カテゴリ	メソッド/プロパティ[2]		機能	参照
CaoWorkspace				
	AddController	M	コントローラに接続	P.9
CaoController				
	FileNames	P	接続可能なファイル名リストを取得	P.11
	AddFile	M	ファイル/フォルダオブジェクトの追加	P.11
	VariableNames	P	接続可能な変数名リストの取得	P.12
	AddVariable	M	変数オブジェクトの追加	P.13
	OnMessage	E	メッセージ受信イベント	P.13
	Execute	M	拡張命令の実行	P.13
CaoFile				
	FileNames	P	接続可能なファイル名リストを取得	P.14
	AddFile	M	ファイル/フォルダオブジェクトの追加	P.15
	VariableNames	P	接続可能な変数名リストの取得	P.16
	AddVariable	M	変数オブジェクトの追加	P.17
CaoVariable				
	Value	P	値の取得/設定	P.18

3.2. メソッド・プロパティ

3.2.1. CaoWorkspace クラス

3.2.1.1. AddController メソッド

CaoWorkspace に，コントローラオブジェクトを追加します．BACnet プロバイダでは，AddController メソッド実行時に渡されたパラメータを参照し，BACnet システムに参入を行います．ただし，複数台のコントローラを追加することはできません．以下に，AddController メソッドの仕様を示します．

書式

```
CaoController AddController
(
        "<コントローラ名>",          // コントローラ名(任意)
        "CaoProv.unitec.BACnet",    // プロバイダ名 (固定)
        "<マシン名>",                // プロバイダ実行マシン名(未使用)
```

[2] M:メソッド，P:プロパティ，E:イベントをそれぞれ示します．

```
        "<オプション>"                    // オプション文字列
)
```

オプション

以下にオプション文字列に指定するオプションを示します．オプション文字列は下記に示す各オプションをカンマ(,)でつなげた文字列となります．

オプション	必須	説明	値域	デフォルト値
MyIP	○	自身の IP アドレス	IPv4 形式	-----
MyPort	--	自身のポート番号	0 - 65535	47808
Broadcast	○	ブロードキャストアドレス[3]	IPv4 形式	-----
InstanceNo	○	自身のデバイスインスタンス番号 ※デバイスインスタンス番号は同一ネットワーク番号内で一意の値を設定する必要があります	0 - 4194302	-----
NetworkNo	--	ネットワーク番号	0 - 2147483646	0

使用例

```
    private CaoEngine m_caoEng;          // Engineオブジェクト
    private CaoWorkspace m_caoWs;        // Workspaceオブジェクト
    private CaoController m_caoCtrl;     // Controllerオブジェクト

    // BACnetに参入する
    private void Entry()
    {
        m_caoEng  = new CaoEngine();
        m_caoWs   = m_caoEng.AddWorkspace("NewWrks", "");
        m_caoCtrl = m_caoWs.AddController("BACnet","CaoProv.unitec.BACnet","",
                                    "MyIP = 192.168.100.5,MyPort = 47808,
                                     Broadcast = 192.168.100.255, InstanceNo = 5,
                                     NetworkNo = 0");
    }

    public Main()
    {
        // 参入
        this.Entry();
    }
```

[3] ブロードキャストアドレスとは，マルチアクセス通信ネットワークに接続された全てのデバイスがデータグラムを受信するためネットワークアドレスである．通常，ブロードキャストアドレスに送信されるメッセージは，特定のホストではなく，ネットワークに接続された全てのホストによって受信される．（*Wikipedia* より）

3.2.2. CaoController クラス

3.2.2.1. FileNames プロパティ

現在のネットワークに存在する B-BC をファイル名のリストとして取得します. 取得したファイル名は, AddFile メソッドの第一引数に使用することができます.

使用例

```
    private CaoEngine m_caoEng;          // Engineオブジェクト
    private CaoWorkspace m_caoWs;        // Workspaceオブジェクト
    private CaoController m_caoCtrl;     // Controllerオブジェクト

    // BACnetに参入する
    private void Entry()
    {
        m_caoEng  = new CaoEngine();
        m_caoWs   = m_caoEng.AddWorkspace("NewWrks", "");
        m_caoCtrl = m_caoWs.AddController("BACnet","CaoProv.unitec.BACnet","",
                                          "MyIP = 192.168.100.5,MyPort = 47808,
                                           Broadcast = 192.168.100.255, InstanceNo = 5,
                                           NetworkNo = 0");
    }

    public Main()
    {
        // 参入
        this.Entry();

        // ファイル名リスト取得
        var ctrlFileNames = m_caoCtrl.FileNames;
    }
```

3.2.2.2. AddFile メソッド

CaoController に, CaoFile オブジェクトを追加します. 以下に, AddFile メソッドの仕様を示します.

書式

```
CaoController AddFile
(
        "<ファイル名>",          // ファイル名(任意)
        "<オプション>"           // オプション文字列
)
```

オプション

以下にオプション文字列に指定するオプションを示します． ただし，FileNames で取得したファイル名を使用した場合は，自動でオプションの値が入力される為，オプションを入力しても無視されます．また，現在のネットワークに存在せず，FileNames で取得したファイル名以外を指定する場合は，必ず InstanceNo オプションを指定してください．

オプション	必須	説明	値域	デフォルト値
InstanseNo=	○	接続する BACnet コントローラのデバイスインスタンス番号を指定する．	0 - 4194302	--

使用例

```
    private CaoEngine m_caoEng;          // Engineオブジェクト
    private CaoWorkspace m_caoWs;        // Workspaceオブジェクト
    private CaoController m_caoCtrl;     // Controllerオブジェクト
    private CaoFile m_ctrlFile;          // Fileオブジェクト

    // BACnetに参入する
    private void Entry()
    {
        m_caoEng  = new CaoEngine();
        m_caoWs   = m_caoEng.AddWorkspace("NewWrks", "");
        m_caoCtrl = m_caoWs.AddController("BACnet","CaoProv.unitec.BACnet","",
                                          "MyIP = 192.168.100.5,MyPort = 47808,
                                           Broadcast = 192.168.100.255, InstanceNo = 5,
                                           NetworkNo = 0");
    }

    public Main()
    {
        // 参入
        this.Entry();

        // ファイルの追加
        m_ctrlFile = m_caoCtrl.AddFile("Dev-100", "InstanceNo = 100");
    }
```

3.2.2.3. VariableNames プロパティ

接続可能な変数名リストを取得します．取得した変数名は， AddVariable メソッドの第一引数に使用することができます．

使用例

```
    private CaoEngine m_caoEng;          // Engineオブジェクト
    private CaoWorkspace m_caoWs;        // Workspaceオブジェクト
    private CaoController m_caoCtrl;     // Controllerオブジェクト

    // BACnetに参入する
```

```
    private void Entry()
    {
        m_caoEng  = new CaoEngine();
        m_caoWs   = m_caoEng.AddWorkspace("NewWrks", "");
        m_caoCtrl = m_caoWs.AddController("BACnet","CaoProv.unitec.BACnet","",
                                          "MyIP = 192.168.100.5,MyPort = 47808,
                                           Broadcast = 192.168.100.255, InstanceNo = 5,
                                           NetworkNo = 0");
    }

    public Main()
    {
        // 参入
        this.Entry();

        // 変数名リスト取得
        var variables = m_caoCtrl.VariableNames;
    }
```

3.2.2.4. AddVariable メソッド

CaoController に変数オブジェクトを追加します. 変数名には 3.3.1 に記載した変数名を指定してください.

以下に, AddVariable の仕様を示します.

書式

```
CaoVariable AddVariable
(
        "<変数名>",             // 変数名
        "<オプション>"          // オプション文字列(省略可能)
)
```

3.2.2.5. OnMessage イベント

B-BC のエラー通知や状態の変化を OnMessage イベントとして受け取ります. 詳細は 3.4 章を参照してください.

3.2.2.6. Execute メソッド

CaoController クラスの Execute メソッドは, コマンドを実行するためのメソッドです. 各コマンドの詳細は 4 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

使用例

```
var result = this.ctrl.Execute("SendTimeSync ");
```

3.2.3. CaoFile クラス

3.2.3.1. FileNames プロパティ

AddFile 時に追加可能なファイル名リストを取得します. ControllerFile 階層のみ使用可能です. 取得した変数名は, AddFile メソッドの第一引数に使用することができます.

使用例

```
private CaoEngine m_caoEng;          // Engineオブジェクト
private CaoWorkspace m_caoWs;        // Workspaceオブジェクト
private CaoController m_caoCtrl;     // Controllerオブジェクト
private CaoFile m_ctrlFile;          // Fileオブジェクト(ControllerFile階層)

// BACnetに参入する
private void Entry()
{
    m_caoEng  = new CaoEngine();
    m_caoWs   = m_caoEng.AddWorkspace("NewWrks", "");
    m_caoCtrl = m_caoWs.AddController("BACnet","CaoProv.unitec.BACnet","",
                                      "MyIP = 192.168.100.5,MyPort = 47808,
                                       Broadcast = 192.168.100.255, InstanceNo = 5,
                                       NetworkNo = 0");
}

public Main()
{
    // 参入
    this.Entry();

    // ファイルの追加
    m_ctrlFile = m_caoCtrl.AddFile("Dev-100", "InstanceNo = 100");

    // ファイル名リスト取得
    var fileNames = m_ctrlFile.FileNames;
}
```

3.2.3.2. AddFile メソッド

各階層の CaoFile に，ファイルオブジェクトを追加します．以下に，AddFile メソッドの仕様を示します．

書式

```
CaoFile AddFile
(
        "<ファイル名>",          // ファイル名
        "<オプション>"           // オプション文字列
)
```

・ObjectTypeFile 階層の追加

ObjectTypeFile 階層を追加する AddFile 時には，<ファイル名>には特定のファイル名のみ指定できます．使用できるファイル名は表 3-2 を参照してください．

表 3-2 ControllerFile 階層で追加可能なファイル名一覧

ファイル名	対応するオブジェクトタイプ
AnalogInput	AnalogInput オブジェクト
AnalogOutput	AnalogOutput オブジェクト
AnalogValue	AnalogValue オブジェクト
BinaryInput	BinaryInput オブジェクト
BinaryOutput	BinaryOutput オブジェクト
BinaryValue	BinaryValue オブジェクト
Multi-stateInput	Multi-stateInput オブジェクト
Multi-stateOutput	Multi-stateOutput オブジェクト
Multi-stateValue	Multi-stateValue オブジェクト
Accumulator	Accumulator オブジェクト
Measurement	計量オブジェクト
PowerDemandMonitoring	電源デマンド監視オブジェクト
PowerDemandControl	電源デマンド制御オブジェクト
GeneratorLoadControl	発電機負荷制御オブジェクト

・ObjectFile 階層の追加

ObjectFile 階層を追加する AddFile 時の第一引数の<ファイル名>を任意で設定可能です．

オプション

以下に ObjectFile 階層を追加する AddFile 時に，指定するオプションを示します.

オプション	必須	説明	値域	デフォルト値
ObjectInstanceNo	○	アクセスする BACnet オブジェクトのインスタンス番号を指定します.	0－4194303	--

使用例

```
    private CaoEngine m_caoEng;          // Engineオブジェクト
    private CaoWorkspace m_caoWs;        // Workspaceオブジェクト
    private CaoController m_caoCtrl;     // Controllerオブジェクト
    private CaoFile m_ctrlFile;          // Fileオブジェクト(ControllerFile階層)
    private CaoFile m_objectTypeFile;    // Fileオブジェクト(ObjectTypeFile階層)
    private CaoFile m_objectFile;        // Fileオブジェクト(ObjectFile階層)

    // BACnetに参入する
    private void Entry()
    {
        m_caoEng  = new CaoEngine();
        m_caoWs   = m_caoEng.AddWorkspace("NewWrks", "");
        m_caoCtrl = m_caoWs.AddController("BACnet","CaoProv.unitec.BACnet","",
                                    "MyIP = 192.168.100.5,MyPort = 47808,
                                     Broadcast = 192.168.100.255, InstanceNo = 5,
                                     NetworkNo = 0");
    }

    public Main()
    {
        // 参入
        this.Entry();

        // ファイルの追加(ControllerFile階層)
        m_ctrlFile = m_caoCtrl.AddFile("Dev-100", "InstanceNo = 100");

        // ファイルの追加(ObjectTypeFile階層)
        m_objectTypeFile= m_ctrlFile.AddFile("AnalogInput", "");

        // ファイルの追加(ObjectFile階層)
        m_objectFile = m_objectTypeFile.AddFile("AIO", "ObjectInstanceNo=0");
    }
```

3.2.3.3. VariableNames プロパティ

接続可能な変数名リストを取得します. ObjectFile 階層のみ使用可能です. ObjectTypeFile クラスで指定したオブジェクトタイプによって取得する変数名リストが異なります. 取得した変数名は, AddVariable メソッドの第一引数に使用することができます.

使用例

```
    private CaoEngine m_caoEng;          // Engineオブジェクト
    private CaoWorkspace m_caoWs;        // Workspaceオブジェクト
    private CaoController m_caoCtrl;     // Controllerオブジェクト
    private CaoFile m_ctrlFile;          // Fileオブジェクト(ControllerFile階層)
    private CaoFile m_objectTypeFile;    // Fileオブジェクト(ObjectTypeFile階層)
    private CaoFile m_objectFile;        // Fileオブジェクト(ObjectFile階層)

    // BACnetに参入する
    private void Entry()
    {
        m_caoEng  = new CaoEngine();
        m_caoWs   = m_caoEng.AddWorkspace("NewWrks", "");
        m_caoCtrl = m_caoWs.AddController("BACnet","CaoProv.unitec.BACnet","",
                                          "MyIP = 192.168.100.5,MyPort = 47808,
                                           Broadcast = 192.168.100.255, InstanceNo = 5,
                                           NetworkNo = 0");
    }

    public Main()
    {
        // 参入
        this.Entry();

        // ファイルの追加(ControllerFile階層)
        m_ctrlFile = m_caoCtrl.AddFile("Dev-100", "InstanceNo = 100");

        // ファイルの追加(ObjectTypeFile階層)
        m_objectTypeFile= m_ctrlFile.AddFile("AnalogInput", "");

        // ファイルの追加(ObjectFile階層)
        m_objectFile = m_objectTypeFile.AddFile("AIO", "ObjectInstanceNo=0");

        // 変数名リスト取得
        var fileVariables = m_objectFile.VariableNames;
    }
```

3.2.3.4. AddVariable メソッド

CaoFile に変数オブジェクトを追加します. ObjectFile 階層のみ使用可能です. 変数名には 3.3.2 に示す変数名を指定して下しい.

以下に, AddVariable の仕様を示します.

書式

```
CaoVariable AddVariable
(
        "<変数名>",              // 変数名
```

```
        "<オプション>"          // オプション文字列(省略可能)
)
```

3.2.4. CaoVariable クラス

3.2.4.1. Value プロパティ

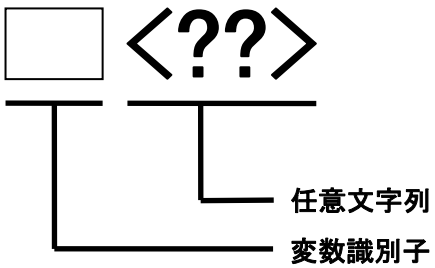
接続する B-BC からデータを取得/設定します. 変数名によって動作が異なります. 詳細は, 3.3.変数一覧を参照してください.

3.3. 変数一覧

各クラスで使用可能な変数一覧を定義します. なお変数は, CaoVariable クラスのオブジェクトを指します. 複数変数を登録(オプションのみ変更したい場合等に有用)するために任意の文字列を付与することが可能です.

変数名に任意文字列を付与するための書式を以下に示します.

複数変数共通指定書式



3.3.1. CaoController クラス変数

変数名	説明	Value		参照
		get	put	
@VERSION	DLL バージョンを取得します.	○	-	P.18

3.3.1.1. @VERSION

DLL のバージョンの取得をします.

データ型

型説明	
VT_BSTR	DLL のバージョンを取得します. *.*.*

使用例

```
    private CaoEngine m_caoEng;                    // Engineオブジェクト
    private CaoWorkspace m_caoWs;                  // Workspaceオブジェクト
    private CaoController m_caoCtrl;               // Controllerオブジェクト
    private CaoVariable m_ctrlVariable;            // Variableオブジェクト

    // BACnetに参入する
    private void Entry()
    {
        m_caoEng  = new CaoEngine();
        m_caoWs   = m_caoEng.AddWorkspace("NewWrks", "");
        m_caoCtrl = m_caoWs.AddController("BACnet","CaoProv.unitec.BACnet","",
                                          "MyIP = 192.168.100.5,MyPort = 47808,
                                           Broadcast = 192.168.100.255, InstanceNo = 5,
                                           NetworkNo = 0");
    }

    public Main()
    {
        // 参入
        this.Entry();

        // 変数追加
        m_ctrlVariable = m_caoCtrl.AddVariable("@VERSION");
        // 値取得
        string value = m_ctrlVariable.Value;
    }
```

3.3.2. CaoFile クラス変数

以下に，ObjectFile 階層にて使用する変数名一覧です.

変数名	説明	Value		参照
		get	put	
@PRESENTVALUE	オブジェクトの現在値を取得します.	○	-	P.19
@STATUSFLAGS	オブジェクトの状態を取得します.	○	-	P.21
PRIORITYVALUE<??>	書込み可能オブジェクトの PresentValue プロパティに Priority(優先度)を指定し，アクセスします.	○	○	P.22

3.3.2.1. @PRESENTVALUE

CaoFile クラスの情報をもとに，BACnet オブジェクトの現在値を取得します.

データ型

型説明	
VT_VARIANT	BACnet オブジェクトの現在値を取得します. BACnet オブジェクト種別によって受け取るデータ型が異なります. 下記に各オブジェクト種別に対応したデータ型を表 3-3 にて記載しております.

表 3-3 オブジェクト種別ごとの取得データ型一覧表

オブジェクト種別	取得データ型	値域
AnalogInput オブジェクト	VT_R4	データ型に依存します
AnalogOutput オブジェクト	VT_R4	データ型に依存します
AnalogValue オブジェクト	VT_R4	データ型に依存します
BinaryInput オブジェクト	VT_UI4	0 - 1
BinaryOutput オブジェクト	VT_UI4	0 - 1
BinaryValue オブジェクト	VT_UI4	0 - 1
MultistateInput オブジェクト	VT_UI4	対象オブジェクトが保持する「Number Of State プロパティ」の数値により異なります.
MultistateOutput オブジェクト	VT_UI4	対象オブジェクトが保持する「Number Of State プロパティ」の数値により異なります.
MultistateValue オブジェクト	VT_UI4	対象オブジェクトが保持する「Number Of State プロパティ」の数値により異なります.
Accumulator オブジェクト	VT_UI4	データ型に依存します
計量オブジェクト	VT_UI4	データ型に依存します
電力デマンド監視オブジェクト	VT_UI4	データ型に依存します
電力デマンド制御オブジェクト	VT_UI4	データ型に依存します
発電機負荷制御オブジェクト	VT_UI4	データ型に依存します

使用例

```
    private CaoEngine m_caoEng;          // Engineオブジェクト
    private CaoWorkspace m_caoWs;        // Workspaceオブジェクト
    private CaoController m_caoCtrl;     // Controllerオブジェクト
    private CaoFile m_ctrlFile;          // Fileオブジェクト(ControllerFile階層)
    private CaoFile m_objectTypeFile;    // Fileオブジェクト(ObjectTypeFile階層)
    private CaoFile m_objectFile;        // Fileオブジェクト(ObjectFile階層)
    private CaoVariable m_fileVariable; // Variableオブジェクト

    // BACnetに参入する
    private void Entry()
    {
```

```
        m_caoEng  = new CaoEngine();
        m_caoWs   = m_caoEng.AddWorkspace("NewWrks", "");
        m_caoCtrl = m_caoWs.AddController("BACnet","CaoProv.unitec.BACnet","",
                                          "MyIP = 192.168.100.5,MyPort = 47808,
                                           Broadcast = 192.168.100.255, InstanceNo = 5,
                                           NetworkNo = 0");
    }

    public Main()
    {
        // 参入
        this.Entry();

        // ファイルの追加(ControllerFile階層)
        m_ctrlFile = m_caoCtrl.AddFile("Dev-100", "InstanceNo = 100");

        // ファイルの追加(ObjectTypeFile階層)
        m_objectTypeFile= m_ctrlFile.AddFile("AnalogInput", "");

        // ファイルの追加(ObjectFile階層)
        m_objectFile = m_objectTypeFile.AddFile("AIO", "ObjectInstanceNo=0");

        // 変数追加
        m_fileVariable = m_objectFile.AddVariable("@PRESENTVALUE");
        // 値取得
        float value = m_fileVariable.value;
    }
```

3.3.2.2. @STATUSFLAGS

CaoFile クラスの情報をもとに，BACnet オブジェクトの状態値を取得します.

データ型

型説明	
VT_UI4	BACnet オブジェクトの状態値を取得します. 状態値は下位 4 ビットをビットフラグとして表現されます. フラグが Ture の場合，対象のビットが 1 になります. 下位 1 ビット目：アラーム状態 下位 2 ビット目：機器の状態 下位 3 ビット目：無効状態 下位 4 ビット目：メンテナンス状態

使用例

```
    private CaoEngine m_caoEng;         // Engineオブジェクト
    private CaoWorkspace m_caoWs;       // Workspaceオブジェクト
```

```
    private CaoController m_caoCtrl;    // Controllerオブジェクト
    private CaoFile m_ctrlFile;         // Fileオブジェクト(ControllerFile階層)
    private CaoFile m_objectTypeFile;   // Fileオブジェクト(ObjectTypeFile階層)
    private CaoFile m_objectFile;       // Fileオブジェクト(ObjectFile階層)
    private CaoVariable m_fileVariable; // Variableオブジェクト

    // BACnetに参入する
    private void Entry()
    {
        m_caoEng  = new CaoEngine();
        m_caoWs   = m_caoEng.AddWorkspace("NewWrks", "");
        m_caoCtrl = m_caoWs.AddController("BACnet","CaoProv.unitec.BACnet","",
                                          "MyIP = 192.168.100.5,MyPort = 47808,
                                           Broadcast = 192.168.100.255, InstanceNo = 5,
                                           NetworkNo = 0");
    }

    public Main()
    {
        // 参入
        this.Entry();

        // ファイルの追加(ControllerFile階層)
        m_ctrlFile = m_caoCtrl.AddFile("Dev-100", "InstanceNo = 100");

        // ファイルの追加(ObjectTypeFile階層)
        m_objectTypeFile = m_ctrlFile.AddFile("AnalogInput", "");

        // ファイルの追加(ObjectFile階層)
        m_objectFile = m_objectTypeFile.AddFile("AI0", "ObjectInstanceNo=0");

        // 変数追加
        m_fileVariable = m_objectFile.AddVariable("@STATUSFLAGS");
        // 値取得
        ulong value = m_fileVariable.value;
    }
```

3.3.2.3. PRIORITYVALUE<??>

オプションで指定する優先順位に格納されている値を取得/設定します. ただし, PriorityArray プロパティ(優先順位テーブル)を保持しているオブジェクトのみに対応しております.対応オブジェクトは表 3-4 を参照してください. また, 下記に優先順位テーブルのプロパティについての説明を記載しています.

表 3-4 PRIORITYVALUE 対応オブジェクト

オブジェクト種別
AnalogOutput オブジェクト

オブジェクト種別
AnalogValue オブジェクト
BinaryOutput オブジェクト
BinaryValue オブジェクト
MultistateOutput オブジェクト
MultistateValue オブジェクト

PriorityArray プロパティ(優先順位テーブル)

PriorityArray プロパティは命令優先順位を設定するためのプロパティです. 1 から 16 までの優先順位を設定することができ, 数字が小さいと優先度高く 1 が最上位です.

PriorityArray プロパティに優先度の異なる複数の値が設定されている場合は, 優先度が一番高い設定が有効になり, 現在値に反映されます.

命令優先順位を設定する際の運用方式として 16 レベル命令優先順位方式と固定レベル命令優先順位方式の 2 つあります. AnalogOutput オブジェクトのルールは固定レベル命令優先順位方式のみです. 運用方式の詳細についてはアズビル株式会社の BACnet 公開仕様書を参照してください.

命令優先順位の値を空にする際は, 設定する優先順位に NULL を設定することで, 値を空にすることができます.

オプション

以下に AddVariable 時に指定するオプションを示します.

オプション	必須	説明	値域	デフォルト値
Priority=	--	優先順位を指定します. 値が小さい方と優先順位が高く設定されます.	1 - 16	8

データ型

型説明	
VT_UI4 or VT_R4 or VT_NULL	BACnet オブジェクトの現在値を取得/設定します. 取得/設定する BACnet オブジェクト種別によってデータ型が異なります. 各オブジェクトの対応データ型は表 3-3 を参照してください. 指定した優先度に値が格納されていない場合, 取得時のデータが VT_NULL となり, VT_NULL 設定時は指定した優先度に格納されている値を NULL で初期化します.

使用例

```
private CaoEngine m_caoEng;          // Engineオブジェクト
private CaoWorkspace m_caoWs;        // Workspaceオブジェクト
private CaoController m_caoCtrl;     // Controllerオブジェクト
private CaoFile m_ctrlFile;          // Fileオブジェクト(ControllerFile階層)
private CaoFile m_objectTypeFile;    // Fileオブジェクト(ObjectTypeFile階層)
```

```
    private CaoFile m_objectFile;       // Fileオブジェクト(ObjectFile階層)
    private CaoVariable m_fileVariable; // Variableオブジェクト

    // BACnetに参入する
    private void Entry()
    {
        m_caoEng  = new CaoEngine();
        m_caoWs   = m_caoEng.AddWorkspace("NewWrks", "");
        m_caoCtrl = m_caoWs.AddController("BACnet","CaoProv.unitec.BACnet","",
                                          "MyIP = 192.168.100.5,MyPort = 47808,
                                           Broadcast = 192.168.100.255, InstanceNo = 5,
                                           NetworkNo = 0");
    }

    public Main()
    {
        // 参入
        this.Entry();

        // ファイルの追加(ControllerFile階層)
        m_ctrlFile = m_caoCtrl.AddFile("Dev-100", "InstanceNo = 100");

        // ファイルの追加(ObjectTypeFile階層)
        m_objectTypeFile = m_ctrlFile.AddFile("AnalogInput", "");

        // ファイルの追加(ObjectFile階層)
        m_objectFile = m_objectTypeFile.AddFile("AIO", "ObjectInstanceNo=0");

        // 優先度変数追加
        m_fileVariable = this.m_objectFile.AddVariable("PRIORITYVALUE_8", null);

        // 値を取得
        float value = m_fileVariable.Value;
    }
```

3.4. イベント一覧

コントローラのエラー通知や状態の変化を OnMessage イベントとして受け取ることが可能です.

Number	説明
0	メッセージ処理中にエラーが発生しました. データとしてエラーコードを保持します.
1	BACnet システム内の BACnet コントローラの状態が変化しました. データ型については 3.4.1.1 を参照してください.

Number	説明
2	COV 通知を受信しました. データ型については 3.4.1.2 を参照してください.
3	Event 通知を受信しました. データ型については 3.4.1.3 を参照してください.

3.4.1. 発生したメッセージのデータ内容詳細

以下に発生したメッセージのデータの詳細を説明します.

3.4.1.1. BACnet コントローラの状態変化

状態が変化した BACnet コントローラの現在の状態と前回の状態を VARIANT 型として扱います.

データ型

型説明		
VT_ARRAY | VT_VARIANT		
0	VT_UI4	状態が変化した BACnet コントローラのデバイスインスタンス番号
1	VT_I4	前回の状態 0: 運用可能 1: 運用上の読み取り専用 2: ダウンロードが必要 3: ダウンロード中 4: 非運用 5: バックアップ中 6: 未定義
2	VT_I4	現在の状態 0: 運用可能 1: 運用上の読み取り専用 2: ダウンロードが必要 3: ダウンロード中 4: 非運用 5: バックアップ中 6: 未定義

3.4.1.2. COV 通知のデータ

COV 通知で受け取ったデータを VARIANT 型で扱います.

データ型

型説明			
VT_ARRAY | VT_VARIANT			
0	VT_UI4		プロセス ID
1	VT_UI4		通知元の BACnet コントローラのデバイスインスタンス番号
2	VT_UI4		オブジェクトタイプ番号
3	VT_UI4		オブジェクトインスタンス番号
4	VT_UI4		残り時間
5	VT_UI4		プロパティの個数
6	VT_ARRAY | VT_VARIANT		
	i	VT_VARIANT	3.3.2.1 と同様のデータ型でプロパティのデータを取得

i：プロパティの個数分

3.4.1.3. Event 通知のデータ

Event 通知で受け取ったデータを VARIANT 型で扱います.

データ型

型説明					
VT_ARRAY | VT_VARIANT					
0	VT_UI4				プロセス ID
1	VT_UI4				通知元の BACnet コントローラのデバイスインスタンス番号
2	VT_UI4				オブジェクトタイプ番号
3	VT_UI4				オブジェクトインスタンス番号
4	VT_ARRAY | VT_VARIANT				タイムスタンプ
	0	VT_I4			タイプ 0. Time 1. Unsigned 2. Datatime
	1	VT_ARRAY | VT_I4			時刻(タイプが 0 の場合)
		0	VT_I4		時
		1	VT_I4		分
		2	VT_I4		秒
		3	VT_I4		1/100 秒
	1	VT_UI4			シーケンス番号(タイプが 1 の場合)
	1	VT_ARRAY | VT_VARIANT			
		0	VT_ARRAY | VT_I4		日付(タイプが 2 の場合)
			0	VT_I4	年
			1	VT_I4	月
			2	VT_I4	日

型説明					
			3	VT_I4	週 1. Monday 2. Tuesday 3. Wednesday 4. Thursday 5. Friday 6. Saturday 7. Sunday
		1	VT_ARRAY | VT_I4		時刻
			0	VT_I4	時
			1	VT_I4	分
			2	VT_I4	秒
			3	VT_I4	1/100 秒
5	VT_UI4				NotificationClass のインスタンス番号
6	VT_UI4				優先順位
7	VT_I4				イベントタイプ
8	VT_VARIANT | VT_ARRAY				メッセージテキスト
	0	VT_I4			文字コード 0. ANSIX3.4 1. DBCS 2. JISC6226 3. ISO10646(UCS-4) 4. ISO10646(UCS-2) 5. ISO-8859-1
	1	VT_I4			コードページ ※文字コードの値が 1 以外なら EMPTY
	2	VT_I4			文字列バイト長
	3	VT_BSTR			メッセージ
9	VT_I4				通知タイプ 0. アラーム 1. イベント 2. 了承要求

型説明		
10	VT_I4	了承要求 0. FALSE 1. TURE ※負の値は受け取っていないことを意味します.
11	VT_I4	前回の状態 0. 正常 1. 機器故障 2. 異常状態 3. 上限異常 4. 下限異常 5. 生命安全アラーム ※負の値は受け取っていないことを意味します.
12	VT_I4	現在の状態 0. 正常 1. 機器故障 2. 異常状態 3. 上限異常 4. 下限異常 5. 生命安全アラーム
13	VT_I4	通知パラメータタイプ 1. 状態変化 2. 値の変化 3. コマンド失敗 4. フローティングリミット 5. 範囲外 10.バッファレディ 11.符号なし範囲 ※負の値は受け取っていないことを意味します. 通知パラメータタイプが負の場合[14]は VT_EMPTY とします.
14	VT_ARRAY | VT_VARIANT	通知パラメータタイプが 1 の場合

型説明				
	0	VT_UI4		新しいステータスタイプ 0. ブール値 1. バイナリ 2. イベントタイプ 3. 極性 4. プログラム変更 5. プログラム状態 6. 故障の理由 7. 信頼性 8. 状態 9. 機器の状態 10. 単位 11. 符号なし値 12. 生命安全モード 13. 生命安全の状態 14. 再起動の理由
	1	VT_UI4		上記のステータスタイプに対応するステータスデータ
	2	VT_ARRAY | VT_I4		
		0	VT_I4	アラーム状態
		1	VT_I4	機器の状態
		2	VT_I4	無効状態
		3	VT_I4	メンテナンス状態
14	VT_ARRAY | VT_VARIANT			通知パラメータタイプが 2 の場合
	0	VT_UI4		タイプ 0. BitString 1. 実数
	1	VT_VARIANT		データ ※タイプによって異なる 0.VT_UI4 1.VT_R4
	2	VT_ARRAY | VT_I4		
		0	VT_I4	アラーム状態
		1	VT_I4	機器の状態
		2	VT_I4	無効状態
		3	VT_I4	メンテナンス状態

型説明				
14	VT_ARRAY | VT_VARIANT			通知パラメータタイプが 3 の場合
	0	VT_VARIANT		コマンド値 ※受け取るデータ型に応じてデータ型が変化する
	1	VT_ARRAY | VT_I4		
		0	VT_I4	アラーム状態
		1	VT_I4	機器の状態
		2	VT_I4	無効状態
		3	VT_I4	メンテナンス状態
	2	VT_VARIANT		設備の状態 ※受け取るデータ型に応じてデータ型が変化する
14	VT_ARRAY | VT_VARIANT			通知パラメータタイプが 4 の場合
	0	VT_R4		参照値
	1	VT_R4		設定値
	2	VT_R4		エラー制限値
	3	VT_ARRAY | VT_I4		
		0	VT_I4	アラーム状態
		1	VT_I4	機器の状態
		2	VT_I4	無効状態
		3	VT_I4	メンテナンス状態
14	VT_ARRAY | VT_VARIANT			通知パラメータタイプが 5 の場合
	0	VT_R4		超過値
	1	VT_R4		閾値
	2	VT_R4		超過制限値
	3	VT_ARRAY | VT_I4		
		0	VT_I4	アラーム状態
		1	VT_I4	機器の状態
		2	VT_I4	無効状態
		3	VT_I4	メンテナンス状態
14	VT_ARRAY | VT_VARIANT			通知パラメータタイプが 10 の場合
	0	VT_ARRAY | VT_VARIANT		
		0	VT_UI4	デバイスインスタンス番号
		1	VT_UI4	オブジェクトタイプ番号
		2	VT_UI4	オブジェクトインスタンス番号
		3	VT_I4	プロパティ番号

型説明				
		4	VT_I4	配列数 ※負の値は配列数を使用しないことを意味します.
	1	VT_UI4		前回の通知
	2	VT_UI4		今回の通知
14	VT_ARRAY | VT_VARIANT			通知パラメータタイプが 11 の場合
	0	VT_UI4		超過値
	1	VT_UI4		超過制限値
	2	VT_ARRAY | VT_I4		
		0	VT_I4	アラーム状態
		1	VT_I4	機器の状態
		2	VT_I4	無効状態
		3	VT_I4	メンテナンス状態

4. コマンドリファレンス

4.1. CaoController クラス

表 4-1 CaoController クラス コマンド一覧

コマンド	機能	ページ
SendTimeSync	時刻同期要求を同報通信で送信します.	32
SendTimeSyncUnicast	時刻同期要求を指定したデバイスに送信します.	32
SendUTCTimeSync	UTC 時刻同期要求を同報通信で送信します.	33

4.1.1. SendTimeSync コマンド

時刻同期要求を同報通信で送信します.
同期する時刻はローカル時間で送信されます.

書式　SendTimeSync()

戻り値　:　なし.

使用例

```
this.ctrl.Execute("SendTimeSync", ":START");
```

4.1.2. SendTimeSyncUnicast コマンド

時刻同期要求を指定したデバイスに送信します.

書式　SendTimeSyncUnicast(<Device ID>)

Device ID　:　[in] 送信先のデバイス ID. (VT_UI4)
戻り値　:　なし.

使用例

```
this.ctrl.Execute("SendTimeSyncUnicast", "1");
```

4.1.3. SendUTCTimeSync コマンド

時刻同期要求を同報通信で送信します.
同期する時刻は協定世界時で送信されます.

書式　SendUTCTimeSync(<command>)

戻り値　:　なし.

使用例

```
this.ctrl.Execute("SendUTCTimeSync");
```

5. BACnet プロバイダによるプログラミング

BACnet プロバイダでは，以下の手順でクライアント PC を BACnet に参入することができます.

- CaoEngine の作成
- CaoWorkspace の作成
- CaoController の作成

BACnet に参入した後は,CaoFile を 3 階層生成し,CaoVariabe オブジェクトを生成することで,BACnet コントローラの情報にアクセスすることができます.

5.1. オブジェクトの優先順位に格納されている値を取得/設定するサンプルプログラミング

ここでは例として AnalogOutput オブジェクトの優先順位に格納されている値を取得/設定するサンプルプログラムを示します．表 5-1 にサンプルプログラムの要件を，図 5-1 にサンプルプログラムの流れをそれぞれ記述しています.

表 5-1 サンプルプログラムの要件

要件	説明
自身の設定	自身の IP アドレスは 192.168.100.5
	ポート番号は 47808
	ブロードキャストアドレスは 192.168.100.255
	自身のインスタンス番号は 5
	ネットワーク番号は 0
処理内容	AnalogOutput オブジェクトの優先順位に格納されている値を読込む.
	AnalogOutput オブジェクトの優先順位に格納されている値に取得した値+1 を書き込む.

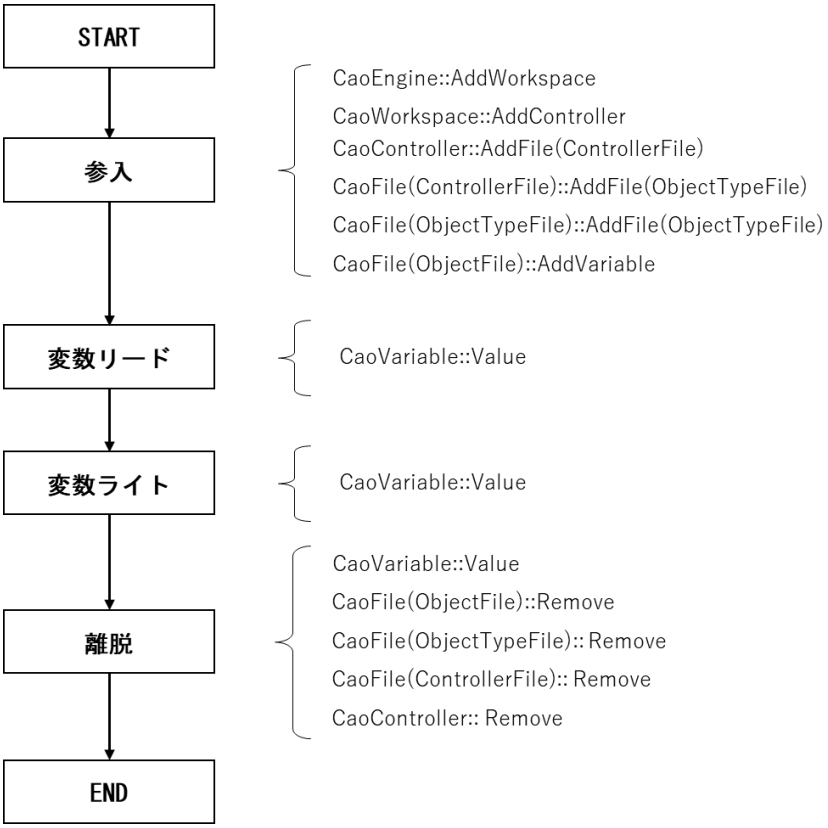


図 5-1 AnalogInput オブジェクトの現在値取得の流れ

以降の節から具体的なコードを示します.

5.1.1. サンプルプログラム

以下にサンプルプログラムの全体像を示します.

Sample GetPutPriorityValue.cs

```
// オブジェクト
private CaoEngine m_caoEng;
private CaoWorkspace m_caoWs;
private CaoController m_caoCtrl;
private CaoFile m_ctrlFile;
private CaoFile m_objectTypeFile;
private CaoFile m_objectFile;
private CaoVariable m_priorityValue;

public GetPutPriorityValue()
{
    // 参入
    this.Entry();
```

```
    // CaoFile 追加
    this.AddCaoFiles();

    // 優先度変数追加
    this.m_priorityValue = this.m_objectFile.AddVariable("PRIORITYVALUE_8", null);

    // 値
    float value;

    // 値を取得
    value = this.m_priorityValue.Value;
    // 取得データに 1 を加算
    value += 1;

    // 値を設定
    this.m_priorityValue.Value = value;

    // 再取得
    value = this.m_priorityValue.Value;

    // CaoFile(ObjectFile 階層)から CaoVariable を削除
    this.m_objectFile.Variables.Remove(this.m_priorityValue.Index);

    // CaoVariable を消去
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_priorityValue);
    this.m_priorityValue = null;

    // CaoFile 解放
    this.RemoveCaoFiles();

    // 離脱
    this.BreakAway();
}

// BACnet に参入する
private void Entry()
{
```

```
    this.m_caoEng = new CaoEngine();
    this.m_caoWs = this.m_caoEng.AddWorkspace("NewWrks", "");
    this.m_caoCtrl = this.m_caoWs.AddController("BACnet",
                                                "CaoProv.unitec.BACnet",
                                                "",
                                                "MyIP=192.168.100.5,MyPort=47808,
                                                  Broadcast=192.168.100.255,InstanceNo=5,
                                                  NetworkNo=0");
}

// BACnet から離脱する
protected void BreakAway()
{
    if (this.m_caoCtrl != null)
    {
        // CaoWorkspace から CaoController を削除
        this.m_caoWs.Controllers.Remove(this.m_caoCtrl.Name);
        // CaoController の消去
        System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoCtrl);
        this.m_caoCtrl = null;
    }
    // CaoEngine から CaoWorkspace を削除
    this.m_caoEng.Workspaces.Remove(this.m_caoWs.Name);
    // CaoWorkspace を消去
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoWs);
    this.m_caoWs = null;
    // CaoEngine を削除
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoEng);
    this.m_caoEng = null;
}

// CaoFile 追加
private void AddCaoFiles()
{
    this.m_ctrlFile = this.m_caoCtrl.AddFile("DEv-100", "InstanceNo=100");
    this.m_objectTypeFile = this.m_ctrlFile.AddFile("AnalogOutput", "");
    this.m_objectFile = this.m_objectTypeFile.AddFile("AO0", "ObjectInstanceNo=0");
```

```
}

// CaoFile 解放
private void RemoveCaoFiles()
{
    if (this.m_objectFile != null)
    {
        // CaoFile(ObjectTypeFile 階層)から CaoFile(ObjectFile 階層)を削除
        this.m_objectTypeFile.Files.Remove(this.m_objectFile.Index);
        // CaoFile(ObjectFile 階層)を消去
        System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_objectFile);
        this.m_objectFile = null;
    }
    if (this.m_objectTypeFile != null)
    {
        // CaoFile(ControllerFile 階層)から CaoFile(ObjectTypeFile 階層)を削除
        this.m_ctrlFile.Files.Remove(this.m_objectTypeFile.Index);
        // CaoFile(ObjectTypeFile 階層)を消去
        System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_objectTypeFile);
        this.m_objectTypeFile = null;
    }
    if (this.m_ctrlFile != null)
    {
        // CaoController から CaoFile(ControllerFile 階層)を削除
        this.m_caoCtrl.Files.Remove(this.m_ctrlFile.Index);
        // CaoFile(ObjectFile 階層)を消去
        System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_ctrlFile);
        this.m_ctrlFile = null;
    }
}
```

5.1.1.1. 参入

BACnet に参入するためには，以下の手順を取ります．

(1) オブジェクトを保持するための変数を用意します．コントローラ参入に必要なオブジェクトは，CaoEngineオブジェクトとCaoWorkspaceオブジェクトとCaoControllerオブジェクトです．

CaoWorkpaceオブジェクトは，CaoControllerオブジェクトをCaoWorkspacesから取得する場合には変数を用意する必要はありません．以下にC#でのコード例を示します．

```
private CaoEngine m_caoEng;                    // Engineオブジェクト
private CaoWorkspace m_caoWs;                  // Workspaceオブジェクト
private CaoController m_caoCtrl;               // Controllerオブジェクト
```

(2) CaoEngineオブジェクトを生成します．CaoEngineオブジェクトはnewキーワードを使って生成します．

```
// CaoEngine オブジェクトの生成
this.m_caoEng = new CaoEngine();
```

(3) CaoWorkspaceオブジェクトを取得もしくは生成します．CaoEngineオブジェクトを生成すると，デフォルトでCaoWorkspacesオブジェクトとCaoWorkspaceオブジェクトを1つずつ生成しています．以下にCaoWorkspaceオブジェクトを新しく生成するコード例とデフォルトのCaoWorkspaceを示します．

```
// CaoWorkspace オブジェクトの生成
this.m_caoWs = this.m_caoEng.AddWorkspace("NewWrks", "");
```

(4) CaoControllerオブジェクトを生成します．CaoControllerオブジェクトを生成するには，使用するプロバイダ名と使用するためのパラメータを設定します．BACnetプロバイダでは，自身のIPアドレス，自身のポート番号，ブロードキャストアドレス，自身のインスタンス番号，ネットワーク番号をオプションで指定します．サンプルでは以下の内容を指定してCaoControllerを作成しています．

- 自身のIPアドレス: 192.168.100.5,
- 自身のポート番号: 47808
- ブロードキャストアドレス: 192.168.100.255
- 自身のインスタンス番号: 5
- ネットワーク番号: 0

```
// CaoController オブジェクトの生成
this.m_caoCtrl = this.m_caoWs.AddController("BACnet",
                                            "CaoProv.unitec.BACnet",
                                            "",
                                            "MyIP=192.168.100.5,MyPort=47808,
                                             Broadcast=192.168.100.255,InstanceNo=5,
                                             NetworkNo=0");
```

5.1.1.2. CaoFile オブジェクト(3 階層)の追加

各 BACnet コントローラの BACnet オブジェクトを指定するためには，以下の手順を取ります．

(1) オブジェクトを保持するための変数を用意します. BACnet オブジェクトを指定するために必要なオブジェクトは, CaoFile オブジェクト(ControllerFile 階層)と CaoFile オブジェクト(ObjectTypeFile 階層)と CaoFile オブジェクト(ObjectFile 階層)です. また変数にアクセスするための CaoVariable オブジェクトも必要になります. 以下に C#でのコード例を示します.

```
private CaoFile m_ctrlFile;                 // Fileオブジェクト(ControllerFile階層)
private CaoFile m_objectTypeFile;           // Fileオブジェクト(ObjectTypeFile階層)
private CaoFile m_objectFile;               // Fileオブジェクト(ObjectFile階層)
private CaoVariable m_priorityValue;        // Variableオブジェクト
```

(2) CaoFileオブジェクト(ControllerFile階層)を生成します. CaoFileオブジェクト(ControllerFile階層)を生成するには, 任意のファイル名と接続先であるBACnetコントローラのインスタンス番号を設定します. 以下にコード例を示します.

```
// CaoFileオブジェクト(ControllerFile階層)の生成
this.m_ctrlFile = this.m_caoCtrl.AddFile("DEv-100", "InstanceNo=100");
```

(3) CaoFileオブジェクト(ObjectTypeFile階層)を生成します. CaoFileオブジェクト(ObjectTypeFile階層)を生成するには, アクセスしたいオブジェクトの種別をファイル名で設定します. 以下にコード例を示します.

```
// CaoFileオブジェクト(ObjectTypeFile階層)の生成
this.m_objectTypeFile = this.m_ctrlFile.AddFile("AnalogOutput", "");
```

(4) CaoFileオブジェクト(ObjectFile階層)を生成します. CaoFileオブジェクト(ObjectFile階層)を生成するには, ファイル名は任意で指定し, アクセスしたいオブジェクトのインスタンス番号をパラメータに設定します. 以下にコード例を示します.

```
// CaoFileオブジェクト(ObjectTypeFile階層)の生成
this.m_objectFile = this.m_objectTypeFile.AddFile("AO0", "ObjectInstanceNo=0");
```

(5) BACnetコントローラのBACnetオブジェクトの優先順位に格納されている値を取得/設定するためにはCaoVariableオブジェクトをCaoFileオブジェクト(ObjectFile階層)に追加する必要があります. サンプルプログラムでは優先順位に格納されている値の取得/設定を行うための変数PRESENTVALUE<??>を追加しています. <??>には任意の文字列を入力してください.

```
// CaoVariableオブジェクトの生成
this.m_priorityValue = this.m_objectFile.AddVariable("PRIORITYVALUE_8", null);
```

5.1.1.3. 優先順位に格納されている値の取得/設定

優先順位に格納されている値を取得/設定するには CaoVariable オブジェクトの Value プロパティを参照/設定します. AnalogOutput オブジェクトの優先順位に格納されている値を取得/設定する場合は, 32 ビット浮動小数点数型を用意する必要があります. 以下にコード例を示します.

```
// 値
```

```
float value;

// 値を取得
value = this.m_priorityValue.Value;

// 取得データに1を加算
value += 1;

// 値を設定
this.m_priorityValue.Value = value;

// 再取得
value = this.m_priorityValue.Value;
```

5.1.1.4. CaoFile オブジェクト(3 階層)の削除

追加した 3 階層の CaoFile オブジェクトの削除を行います. 以下にコード例を示します.

```
// CaoFile(ObjectTypeFile 階層)から CaoFile(ObjectFile 階層)を削除
this.m_objectTypeFile.Files.Remove(this.m_objectFile.Index);
// CaoFile(ObjectFile 階層)を消去
System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_objectFile);
this.m_objectFile = null;

// CaoFile(ControllerFile 階層)から CaoFile(ObjectTypeFile 階層)を削除
this.m_ctrlFile.Files.Remove(this.m_objectTypeFile.Index);
// CaoFile(ObjectTypeFile 階層)を消去
System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_objectTypeFile);
this.m_objectTypeFile = null;

// CaoController から CaoFile(ControllerFile 階層)を削除
this.m_caoCtrl.Files.Remove(this.m_ctrlFile.Index);
// CaoFile(ObjectFile 階層)を消去
System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_ctrlFile);
this.m_ctrlFile = null;
```

5.1.1.5. 離脱

コントローラと切断する場合には, 生成したオブジェクトを消去すると共に, オブジェクトを管理するコレクションクラスから消去するオブジェクトを削除します. 以下にコード例を示します.

```
// CaoWorkspaceからCaoControllerを削除
this.m_caoWs.Controllers.Remove(this.m_caoCtrl.Name);
// CaoControllerの消去
System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoCtrl);
this.m_caoCtrl = null;

// CaoEngineからCaoWorkspaceを削除
this.m_caoEng.Workspaces.Remove(this.m_caoWs.Name);
// CaoWorkspaceを消去
```

```
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoWs);
    this.m_caoWs = null;

    // CaoEngineを削除
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoEng);
    this.m_caoEng = null;
```

5.2. BACnet コントローラの状態変化を自動受信するサンプルプログラミング

ここでは例として AnalogInput オブジェクトの現在値を取得するサンプルプログラムを示します. 表 5-2 にサンプルプログラムの要件を, 図 5-2 にサンプルプログラムの流れをそれぞれ記述しています.

表 5-2 サンプルプログラムの要件

要件	説明
自身の設定	自身の IP アドレスは 192.168.100.5
	ポート番号は 47808
	ブロードキャストアドレスは 192.168.100.255
	自身のインスタンス番号は 5
	ネットワーク番号は 0
処理内容	BACnet コントローラからの状態変化を OnMessage で受信する

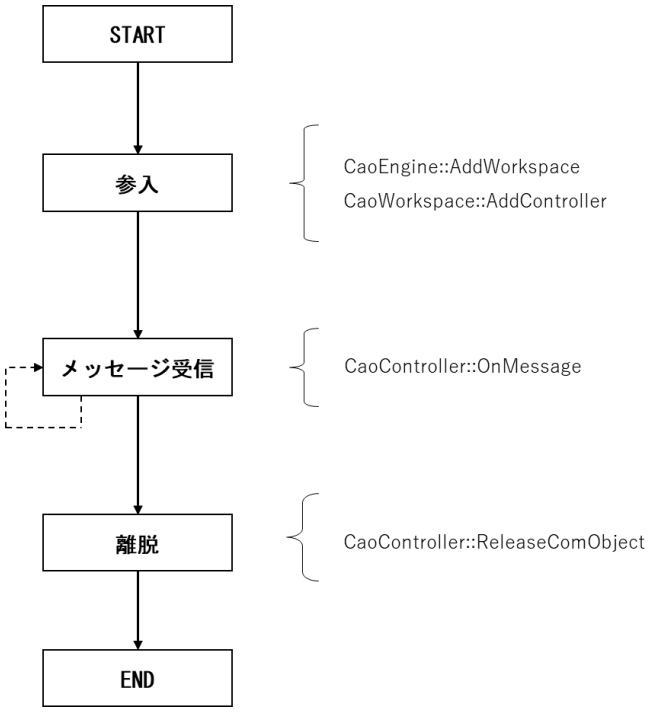


図 5-2 BACnet コントローラの状態変化取得の流れ

以降の節から具体的なコードを示します.

5.2.1. サンプルプログラム

以下にサンプルプログラムの全体像を示します.

Sample ChangeSystemStatusFeature.cs

```
// オブジェクト
Private CaoEngine m_caoEng;
Private CaoWorkspace m_caoWs;
Private CaoController m_caoCtrl;

public ChangeSystemStatusFeature()
{
    // 参入
    this.Entry();
    // OnMessage イベントハンドラの登録
    this.m_caoCtrl.OnMessage +=
                        new _ICaoControllerEvents_OnMessageEventHandler(OnMessage);
    // 離脱
    this.BreakAway();
```

```
}

// メッセージ受信メソッド
private void OnMessage(CaoMessage pICaoMsg)
{
    if (pICaoMsg.Number == 1)      // 機器の状態変化の通知のメッセージを受信した場合
    {
        object[] msgValues = new object[3];
        msgValues = pICaoMsg.Value;

        // 通知元のインスタンス番号
        Ulong deviceInstanceNo = Convert.ToUInt64(msgValues[0]);

        // 前回の状態値
        int oldStatus = Convert.ToInt32(msgValues[1]);

        // 現在の状態値
        int nowStatus = Convert.ToInt32(msgValues[2]);
    }
}

// BACnet に参入する
private void Entry()
{
    this.m_caoEng = new CaoEngine();
    this.m_caoWs = this.m_caoEng.AddWorkspace("NewWrks", "");
    this.m_caoCtrl = this.m_caoWs.AddController("BACnet",
                                                "CaoProv.unitec.BACnet",
                                                "",
                                                "MyIP=192.168.100.5,MyPort=47808,
                                                Broadcast=192.168.100.255,
                                                InstanceNo=5,NetworkNo=0");
}


// BACnet から離脱する
Private void BreakAway()
```

```
{
    If (this.m_caoCtrl != null)
    {
        // CaoWorkspace から CaoController を削除
        this.m_caoWs.Controllers.Remove(this.m_caoCtrl.Index);
        // CaoController の消去
        System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoCtrl);
        this.m_caoCtrl = null;
    }
    // CaoEngine から CaoWorkspace を削除
    this.m_caoEng.Workspaces.Remove(this.m_caoWs.Index);
    // CaoWorkspace を消去
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoWs);
    this.m_caoWs = null;
    // CaoEngine を削除
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoEng);
    this.m_caoEng = null;
}
```

5.2.1.1. 参入

BACnet に参入するためには，以下の手順を取ります．

(1) オブジェクトを保持するための変数を用意します．コントローラ参入に必要なオブジェクトは，CaoEngineオブジェクトとCaoWorkspaceオブジェクトとCaoControllerオブジェクトです．CaoWorkpaceオブジェクトは，CaoControllerオブジェクトをCaoWorkspacesから取得する場合には変数を用意する必要はありません．以下にC#でのコード例を示します．

```
private CaoEngine m_caoEng;                 // Engineオブジェクト
private CaoWorkspace m_caoWs;               // Workspaceオブジェクト
private CaoController m_caoCtrl;            // Controllerオブジェクト
```

(2) CaoEngineオブジェクトを生成します．CaoEngineオブジェクトはNewキーワードを使って生成します．

```
// CaoEngine オブジェクトの生成
this.m_caoEng = new CaoEngine();
```

(3) CaoWorkspaceオブジェクトを取得もしくは生成します．CaoEngineオブジェクトを生成すると，デフォルトでCaoWorkspacesオブジェクトとCaoWorkspaceオブジェクトを1つずつ生成してい

ます．以下にCaoWorkspaceオブジェクトを新しく生成するコード例とデフォルトのCaoWorkspaceを示します．

```
// CaoWorkspace オブジェクトの生成
this.m_caoWs = this.m_caoEng.AddWorkspace("NewWrks", "");
```

(4) CaoControllerオブジェクトを生成します．CaoControllerオブジェクトを生成するには，使用するプロバイダ名と使用するためのパラメータを設定します．BACnetプロバイダでは，自身のIPアドレス，自身のポート番号，ブロードキャストアドレス，自身のインスタンス番号，ネットワーク番号をオプションで指定します．サンプルでは以下の内容を指定してCaoControllerを作成しています．

- 自身のIPアドレス: 192.168.100.5,
- 自身のポート番号: 47808
- ブロードキャストアドレス: 192.168.100.255
- 自身のインスタンス番号: 5
- ネットワーク番号: 0

```
' CaoController オブジェクトの生成
this.m_caoCtrl = this.m_caoWs.AddController("BACnet",
                                            "CaoProv.unitec.BACnet",
                                            "",
                                            "MyIP=192.168.100.5,MyPort=47808,
                                             Broadcast=192.168.100.255,InstanceNo=5,
                                             NetworkNo=0");
```

5.2.1.2. メッセージ受信

他のBACnetコントローラから送られてくる通知の情報はCaoController::OnMessageイベントをハンドルすることで取得可能となります．サンプルプログラムでは，OnMessage イベントハンドラを用意して CaoController::OnMessage イベントをハンドルしています．各メッセージの仕様については，3.4 を参照してください．

```
// 機器の状態変化の通知のメッセージを受信した場合
if (pICaoMsg.Number == 1)
{
    object[] msgValues = new object[3];
    msgValues = pICaoMsg.Value;


    // 通知元のインスタンス番号
    ulong deviceInstanceNo = Convert.ToUInt64(msgValues[0]);

    // 前回の状態値
    int oldStatus = Convert.ToInt32(msgValues[1]);
```

```
        // 現在の状態値
        int nowStatus = Convert.ToInt32(msgValues[2]);
    }
```

5.2.1.3. 切断

コントローラと切断する場合には，生成したオブジェクトを消去すると共に，オブジェクトを管理するコレクションクラスから消去するオブジェクトを削除します．以下にコード例を示します．

```
    // CaoWorkspaceからCaoControllerを削除
    this.m_caoWs.Controllers.Remove(this.m_caoCtrl.Name);
    // CaoControllerの消去
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoCtrl);
    this.m_caoCtrl = null;

    // CaoEngineからCaoWorkspaceを削除
    this.m_caoEng.Workspaces.Remove(this.m_caoWs.Name);
    // CaoWorkspaceを消去
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoWs);
    this.m_caoWs = null;

    // CaoEngineを削除
    System.Runtime.InteropServices.Marshal.ReleaseComObject(this.m_caoEng);
    this.m_caoEng = null;
```

6. BACnet プロバイダエラーコード

本プロバイダには，0x8011****でマスクした以下の独自エラーコードが存在します．（表 6-1 独自エラーコード表参照）

ORiN2 の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 6-1 独自エラーコード表

エラー番号	説明
0x80110002	排他制御に失敗した．
0x80110003	対象オブジェクトが存在ない．

また，本プロバイダは，API のエラーコードを「0x8010****」でマスクして返します．（表 6-2 API からのエラーコード表参照）

表 6-2 API からのエラーコード

エラー番号	説明
0x8010FFFF	すでにOpen された状態でOpen 関数をコールした．
0x8010FFFE	すでにClose された状態でClose 関数をコールした．
0x8010FFFD	バックアップファイルの読み書きに失敗した．
0x8010FFFC	追加しようとした情報がすでに登録済みであった．
0x8010FFFB	指定したパラメータが不正であった．
0x8010FFFA	指定したパラメータに関する情報が見つからなかった．
0x8010FFF9	ACnet.ini に設定されたIP アドレスがPC に登録されていないまたは既に使用されているなどの理由により，BACnet通信の開始に失敗した．
0x8010FFF8	リソース不足により実行できなかった．
0x8010FFF7	BACnet 通信が開始されていない．
0x8010FC19	プロテクト解除のための解除キーが見つからなかった． ※プロテクトがかかったDLL を使用している場合のみ．
0x8010D8F1	不明のエラーが発生した．
0x80100001	相手デバイスからError応答があった．
0x80100002	相手デバイスからReject応答があった．
0x80100003	相手デバイスからAbort応答があった．
0x80100004	相手デバイスへの要求がリトライ回数を超えた．
0x80100005	相手デバイスがBACnetから離脱している．
0x8010000D	相手デバイスが存在しない．

付録A. API 対応表

CaoWorkspace::AddController

API 関数名
BD_OpenDirect
BD_SetDeviceStatusChangeCallback
BD_SetCOVCallback
BD_SetEventCallback
BD_SetSystemStatus
BD_SendWhoisWithRange
BD_SendTimeSync
BD_SendIam
BD_Close

CaoController

API 関数名
BD_SendTimeSyncUnicast
BD_SendWhoisWithRange

CaoVariable

変数名	Get_Value	Set_Value
@PRESENTVALUE	BD_ReadPropertyMultiple	---
@SYSTEMSTATUS	BD_ReadPropertyMultiple	---
PRIORITYVALUE<??>	BD_ReadPropertyMultiple	BD_WritePropertyMultiple

付録B. イニシャル手順

本プロバイダでは，BACnet に参入/離脱する際に BACnet 公開仕様書のイニシャル手順に記載されている手順をもとに参入/離脱します．参入時は，参入シーケンスをもとに参入し，離脱時は，離脱シーケンスをもとに離脱します．

付録B.1. 参入シーケンス

以下に BACnet に参入する際に本プロバイダと他の BACnet コントローラとの手順を記載します

手順

本プロバイダ		他の BACnet コントローラ
1) 参入開始(DOWNLOAD_REQURED) Service = UnconfimedCOVNotification 'Subscriber Process Identifier' = 0 'Initiating Device Identifier' = (Device, Instance N) 'Monitored Object Identifier' = (Device, Instance N) 'Time Remaining' = 0 'List of Values' = ((System_Status, DOWNLOAD_REQUIRED), (Time_Of_Device_Restart, (YYYYMMDD(W), HH:MM:SS.XX)),	→	

本プロバイダ		他の BACnet コントローラ
(Last_Restart_Reason, XXXXX))		
2) Who-Is 送信(ブロードキャスト) Service=Who-Is Device Instance Low Limit = XX Device Instance Hi Limit = ZZ	→ ←	1) I-Am 送信(ブロードキャスト) Service = I-Am 'I-Am Device Identifier' = (Device, Instance N) 'MAX APDU Length Accepted' = 1024 'Segmentation Supported' = SEGMENTED__BOTH 'Vendor Identifier' = 85
3) システム時刻の確定 Service = UnconfimedCOVNotification 'Subscriber Process Identifier' = 0 'Initiating Device Identifier' = (Device, Instance N) 'Monitored Object Identifier' = (Device, Instance N) 'Time Remaining' = 0 'List of Values' = ((System_Status, DOWNLOAD_IN_PROGLESS), (Time_Of_Device_Restart, (YYYYMMDD(W), HH:MM:SS.XX)), (Last_Restart_Reason, XXXXX))	→	
4) 参入終了	→	

本プロバイダ		他の BACnet コントローラ
Service = UnconfimedCOVNotification 'Subscriber Process Identifier' = 0 'Initiating Device Identifier' = (Device, Instance N) 'Monitored Object Identifier' = (Device, Instance N) 'Time Remaining' = 0 'List of Values' = ((System_Status, OPERATIONAL), (Time_Of_Device_Restart, (YYYYMMDD(W), HH:MM:SS.XX)), (Last_Restart_Reason, XXXXX))		

付録B.2. 離脱シーケンス

以下に本プロバイダが BACnet から離脱する際の手順を記載します.

手順

本プロバイダ		他の BACnet コントローラ
1) 離脱 Service = UnconfimedCOVNotification 'Subscriber Process Identifier' = 0 'Initiating Device Identifier' = (Device, Instance N)	→	

本プロバイダ		他の BACnet コントローラ
‘Monitored Object Identifier’ = (Device, Instance N) ‘Time Remaining’ = 0 (System Status = Non_OPERATIONAL)		

付録B.3. BACnet コントローラが参入した場合

以下に本プロバイダが BACnet に参入している際に，他の BACnet コントローラが参入してきた際の手順を記載します.

手順

本プロバイダ		他の BACnet コントローラ
	←	1) 参入開始(DOWNLOAD_REQURED) Service = UnconfimedCOVNotification ‘Subscriber Process Identifier’ = 0 ‘Initiating Device Identifier’ = (Device, Instance N) ‘Monitored Object Identifier’ = (Device, Instance N) ‘Time Remaining’ = 0 ‘List of Values’ = ((System_Status, DOWNLOAD_REQUIRED), (Time_Of_Device_Restart, (YYYYMMDD(W), HH:MM:SS.XX)), (Last_Restart_Reason, XXXXX))
3) I-Am 送信(ブロードキャスト) Service = I-Am ‘I-Am Device Identifier’ = (Device,	← →	2) Who-Is 送信(ブロードキャスト) Service=Who-Is Device Instance Low Limit = XX

本プロバイダ		他の BACnet コントローラ
Instance N) 'MAX APDU Length Accepted' = 1024 'Segmentation Supported' = SEGMENTED_BOTH 'Vendor Identifier' = 85		Device Instance Hi Limit = ZZ
	←	4) 参入終了(OPERATIONAL) Service = UnconfimedCOVNotification 'Subscriber Process Identifier' = 0 'Initiating Device Identifier' = (Device, Instance N) 'Monitored Object Identifier' = (Device, Instance N) 'Time Remaining' = 0 'List of Values' = ((System_Status, OPERATIONAL), (Time_Of_Device_Restart,(YYYYMMDD(W),HH:MM:SS.XX)), (Last_Restart_Reason, XXXXX))