b-CAP プロバイダ

b-CAP 通信

Version 1.4.1

ユーザーズ ガイド

December 4, 2024

【備考】

【改版履歴】

バージョン	日付	内容
1.0.0.0	2006-08-02	初版.
1.0.1.0	2007-06-23	Interval オプションの追加.
1.0.2.0	2007-11-21	UDP オプションの追加.
1.0.3.0	2008-01-19	bCapListener の説明補足.
1.0.4.0	2009-04-01	MyIP オプションの追加.
1.1.0.0	2009-07-21	bCapListener に優先度設定オプション(/R)の追加, UDP リトライ機能対応.
1.1.1.0	2010-02-10	bCapService, bCapConfig の追加, エラーコード追加
1.1.2.0	2010-08-20	b-CAP/COM 通信対応
1.1.3.0	2011-05-15	bCapConfig 修正
1.1.4.0	2012-06-05	各通信モードに最大パケットサイズを設定
1.1.4	2012-07-17	ドキュメントのバージョンルールを変更
1.2.0	2012-08-06	WDT オプション, AsyncCancel オプション追加.
1.2.1	2012-09-06	KeepAlive 時間指定オプション, InvokeTimeout オプション追加.
1.2.2	2013-01-29	b-CAP ZIP 圧縮機能対応
1.3.0	2014-10-08	SSL によるセキュア通信対応
1.3.1	2016-04-13	b-CAP/UDP の最大接続数の制御機能追加
1.3.2	2018-11-29	連続接続に対応.
1.3.3	2018-12-17	SSL 通信時のハンドシェイクのリトライ追加.
1.3.4	2020-11-01	メモリ関連の処理修正 全体的に処理を修正
1.3.5	2021-08-19	起動, 終了処理の修正 再接続時処理の修正 メモリ関連処理の修正 排他処理の修正 メッセージ処理の非同期対応
1.4.0	2021-12-20	OpenSSL バージョンアップ.
1.4.1	2022-01-06	SSL オプションの不具合修正.
	2022-01-31	証明書作成方法の変更.
	2024-12-04	OpenSSL バージョンアップ.

【動作確認機器】

機種	バージョン	注意事項

目次

1. はじめに

本書は b-CAP を使用して，リモートマシンの CAO と通信を行うプロバイダである b-CAP プロバイダのユーザーズガイドです．

b-CAP は CAP の概念を踏襲しつつ，通信速度の向上を狙ったプロトコルです．このため b-CAP は，TCP ストリーム通信を使用して，CAP と同様の機能を提供します．

本書は，この b-CAP プロバイダの機能と実装されているメソッドについて説明します．

この製品は OpenSSL ツールキットを利用するために OpenSSL プロジェクト[1]によって開発されたソフトウェアを含んでいます．

[1] https://www.openssl.org/

2. プロバイダの概要

2.1. 概要

b-CAP プロバイダは, 通信仕様として b-CAP(Bynary CAP)を採用しています. b-CAP は CAP の概念を踏襲して通信速度の向上を行ったプロトコルです.

以下に他のリモート通信形態の比較図を示します.

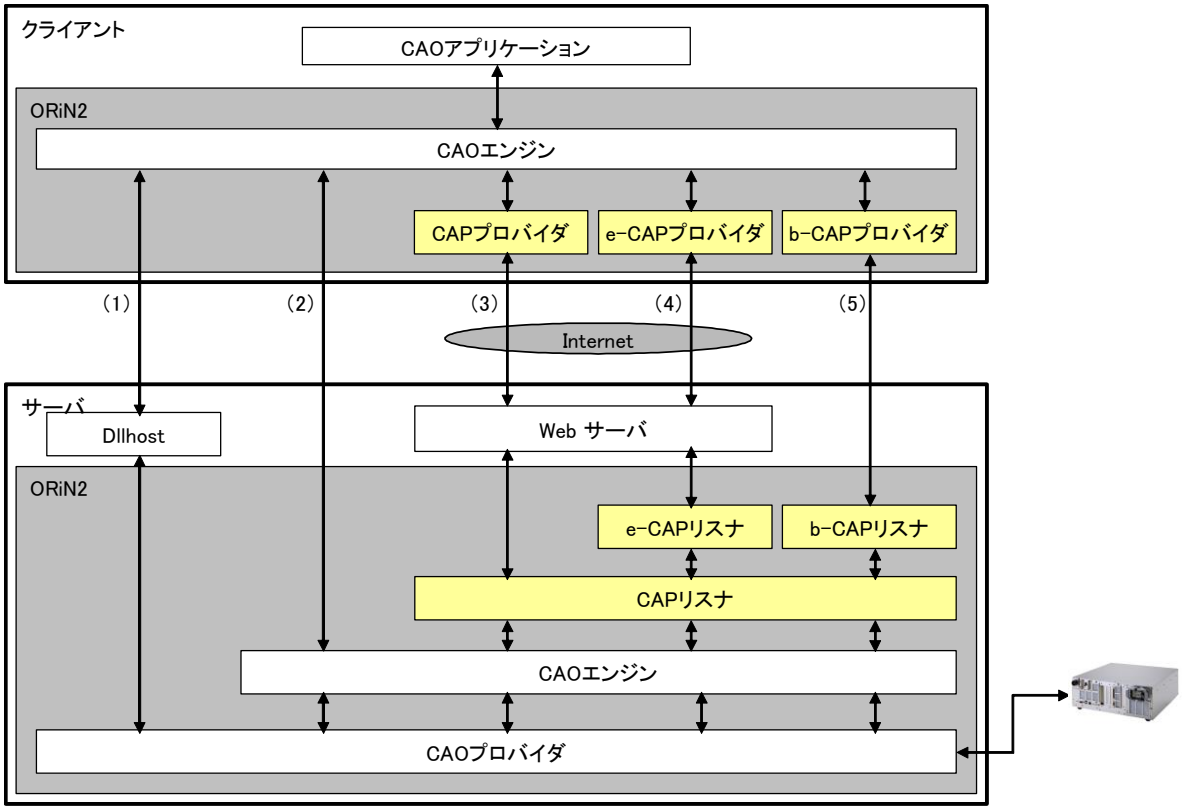

図 2-1 通信形態の比較

b-CAP は, TCP でメソッド呼び出し及び実行結果を表現した b-CAP メッセージを通信します. b-CAP の通信プロトコルの詳細については, b-CAP 通信仕様書を参照してください.

b-CAP プロバイダから送信したメッセージをサーバ側で処理するためのプログラムとして b-CAP リスナがあります. b-CAP リスナは CAP リスナを通してメッセージによって指定された CAO のメソッドを実行します.

この b-CAP プロバイダと b-CAP リスナを使用することにより b-CAP によるリモート CAO エンジンを操作することができます.

以下に b-CAP の接続例を示します. この中で[3]が b-CAP プロバイダと b-CAP リスナの接続例になります.

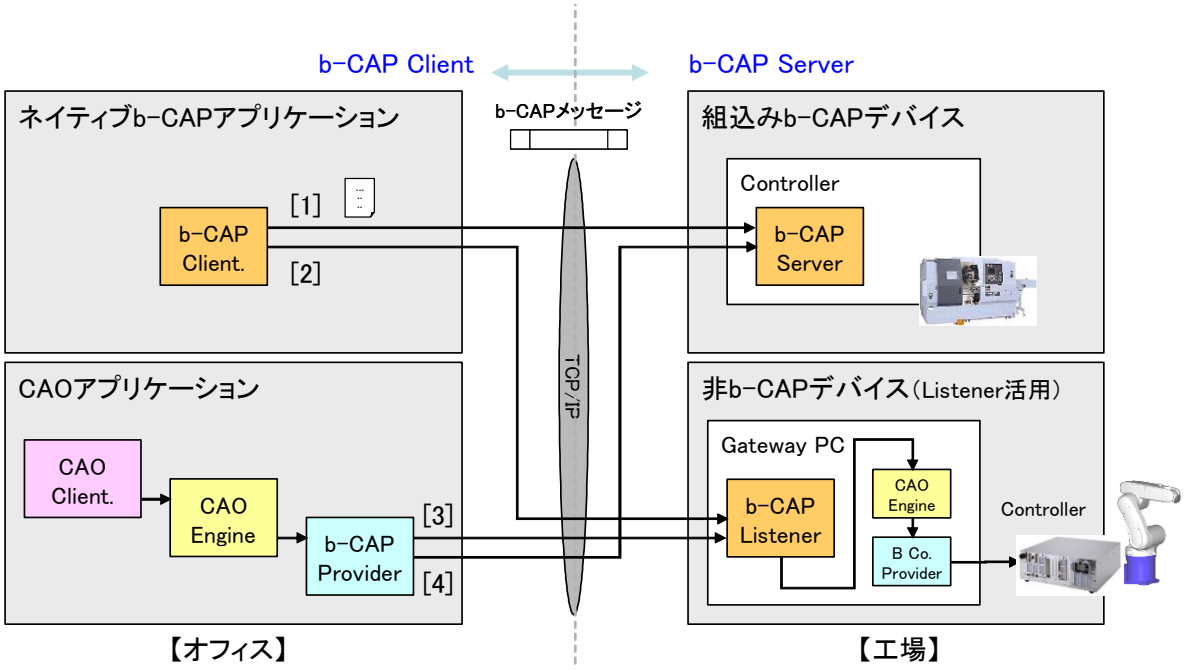

図 2-2 b-CAP による接続形態

b-CAP プロバイダでは，通信方法として TCP，UDP，COM 通信の3種類を用意しています．使用する通信方法は接続時に指定します．

b-CAP プロバイダでは，通信方法によってパケットの最大サイズが異なります．

表 2-1 最大パケットサイズ

通信方法	最大パケットサイズ
TCP	4G Byte
UDP	504 Byte
COM	504 Byte

2.1.1. b-CAP プロバイダのセットアップ

b-CAP プロバイダを使用するには，CAO から参照できるようにレジストリに登録する必要があります．

表 2-2 b-CAP プロバイダ

ファイル名	CaoProvBCAP.dll
ProgID	CaoProv. b-CAP
レジストリ登録[2]	regsvr32 CaoProvBCAP.dll
レジストリ登録の抹消	regsvr32 /u CaoProvBCAP.dll

2.1.2. メッセージ

リモートエンジンで発生したメッセージの取得は AddController メソッドの“Message”オプションで切り替えることができます．

メッセージを OFF にしたときは，リモートエンジンで発生したメッセージは取得できません．

メッセージを ON にしたときは，メッセージオプションでエンジン制御メッセージのビットフラグが OFF のメッセージのみ取得できます．

また，他のメソッドを実行中の時には，メッセージを取得することができません．メソッド実行中に発生したメッセージについては，実行後にまとめて取得します．

2.1.3. SSL によるセキュア通信

バージョン 1.3.0 以上の b-CAP プロバイダでは SSL によるセキュアな通信を構築することができます．[3]

既存の TCP 通信を暗号化することにより通信のセキュリティが確保できます．

SSL によるセキュアな通信を行うためには b-CAP プロバイダと b-CAP リスナ等のサーバ側のプログラムの両方のセットアップを適切に行う必要があります．

詳細なセットアップ方法については「6.1.SSL によるセキュア通信」を参照してください．

[2] 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．
[3] TCP 通信のみ対応，UDP 通信では SSL によるセキュアな通信は行えません．

2.2. メソッド・プロパティ

2.2.1. CaoWorkspace::AddController メソッド

このメソッドを実行するとサーバで CAO と CAO プロバイダを起動し，接続します．このとき接続先のマシンで bCapLintener または b-CAP サーバが起動していないときは，このメソッドは失敗します．

リモート起動するプロバイダに必要なパラメータは，このメソッドのオプション文字列に指定します．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 2-3 CaoWorkspace::AddController のオプション文字列

オプション	意味
Server[=<IP アドレス> [:<ポート番号>]]	b-CAP サーバの IP アドレスとポート番号を指定します． （デフォルト値：“127.0.0.1:5007”）
COM=<COM Port> [:<Baud Rate> [:<Parity> :<DataBits> :<StopBits> [:Flow]]]	COM 通信設定．このオプションを設定した場合，Server，MyIP，UDP オプションは無視されます． <COM Port> : COM ポート番号． ‘1’-COM1，‘2’-COM2，… <BaudRate> : 通信速度． 4800，9600，19200，38400，57600，115200． <Parity> : パリティ． ‘N’-NONE，‘E’-EVEN，‘O’-ODD． <DataBits> : データビット数． ‘7’-7bit，‘8’-8bit． <StopBits> : ストップビット数． ‘1’-1bit，‘2’-2bit． <Flow> : フロー制御．(デフォルト：‘0’-フロー制御なし) ‘1’-Xon/Xoff，‘2’-ハードウェア制御． OR をとって指定できます．
MyIP[=<ローカル IP アドレス>]	複数の NIC を使う場合にこのオプションで IP アドレスを指定して NIC を選択することができます．省略した場合は，自動的に選択されます． ローカルマシンに割り当てられていない IP アドレスを指定したときはエラーを返します． RS232C 接続の場合，このオプションは無視されます．
Provider=<プロバイダ名>	リモート起動するプロバイダ名． （デフォルト値：空文字列）
Machine[=<マシン名>]	WEB サーバと異なるマシンでリモートプロバイダを起動するときに指定します． （デフォルト値：空文字列）

Option[=<オプション文字列>]	リモートプロバイダに必要なオプション文字列を指定します.（デフォルト値:空文字列）
Message[=<True/False>]	メッセージ取得の有無. True:メッセージ取得あり（デフォルト） False:メッセージ取得なし
Interval=<ポーリング間隔>	メッセージ取得間隔（ms）を指定します.（デフォルト値:1000 ms）
UDP[=<True/False>]	UDP による通信設定 True:UDP False:TCP（デフォルト） UDP 通信の場合パケットの最大サイズは 488 バイトになります.
ConnTimeout=<タイムアウト時間>	接続時のタイムアウト時間を設定します.（デフォルト:5000ms） このオプション省略時に Timeout オプションが指定されたときは，Timeout オプションの値を設定します. 設定値が 1000ms 以下の場合は，1000ms に設定されます.
Timeout=<タイムアウト時間>	送受信時のタイムアウト時間.（デフォルト:500 ms）
TORetry=<リトライ回数>	UDP 送受信時のリトライ回数. 1～7（デフォルト:5） 1 以下の場合，1 として扱われます. 7 以上の場合，7 として扱われます. UDP のタイムアウト応答時間は，以下の式で算出されます. タイムアウト応答時間 = <Timeout>×<TORetry>
WDT=<ウォッチドッグタイマ時間>	b-CAP サーバのウォッチドッグタイマ間隔(ms)を指定します.（デフォルト:なし） b-CAP サーバでコマンドの実行中に指定時間が経過したとき，b-CAP プロバイダに対して特殊パケット（実行中通知パケット）を送信します. b-CAP プロバイダは，サーバから実行中通知パケットを受け取るごとにタイムアウトのカウントをリセットします. この設定は,80ms より小さい値を指定することができません. この設定は，b-CAP サーバが bCapService 又は bCapListener の場合のみ指定することができます.
InvokeTimeout=<タイムアウト時間>	コマンド実行のタイムアウト時間を ms 単位で指定します.（デフォルト:180000 ms） コマンド処理に要する時間がこの時間を超える場合は，サーバはコマンドの実行完了待機を中止します. この設定は,80ms より小さい値を指定することができません.

	この設定は, b-CAP サーバが bCapService 又は bCapListener の場合のみ指定することができます.
AsyncCancel[=<True/False>]	非同期キャンセルモードの指定 True: 非同期キャンセルモード[4] False: 通常モード(同期キャンセルモード) 非同期キャンセルモードのときは, CaoController::Execute()の"ProviderCancel"及び"ProviderClear"コマンドを非同期で実行します.(デフォルト:False) TCP 接続以外の場合は, このオプションは無視されます. [備考] このオプションは, b-CAP プロバイダのレジストリに登録することもできます. 登録方法は CaoConfig を使用して b-CAP プロバイダの Parameter に指定します. AddController()でこのオプションを省略した場合は, レジストリで指定した値を使用します.
ProtcolVersion[=<バージョン>]	b-CAP プロトコルのバージョンを指定します.(デフォルト:0) b-CAP の ZIP 圧縮通信を使用する場合は, このオプションの値を1にしてください.
ZIPMode[=<圧縮レベル>]	b-CAP の ZIP 圧縮通信時の圧縮レベルを指定します.(デフォルト:-1) -1 デフォルトの圧縮設定(圧縮レベル:6) 0 ZIP 圧縮なし 1～9 ZIP 圧縮レベル 値が小さいほど圧縮時間が優先されます. 値が大きいほど圧縮率が優先されます. ZIP 圧縮通信を有効にするために ProtcolVersion オプションに1を指定してください.
ZipThreshold[=<圧縮サイズ閾値>]	b-CAP の ZIP 圧縮サイズ閾値を指定します. サイズはキロバイト単位で指定します.(デフォルト:1KB) b-CAP パケットのサイズが指定した値を上回る場合, ZIP 圧縮通信を行います.

[4] 非同期キャンセルモードによる接続は, b-CAP サーバ側のプログラムが bCapListener.exe もしくは bapService.exe の場合のみサポートしています.

Debug[=<True/False>]	デバッグモードの指定　True:デバッグモード False:通常モード デバッグモードのときは, 以下の変数を使用することができます. $LAST_SEND_PACKET$ $LAST_RECEIVE_PACKET$
SSL[=<True/False>]	バージョン 1.3.0 以上のみ SSL によるセキュア通信設定 True:SSL False:TCP(デフォルト)
Certificate=<証明書ファイル名>	SSL＝True 時のみ 証明書ファイル名を指定します. (デフォルト:空文字)
PrivateKey=<秘密鍵ファイル名>	SSL＝True 時のみ 秘密鍵ファイル名を指定します. (デフォルト:空文字)
Password=<パスワード>	SSL＝True 時のみ 秘密鍵のパスワードを指定します. (デフォルト:空文字)
CA=<証明機関(CA)証明書ファイル名>	SSL＝True 時のみ 証明機関(CA)の証明書ファイル名を指定します. (デフォルト:空文字)

以下に AddController メソッドを実行するときの例を示します.

```
AddController
(
    "RC1",                                          // コントローラ名 ＝ RC1
    "CaoProv.b-CAP",                                // 固定
    "",                                             // CAO エンジンプロセスで CAP プロバイダを実行
    "Server=10.8.109.116:5007,Provider=CaoProv.DataStore"
                                                    // IP アドレス : 10.8.109.116, ポート番号 : 5007
                                                    // で DataStore プロバイダを起動します.
);
```

また, 以下に SSL によるセキュア通信を使用するときの例を示します.

```
AddController
(
     “RC1”,                                          // コントローラ名 ＝ RC1
     “CaoProv.b-CAP”,                                // 固定
     “”,                                             // CAO エンジンプロセスで CAP プロバイダを実行
     “Server=10.8.109.116:5107,Provider=CaoProv.DataStore,” _
     & “SSL,Certificate=C:¥store¥client.pem,PrivateKey=C:¥store¥client.pem,” _
     & “Password=pass,CA=C:¥store¥rootcert.pem”
                                                     // IP アドレス：10.8.109.116, ポート番号：5107
                                                     // 証明書ファイル名 ＝ C:¥store¥client.pem
                                                     // 秘密鍵ファイル名 ＝ C:¥store¥client.pem
                                                     // パスワード ＝ pass
                                                     // 証明機関（CA）証明書ファイル名 ＝
                                                     //     C:¥store¥rootcert.pem
                                                     // で DataStore プロバイダを起動します.
);
```

【参考】

AddControllerメソッドを実行すると, b-CAPの(1)Service_Start,(2)Controller_Connectの2つの関数がこの順で呼び出されます. また, AddController により作成された CaoController オブジェクトが消滅する際には,(3) Controller_Disconnect,(4) Service_Stop の 2 つの関数がこの順で呼び出されます.

2.2.2. AddController 以外のメソッド・プロパティ

b-CAP 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います. 前述の AddController(2.2.1)以外のメソッド, プロパティはサーバの CAO で同名のメソッド, プロパティを実行します.

2.3. 変数一覧

b-CAP プロバイダ固有の変数はありません.

2.4. エラーコード

b-CAP 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. b-CAP リスナ

3.1. 概要

b-CAP リスナは, b-CAP メッセージを受け取り, 対応する CAO のメソッドを実行します. b-CAP による通信を行う時にはあらかじめサーバ側で bCapListener を起動しておく必要があります.

b-CAP リスナには以下の2種類があります.

表 3-1 b-CAP リスナプログラム

プログラム名	プログラム種別
bCapListener.exe	コンソールアプリケーション
bCapService.exe	Windows サービスプログラム

b-CAP リスナを実行する際には, b-CAP リスナの起動ユーザに対して CAO はアクセス権限を設定しておく必要があります.

3.2. bCapListener.exe

3.2.1. 実行パラメータの設定

bCapLitener.exe の設定は起動時のコマンドラインで行います.

bCapListener.exe では以下のコマンドが使用できます. これらのコマンドは任意に組み合わせることができます.

```
bCapListener.exe [/P:ポート番号] [/C:最大クライアント数] [/[-]K]
```

ここで, コマンドオプションには大文字・小文字の区別はありません. []は省略可能であることを意味しています.

以下に各コマンドの詳細について説明します.

/P:ポート番号

サーバで通信に使用する TCP または UDP のポート番号を指定します. (デフォルト:5007)

※SSL によるセキュア通信時のデフォルトは 5107 となります

/C:最大クライアント数

サーバに同時に接続できるクライアントの数を指定します.（デフォルト:10）

/R:プロセス優先度

プロセス優先度（0: IDLE, 1: NORMAL, 2: HIGH, 3: REALTIME)を指定します.（デフォルト:2）

/T:タイムアウト時間

通信タイムアウト時間を指定します．（デフォルト：500）

/U

UDP で通信を行います．（デフォルト：TCP）

/K

TCP のキープアライブ機能を使用します．（デフォルト：ON）

“/-K”のようにハイフンをつけて指定した場合はこの機能を無効に設定します．

/S

SSL で通信をおこないます．（デフォルト：TCP）

/CF:証明書ファイル名

証明書ファイル名を指定します．（デフォルト：空文字）

※SSL 通信時のみ

/PK:秘密鍵ファイル名

秘密鍵ファイル名を指定します．（デフォルト：空文字）

※SSL 通信時のみ

/PW:パスワード

秘密鍵のパスワードを指定します．（デフォルト：空文字）

※SSL 通信時のみ

/CA:証明機関（CA）証明書ファイル名

証明機関（CA）の証明書ファイル名を指定します．（デフォルト：空文字）

※SSL 通信時のみ

/CRL:証明書失効リストファイル名

証明書失効リストファイル名を指定します．（デフォルト：空文字）

※SSL 通信時のみ

3.3. bCapService.exe

3.3.1. セキュリティ設定

bCapService は，システムユーザで起動します．このため CAO にシステムユーザのアクセス権を設定する必要があります．設定手順は以下のようになります．

1. コマンドプロンプトで"dcomcnfg"を実行します．
2. “コンポーネントサービス”ウィンドウのツリーから以下の場所を選択します．
   コンソールルート→コンポーネントサービス→コンピュータ→マイコンピュータ→DCOM の構成
3. 一覧から CAO を選択し，メニューの“操作(A)”→“プロパティ(R)”を選択します．
4. CAO のプロパティ画面のセキュリティタブを選択し，アクセス許可で“カスタマイズ(M)”を選択し，“編集(D)”をクリックします．

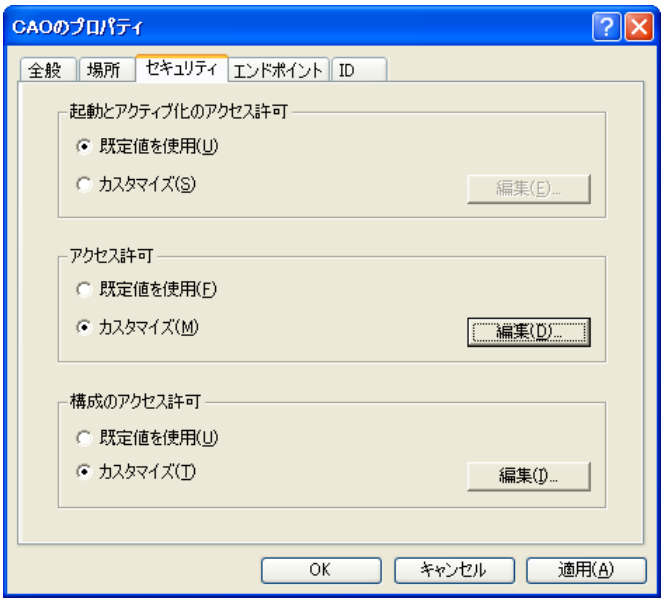

図 3-1 CAO の DCOM 設定画面(セキュリティタブ)

5. “追加(D)”ボタンをクリックし，ローカルマシンの“SYSTEM”ユーザを追加します．[5]
6. SYSTEM ユーザのローカルアクセスを許可にして OK ボタンをクリックします．

[5] 既に SYSTEM ユーザがある場合はこの手順は不要です．

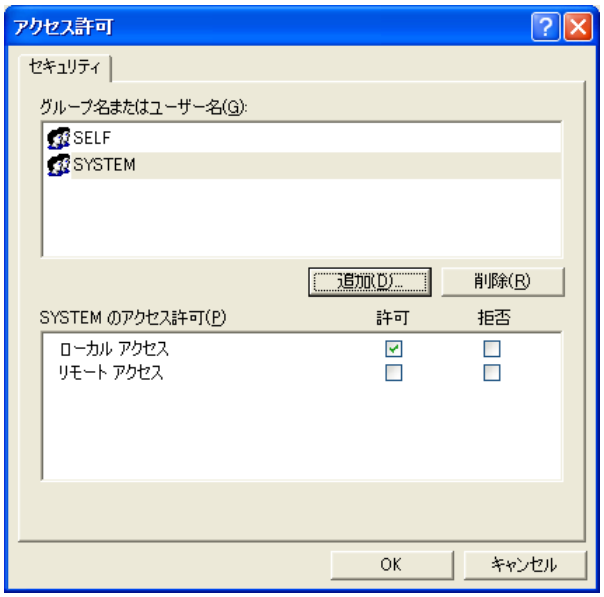

図 3-2 CAO のアクセス許可の設定画面

7. CAO のプロパティ画面の ID タブを選択し，“対話ユーザー(I)”を選択します．[6]

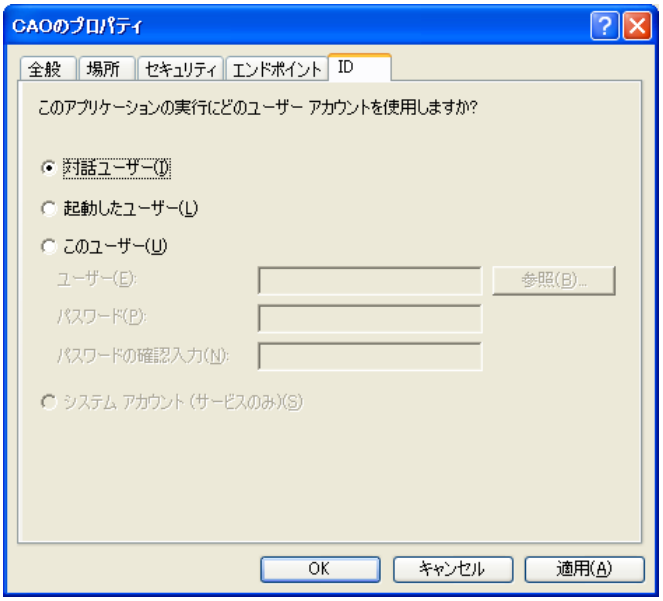

図 3-3 CAO の DCOM 設定画面(ID タブ)

[6] 仮に，「起動したユーザ」を選択した場合，別のクライアントプログラムが CAO を起動した時に 2 つの CAO プロセス(ログインユーザとシステムユーザ)が起動してしまうため正しく通信することができないばかりでなく，負荷が倍になってしまうので注意して下さい．

8. CAO のプロパティ画面の OK ボタンをクリックします.

3.3.2. 実行パラメータの設定

bCapService.exe の設定には, bCapConfig.exe を使って行います. 設定情報はレジストリに登録されます.
bCapConfig.exe の詳細については, 4 を参照してください.

4. bCapConfig

4.1. 概要

b-CAP Configuration Manager(実行ファイル名 bCapConfig.exe，以後 bCapConfig)は，マシン内の bCapService の通信設定を行うツールです．ここで設定した情報はレジストリに記録され，bCapService の起動時に読み込まれます．

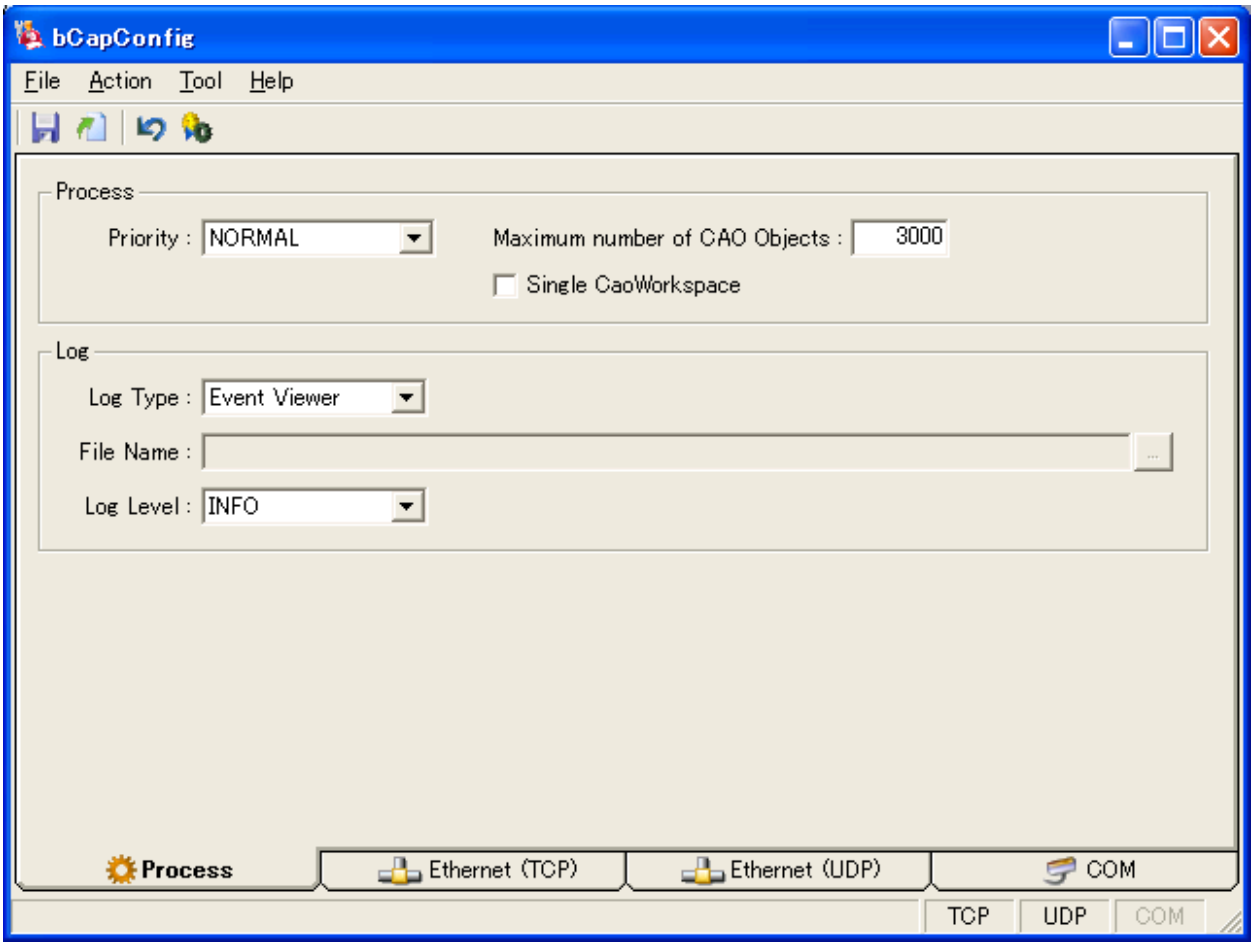

図 4-1 bCapConfig 画面

4.2. 操作方法

4.2.1. メニュー

4.2.1.1. File メニュー

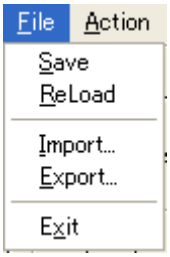

図 4-2 File メニュー

[保存] – Save

レジストリに bCapConfig の情報を設定します.

[再読み込み] – Reload

レジストリから bCapConfig の情報を取得します.

[XML インポート] – Import

XML ファイルから bCapConfig の情報を取得します.

[XML エクスポート] – Export

XML ファイルに bCapConfig の情報を設定します.

[終了] – Exit

bCapConfig を終了します.

4.2.1.2. Action メニュー

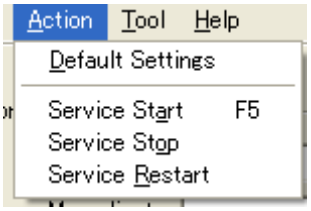

図 4-3 Action メニュー

[デフォルト設定] – Default Setting

現在の表示内容をデフォルトの設定内容に戻します.

[サービス開始] – Service Start

CAO サービスを開始します. CAO をサービスとして登録してある場合のみ, 使用できます.

[サービス停止] – Service Stop

CAO をサービス停止します. CAO をサービスとして登録してある場合のみ, 使用できます.

[サービス再起動] – Service Start

CAO サービスを再起動します. CAO をサービスとして登録してある場合のみ, 使用できます.

4.2.1.3. Tool メニュー

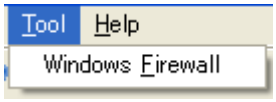

図 4-4 Tool メニュー

[Windows ファイアウォール設定] – Windows Firewall

Windows ファイアウォールの設定画面を表示します.

4.2.1.4. Help メニュー

図 4-5 Help メニュー

[バージョン] –**Version**

bCapConfig のバージョン情報を表示します.

4.2.2. タブ入力

4.2.2.1. Process タブ

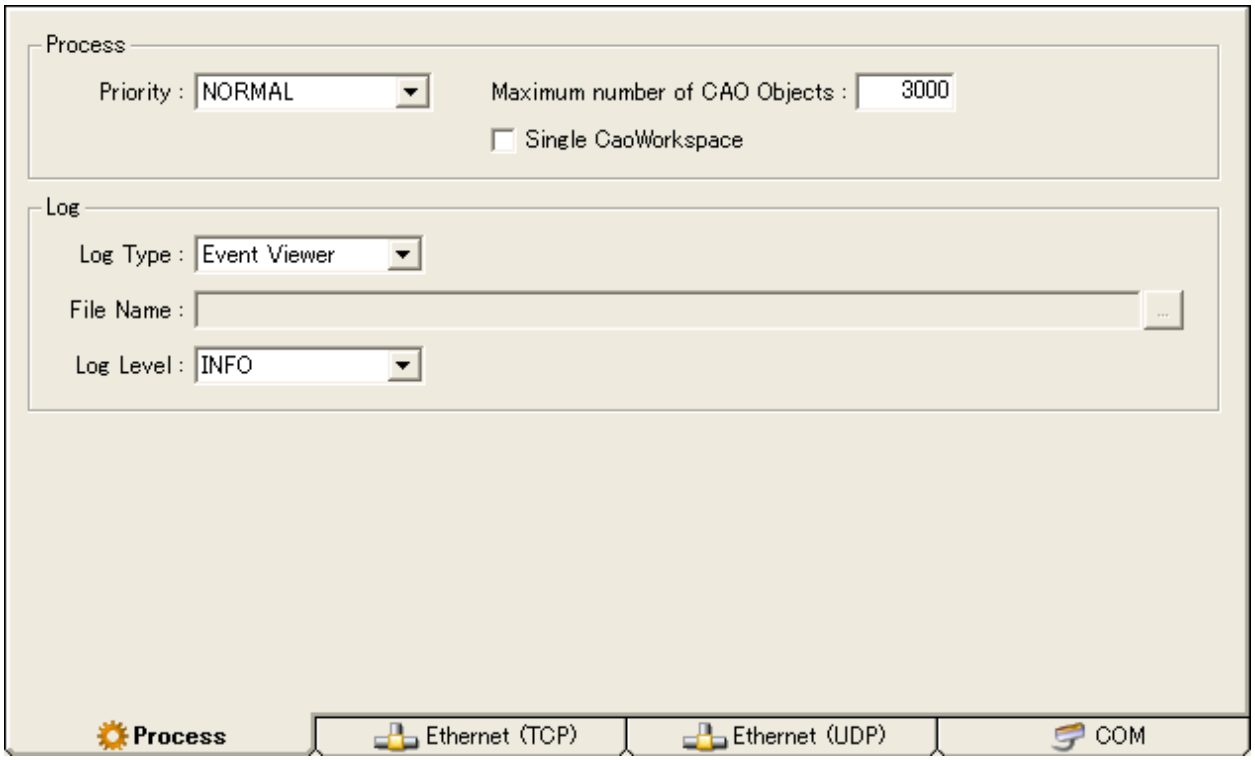

図 4-6 Process タブ

[プロセス優先度] – Process Priority

CAO エンジンのプロセス優先度を設定します. 優先度に対する調整は以下の通りです.

REAL TIME > HIGH > **NORMAL** > IDEL

[単一ワークススペース使用] – Single CaoWorkspace

bCapService がコントローラオブジェクトを生成する際にワークスペースオブジェクトを生成するかどうかを選択します.

この項目にチェックがないときは, コントローラ毎に個別のワークスペースが生成されます.

この項目にチェックがあるときは, すべてのコントローラが同一ワークスペース内に生成されます.

[ログタイプ] - Log Type

CAO.exe のログの出力タイプを選択します. ログの出力タイプは以下のものを選択することができます. 設定できる項目を以下に説明します.

表 4-1 ログタイプ

出力先	備考
Console	コンソールに出力します
Message Box	メッセージボックスに出力します(サービス起動時)
Event Viewer	イベントビューワに出力します(サービス起動時)
Debug Viewer	デバッグ出力します.
Text File	指定したテキストファイルに出力します.

[ログ出力ファイル名] – File Name

ログタイプが Text File の場合, ここにファイルパスを指定します.

[ログレベル] – Log Level

ログの出力レベルを設定します. (設定したログレベル以上のログを出力します)ログレベルの設定は, 以下の 5 つのレベルから選択することができます. “FATAL”がもっとも深刻度が高く, “DEBUG”に近づくほど深刻度は低くなります. 標準では“INFO”に設定されています.

FATAL > ERROR > WARN > **INFO** > DEBUG

4.2.2.2. Ethernet (TCP)タブ

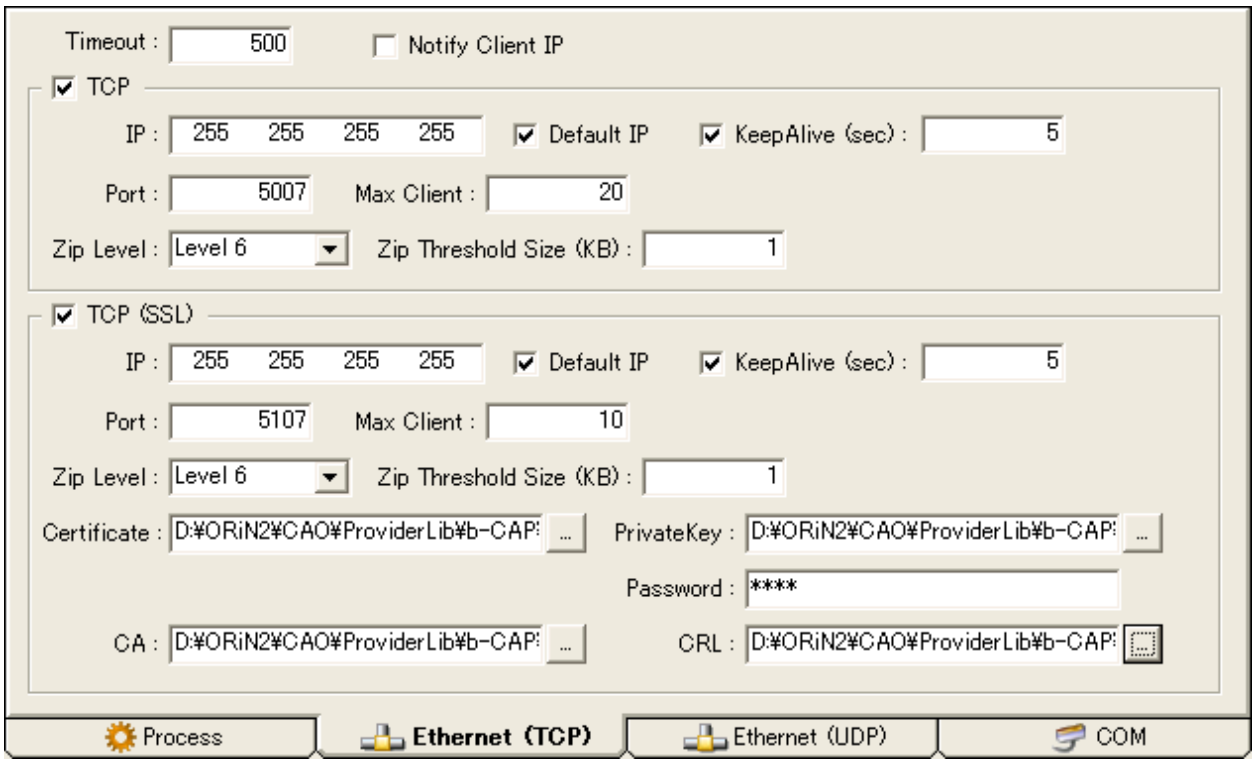

図 4-7 Ethernet (TCP)タブ

[TCP 設定] – TCP

b-CAP/TCP 通信設定を指定します. この設定がチェックされていないときは, bCapService.exe は, b-CAP/TCP 通信を行ないません.

[TCP 用 IP アドレスの設定] – TCP → IP

b-CAP/TCP 通信時に使用する IP アドレスを指定します. DefaultIP がチェックされているときは, この設定は無視されます.

[TCP 用デフォルト IP の設定] – TCP → DefaultIP

b-CAP/TCP 通信時にデフォルト IP アドレスの使用をするかを指定します. この設定がチェックされているときは, IP アドレスの設定は無視されます.

[TCP 用ポート番号の設定] – TCP → Port

b-CAP/TCP 通信時に使用する TCP ポート番号を指定します.

[TCP 最大クライアント数] – TCP → Max client

b-CAP/TCP 通信時にサーバに同時に接続できるクライアントの数を指定します.

[TCP 用キープアライブ設定] – KeepAlive

b-CAP/TCP 通信時のキープアライブ時間を設定します.
この設定にチェックが入っていないときはキープアライブを使用しません.

[TCP 用圧縮レベル設定] – TCP → Zip Level

b-CAP/TCP 通信時の圧縮レベルを設定します.

[TCP 用圧縮閾値サイズ設定] – TCP → Zip Threshold Size

b-CAP/TCP 通信時の圧縮閾値サイズを設定します.

[TCP (SSL)設定] – TCP（SSL）

b-CAP/TCP (SSL)通信設定を指定します. この設定がチェックされていないときは, bCapService.exe は, b-CAP/TCP (SSL)通信を行いません.

[TCP (SSL)用 IP アドレスの設定] – TCP (SSL) → IP

b-CAP/TCP (SSL)通信時に使用する IP アドレスを指定します. DefaultIP がチェックされているときは, この設定は無視されます.

[TCP (SSL)用デフォルト IP の設定] – TCP (SSL) → DefaultIP

b-CAP/TCP (SSL)通信時にデフォルト IP アドレスを使用するかを指定します. この設定がチェックされているときは, IP アドレスの設定は無視されます.

[TCP (SSL)用ポート番号の設定] – TCP (SSL) → Port

b-CAP/TCP (SSL)通信時に使用する TCP ポート番号を指定します.

[TCP (SSL)最大クライアント数] – TCP (SSL) → Max client

b-CAP/TCP (SSL)通信時にサーバに同時に接続できるクライアントの数を指定します.

[TCP (SSL)用キープアライブ設定] – TCP (SSL) → KeepAlive

b-CAP/TCP (SSL)通信時のキープアライブ時間を設定します.
この設定にチェックが入っていないときはキープアライブを使用しません.

[TCP (SSL)用圧縮レベル設定] – TCP (SSL) → Zip Level

b-CAP/TCP (SSL)通信時の圧縮レベルを設定します.

[TCP (SSL)用圧縮閾値サイズ設定] – TCP (SSL) → Zip Threshold Size

b-CAP/TCP (SSL)通信時の圧縮閾値サイズを設定します.

[証明書ファイル名設定] – TCP (SSL) → Certificate

b-CAP/TCP (SSL)通信時に使用する証明書ファイル名を設定します.

[秘密鍵ファイル名設定] – TCP (SSL) → PrivateKey

b-CAP/TCP (SSL)通信時に使用する秘密鍵ファイル名を設定します.

[秘密鍵パスワード設定] – TCP (SSL) → Password

b-CAP/TCP (SSL)通信時に使用する秘密鍵のパスワードを設定します.

[証明機関(CA)証明書ファイル名設定] – TCP (SSL) → CA

b-CAP/TCP (SSL)通信時に使用する証明機関(CA)の証明書ファイル名を設定します.

[証明書失効リストファイル名設定] – TCP (SSL) → CRL

b-CAP/TCP (SSL)通信時に使用する証明書失効リストファイル名を設定します.

[通信タイムアウト時間の設定] – Timeout

通信時に使用するタイムアウト時間を指定します.

4.2.2.3. Ethernet (UDP)タブ

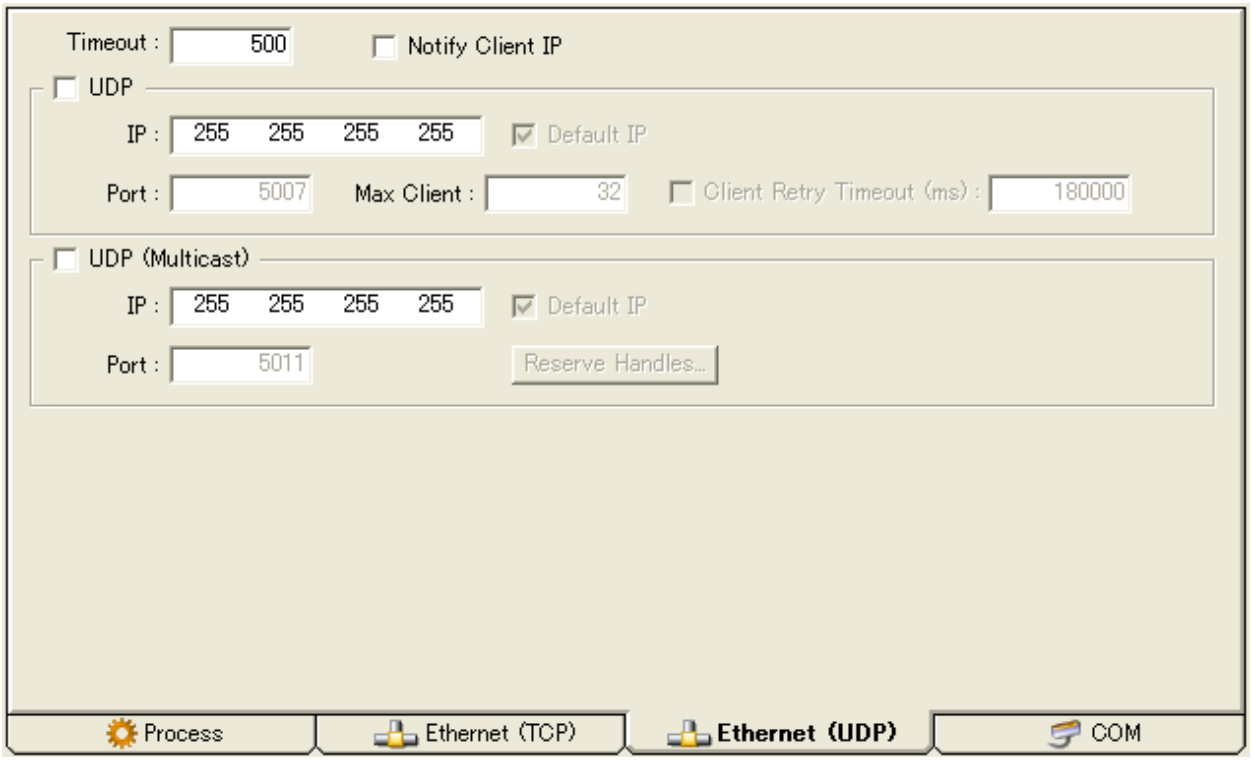

図 4-8 Ethernet (UDP)タブ

[UDP 設定] – UDP

b-CAP/UDP 通信設定を指定します. この設定がチェックされていないときは, bCapService.exe は, b-CAP/UDP 通信を行ないません.

[UDP 用 IP アドレスの設定] – UDP → IP

b-CAP/UDP 通信時に使用する IP アドレスを指定します. DefaultIP がチェックされているときは, この設定は無視されます.

[UDP 用デフォルト IP の設定] – UDP → DefaultIP

b-CAP/UDP 通信時にデフォルト IP アドレスを使用するかを指定します. この設定がチェックされているときは, IP アドレスの設定は無視されます.

[UDP 用ポート番号の設定] – UDP → Port

b-CAP/UDP 通信時に使用する UDP ポート番号を指定します.

[UDP 最大接続数の設定] – UDP → Max Client

b-CAP/UDP 通信時に同時に接続できるクライアント数を指定します.

[UDP リトライ用実行結果保持時間] – UDP → Client Retry Timeout (ms)

b-CAP/UDP 通信時でリトライが発生した時に返す実行結果の保持時間を指定します.

この保持時間内にリトライが発生した場合, コマンドは実行されず, 直前の実行結果を返します.

この設定に対し, クライアントアプリケーションの通信タイムアウト時間が短い場合, リトライにより予期せぬタイミングでコマンドが実行されることがあります.

[UDP (Multicast)設定] – UDP (Multicast)

b-CAP/UDP (Multicast) 通信設定を指定します. この設定がチェックされていないときは, bCapService.exe は, b-CAP/UDP(Multicast)通信を行ないません.

[UDP (Multicast)用 IP アドレスの設定] – UDP (Multicast) → IP

b-CAP/UDP (Multicast)通信時に使用する IP アドレスを指定します. DefaultIP がチェックされているときは, この設定は無視されます.

[UDP (Multicast)用デフォルト IP の設定] – UDP (Multicast) → DefaultIP

b-CAP/UDP (Multicast)通信時にデフォルト IP アドレスを使用するかを指定します. この設定がチェックされているときは, IP アドレスの設定は無視されます.

[UDP (Multicast)用ポート番号の設定] – UDP (Multicast) → Port

b-CAP/UDP (Multicast)通信時に使用する UDP ポート番号を指定します.

[UDP (Multicast)用予約ハンドル設定] – UDP (Multicast) → Reserve Handles…

b-CAP/UDP (Multicast)通信時に使用する予約ハンドルを設定します.

押下することにより予約ハンドル入力ダイアログが表示されます.

詳細については 4.2.2.5 を参照してください.

[通信タイムアウト時間の設定] – Timeout

通信時に使用するタイムアウト時間を指定します.

4.2.2.4. COM タブ

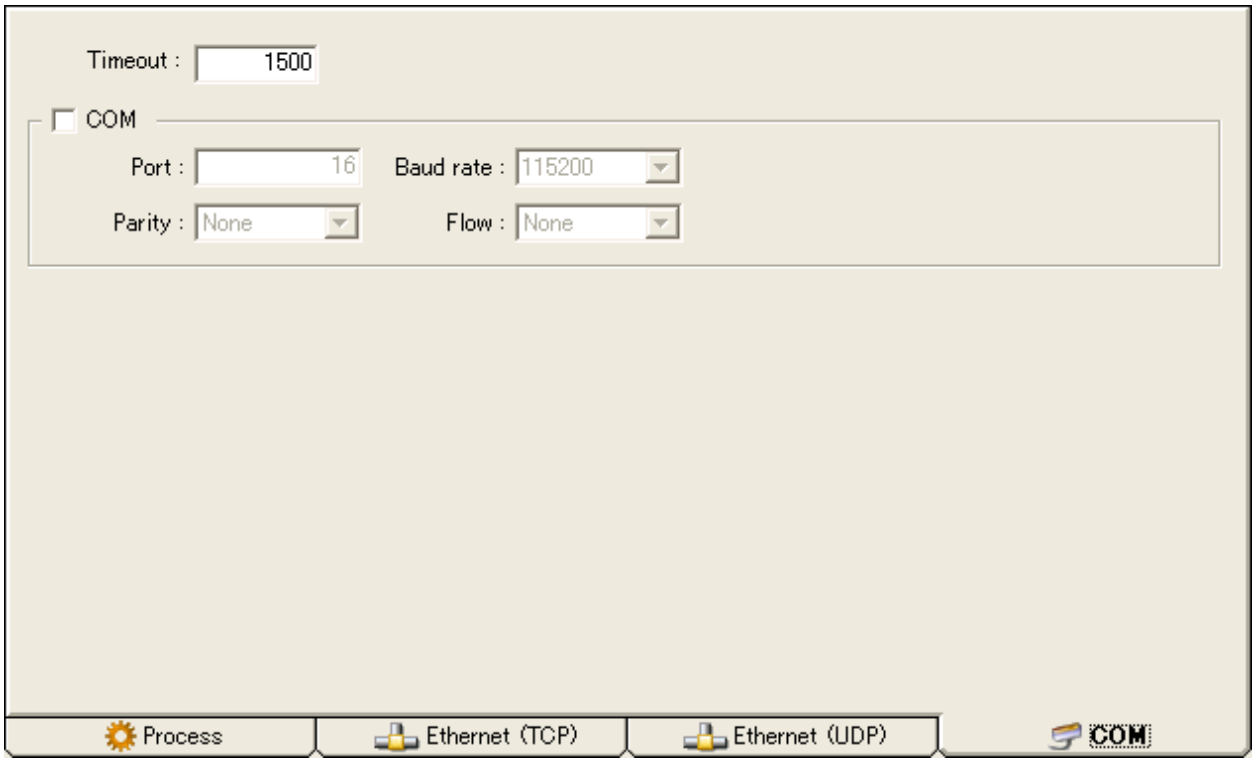

図 4-9 COM タブ

[COM 設定] – COM

b-CAP/COM 通信設定を指定します. この設定がチェックがされていないときは, bCapService.exe は, b-CAP/COM 通信を行ないません.

[COM 用ポート番号の設定] – COM → Port

b-CAP/COM 通信時に使用する COM ポート番号を指定します.

[COM 用ボーレートの設定] – COM → Baud rate

b-CAP/COM 通信時に使用するボーレートを指定します.

[COM 用パリティの設定] – COM → Parity

b-CAP/COM 通信時に使用するパリティの種類を指定します.

[COM 用フロー制御の設定] – COM → Flow

b-CAP/COM 通信時に使用するフロー制御の種類を指定します.

[通信タイムアウト時間の設定] – Timeout

通信時に使用するタイムアウト時間を指定します.

4.2.2.5. 予約ハンドル入力ダイアログ

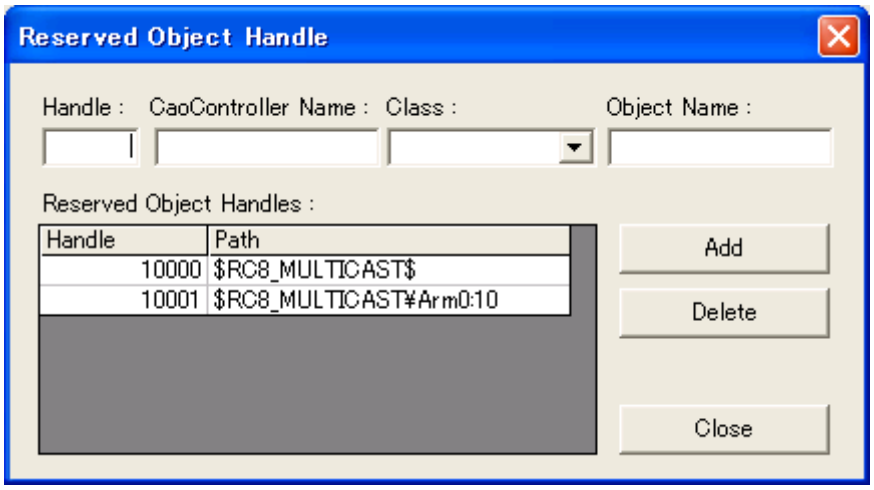

図 4-10 予約ハンドル入力ダイアログ

[ハンドル番号] – Handle

追加するハンドル番号を指定します.

[CAO コントローラ名] – CaoController Name

追加する CAO コントローラ名を指定します.

[クラス] – Class

追加するクラスを一覧から選択します.

表 4-2 クラス

値	クラス名
6	Extension
8	File
10	Robot
12	Task
14	Variable
16	Command

[オブジェクト名] – Object

追加するオブジェクト名を指定します.

[予約ハンドル一覧] – Reserved Object Handles

予約ハンドルの一覧を表示します.

[追加] – Add

入力されたハンドルを予約ハンドル一覧に追加します.

[削除] – Delete

選択された予約ハンドルを予約ハンドル一覧から削除します.

[閉じる] – Close

このダイアログを閉じます.

5. サンプルプログラム

以下にサーバ(IP アドレス:10.8.109.116)で DataStore プロバイダを起動し，変数への値の設定，取得を行うサンプルを示します.

このときサーバで bCapListerner が起動済みであるとします.

List 5-1 Sample.frm

```
Private eng As CaoEngine
Private ctrl As CaoController
Private var As CaoVariable

Private Sub Form_Load()

    Dim ws As CaoWorkspace

    Set eng = New CaoEngine
    Set ws = eng.Workspaces(0)

    ' サーバと接続
    Set ctrl = ws.AddController("RC1", _
                                "CaoProv.b-CAP", _
                                "", _
                                "Provider=CaoProv.DataStore,Server=10.8.109.116")

    ' 変数の取得
    Set var = ctrl.AddVariable("Var1")

End Sub

' 変数の設定
Private Sub Command1_Click()
    var = Text1.Text
End Sub

' 変数の取得
Private Sub Command2_Click()
    Text1.Text = var
End Sub
```

また, 以下に SSL 通信を使用したサンプルを示します.

List 5-2 Sample2.frm

```
Private eng As CaoEngine
Private ctrl As CaoController
Private var As CaoVariable

Private Sub Form_Load()

    Dim ws As CaoWorkspace

    Set eng = New CaoEngine
    Set ws = eng.Workspaces(0)

    ' サーバと接続
    Set ctrl = ws.AddController("RC1", _
                                "CaoProv.b-CAP", _
                                "", _
                                "Provider=CaoProv.DataStore,Server=10.8.109.116,5107,SSL," _
                                 & "Certificate=C:¥store¥client.pem," _
                                 & "PrivateKey=C:¥store¥client.pem," _
                                 & "Password=pass," _
                                 & "CA=C:¥store¥rootcert.pem")

    ' 変数の取得
    Set var = ctrl.AddVariable("Var1")

End Sub

' 変数の設定
Private Sub Command1_Click()
    var = Text1.Text
End Sub

' 変数の取得
Private Sub Command2_Click()
    Text1.Text = var
End Sub
```

6. 付録

6.1. SSL によるセキュア通信

以下に SSL によるセキュア通信のセットアップ方法を示します.

ここでは以下の説明を行います.

概要

必要ファイルの作成

ルート CA 証明書と証明書失効リストの作成

サーバ証明書と秘密鍵の作成

クライアント証明書と秘密鍵の作成

6.1.1. 概要

ルート CA 証明書と証明書失効リストの作成, その証明書で署名したサーバ証明書とクライアント証明書を作成する手順を説明します.

作成には openssl.exe を使用します.

以下, C:¥に ORiN2 がインストールされているとして説明します.

パラメータは適宜変更してください.

コマンドの詳細については openssl.exe のヘルプを参照ください.

本章でのコマンドの実行は C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.cnf のディレクトリの設定(45 行目)を,

```
dir             = ./demoCA              # Where everything is kept
→
dir             = .                     # Where everything is kept
```

と修正した状態で行った説明としております.

コマンド実行時は必要ファイルを出力する任意のフォルダで行ってください.

本章の説明は検証での使用を前提としております.

6.1.2. 必要ファイルの作成

コマンド実行時に入力・出力に必要となるファイルの作成手順を説明します.

6.1.2.1. index.txt の作成

「署名した証明書のデータベースインデックスファイル」を作成します.

以下のコマンドを実行します.

```
type nul > index.txt
```

6.1.2.2. crlnumber の作成

「次回 CRL 番号ファイル」を作成します.

以下のコマンドを実行します.

```
echo 00 > crlnumber
```

6.1.2.3. serial の作成

「証明書のシリアル番号ファイル」を作成します.

以下のコマンドを実行します.

```
echo 01 > serial
```

6.1.3. ルート CA 証明書と証明書失効リストの作成

サーバ証明書, クライアント証明書に署名する為のルート CA 証明書と証明書失効リストを作成します.

rootcert.pem => ルート CA 証明書

rootcrl.crl => 証明書失効リスト

となります.

6.1.3.1. ルート CA の証明書署名要求と秘密鍵の作成

ルート CA 証明書の作成に必要な証明書署名要求と秘密鍵を作成します.

rootreq.csr => ルート CA 証明書署名要求

rootkey.pem => ルート CA 秘密鍵

となります.

例)

- 有効期限:10 年(3650 日)
- 出力証明書署名要求ファイル:rootreq.csr
- 発行者:国-JP, 市町村-Your Loccality, 都道府県-Your State, 組織-Your Organization, 部門-Your Division, 名称-Your CA
- 暗号鍵強度:RSA(1024 ビット)
- ハッシュ関数:SHA1
- 出力秘密鍵ファイル:rootkey.pem
- パスワード:password

以下のコマンドを実行します.

```
C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.exe req -config
"C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.cnf" -new -days 3650 -out rootreq.csr -subj "/C=JP/L=Your
Locality/ST=Your State/O=Your Organization/OU=Your Division/CN=Your CA" -newkey rsa:1024
-sha1 -keyout rootkey.pem -passout pass:password
```

6.1.3.2. ルート CA 証明書の作成

作成した証明書署名要求と秘密鍵を使用してルート CA 証明書を作成します.

rootcert.pem => ルート CA 証明書

となります.

例)

- 入力証明書署名要求ファイル:rootreq.csr
- ハッシュ関数:SHA1
- 入力秘密鍵ファイル:rootkey.pem
- 出力ルート CA 証明書ファイル:rootcert.pem

- パスワード:password
- 有効期限:10 年(3650 日)

以下のコマンドを実行します.

```
C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.exe x509 -req -in rootreq.csr -sha1 -extfile
"C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.cnf" -extensions v3_ca -signkey rootkey.pem -CAserial serial
-out rootcert.pem -passin pass:password -days 3650
```

6.1.3.3. 証明書失効リストの作成

作成したルート CA 証明書とルート CA 秘密鍵を使用して証明書失効リストを作成します.

rootcrl.crl => 証明書失効リスト

となります.

例)

- 有効期限:1 年(365 日)
- 出力証明書失効リストファイル:rootcrl.crl
- 入力ルート CA 証明書ファイル:rootcert.pem
- 入力ルート CA 秘密鍵ファイル:rootkey.pem
- パスワード:password

以下のコマンドを実行します.

```
C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.exe ca -config
"C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.cnf" -gencrl -crldays 365 -out rootcrl.crl -cert rootcert.pem
-keyfile rootkey.pem -passin pass:password
```

6.1.4. サーバ証明書とサーバ秘密鍵の作成

ルート CA 証明書とルート CA 秘密鍵を使用して, サーバ証明書とサーバ秘密鍵を作成します.

servercert.pem => サーバ証明書

serverkey.pem => サーバ秘密鍵

となります.

6.1.4.1. サーバの証明書署名要求と秘密鍵の作成

サーバ証明書の作成に必要な証明書署名要求と秘密鍵を作成します.

serverreq.csr => サーバ証明書署名要求

serverkey.pem => サーバ秘密鍵

となります.

例)

- 出力証明書署名要求ファイル:serverreq.csr
- 発行者:国-JP, 市町村-Your Loccality, 都道府県-Your State, 組織-Your Organization, 部門-Your Division, 名称-bCAP
- 暗号鍵強度:RSA(1024 ビット)
- ハッシュ関数:SHA1
- 出力秘密鍵ファイル:serverkey.pem
- パスワード:password

以下のコマンドを実行します.

サーバ証明書の発行者の名称は必ず「bCAP」としてください.

```
C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.exe req -config
"C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.cnf" -new -out serverreq.csr -subj "/C=JP/L=Your
Locality/ST=Your State/O=Your Organization/OU=Your Division/CN=bCAP" -newkey rsa:1024 -sha1
-keyout serverkey.pem -passout pass:password
```

6.1.4.2. サーバ証明書の作成

作成した証明書署名要求と秘密鍵を使用してサーバ証明書を作成します.

servercert.pem => サーバ証明書

となります.

例)

- 入力証明書署名要求ファイル:serverreq.csr
- 出力サーバ証明書ファイル:servercert.pem
- パスワード:password
- 有効期限:10 年(3650 日)
- 証明書失効リスト有効期限:10 年(3650 日)
- ルート CA 証明書ファイル:rootcert.pem
- ルート CA 秘密鍵ファイル:rootkey.pem
- 出力ディレクトリ:.(ドット)

以下のコマンドを実行します.

```
C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.exe ca -config
"C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.cnf" -extensions usr_cert -batch -in serverreq.csr -out
servercert.pem -passin pass:password -days 3650 -crldays 3650 -cert rootcert.pem -keyfile rootkey.pem
-outdir .
```

6.1.5. クライアント証明書とクライアント秘密鍵の作成

ルート CA 証明書とルート CA 秘密鍵を使用して, クライアント証明書とクライアント秘密鍵を作成します.

clientcert.pem => クライアント証明書

clientkey.pem => クライアント秘密鍵

となります.

6.1.5.1. クライアントの証明書署名要求と秘密鍵の作成

クライアント証明書の作成に必要な証明書署名要求と秘密鍵を作成します.

clientreq.csr => クライアント証明書署名要求

clientkey.pem => クライアント秘密鍵

となります.

例)

- 出力証明書署名要求ファイル:clientreq.csr
- 発行者:国-JP, 市町村-Your Loccality, 都道府県-Your State, 組織-Your Organization, 部門-Your Division, 名称-Your CommonName

- 暗号鍵強度:RSA(1024 ビット)
- ハッシュ関数:SHA1
- 出力秘密鍵ファイル:clientkey.pem
- パスワード:password

以下のコマンドを実行します.

```
C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.exe req -config
"C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.cnf" -new -out clientreq.csr -subj "/C=JP/L=Your
Locality/ST=Your State/O=Your Organization/OU=Your Division/CN=Your CommonName" -newkey
rsa:1024 -sha1 -keyout clientkey.pem -passout pass:password
```

6.1.5.2. クライアント証明書の作成

作成した証明書署名要求と秘密鍵を使用してクライアント証明書を作成します.

clientcert.pem => サーバ証明書

となります.

例)

- 入力証明書署名要求ファイル:clientreq.csr
- 出力サーバ証明書ファイル:clientcert.pem
- パスワード:password
- 有効期限:10 年(3650 日)
- 証明書失効リスト有効期限:10 年(3650 日)
- ルート CA 証明書ファイル:rootcert.pem
- ルート CA 秘密鍵ファイル:rootkey.pem
- 出力ディレクトリ:.(ドット)

以下のコマンドを実行します.

```
C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.exe ca -config
"C:¥ORiN2¥Tools¥OpenSSL¥Bin¥openssl.cnf" -extensions usr_cert -batch -in clientreq.csr -out
clientcert.pem -passin pass:pass -days 3650 -crldays 3650 -cert rootcert.pem -keyfile rootkey.pem
-outdir .
```

6.1.6. 各アプリケーションの設定と接続

作成した

- ルート CA 証明書ファイル(rootcert.pem)
- サーバ/クライアント証明書ファイル(servercert.pem/clientcert.pem)
- サーバ/クライアント秘密鍵ファイルとそのパスワード(serverkey.pem/clientkey.pem)(サーバ/クライアントの証明書署名要求と秘密鍵作成時に入力したパスワード)
- 証明書失効リストファイル(サーバのみ)(rootcrl.crl)

を指定して各アプリを起動し接続します.

設定方法については

- bCapListener.exe 3.2.1
- bCapService.exe => bCapConfig 4.2.2.2

・ CaoProvBCAP 2.2.1

の設定方法を参照してください.