JRCS
SMS-55 プロバイダ
ユーザーズ ガイド

Version 1.0.0

November 12, 2021

備考:

【改版履歴】

バージョン	日付	内容
1.0.0	2021-11-12	初版.

【対応機種】

機種	バージョン	注意事項
SMS-55	--	SMS-55 との接続にシリアル Ethernet 変換器を使用し UDP 接続のみ対応

【動作確認機種】

機種	バージョン	注意事項
SMS-55	--	シリアル Ethernet 変換器として MOXA 製 NPort5150 を使用

目次

1. はじめに

本書は，JRCS 社の船舶用アラーム・モニタリング＆コントロールシステム(AMS,AMCS)である SMS-55 デバイスからチャネルデータを受信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 SMS-55 プロバイダと呼称します．

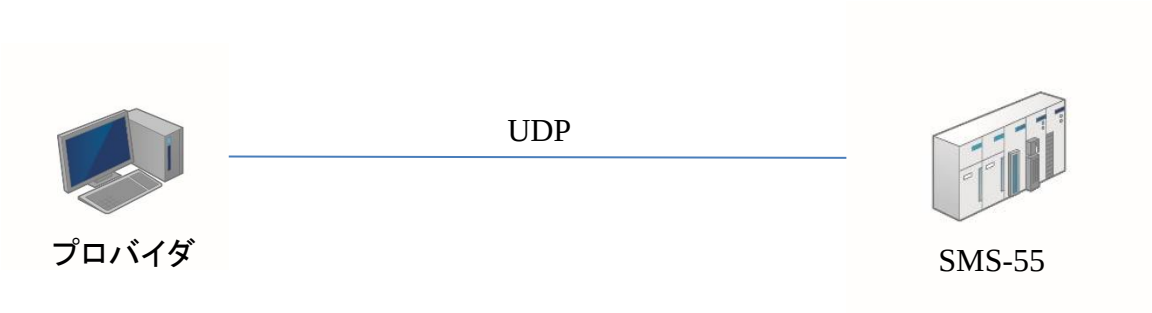

図 1-1 構成図

また，本プロバイダ及びデバイスそれぞれの対応を図 1-2 に表します．
(※一例です．全てを表しているわけではありません．)

図 1-2 プロバイダの構成とデバイス情報との対応図

1.1. SMS-55 の送信データ概要

SMS-55 は，一定周期ごとに日付と複数のチャネルデータをシリアルデータとして送信します．図 1-3 が送信データの概要図になります．

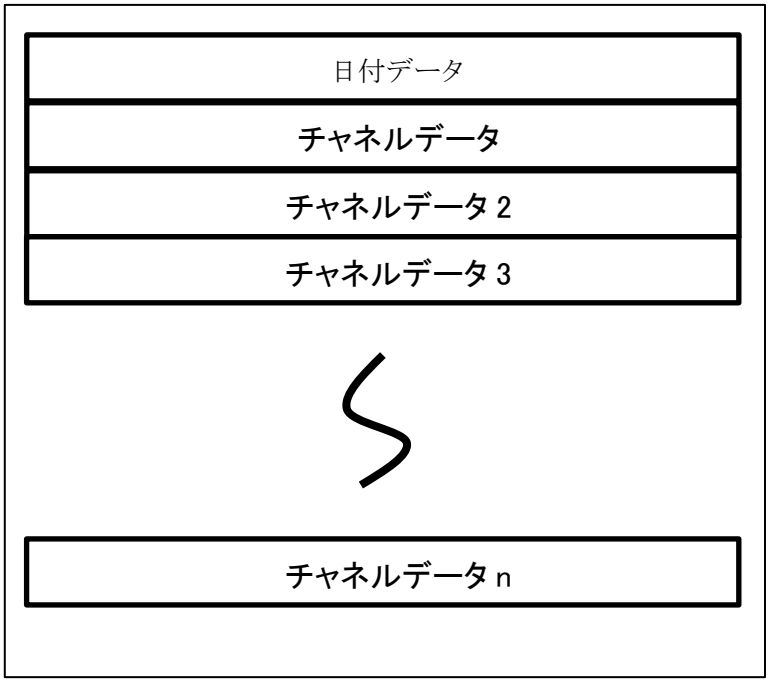

図 1-3 SMS-55 送信データ概要

1.1.1. チャネルデータ概要

チャネルデータは 9 文字のデータで構成されおり, 先頭 1 文字をステータス, 残り 8 文字をデータとして扱います. 図 1-4 の例では, 先頭 1 文字目のステータスが「R」, データが「_ _ _ _ _ 9 9 9」("_" はスペース)のチャネルデータということになります.

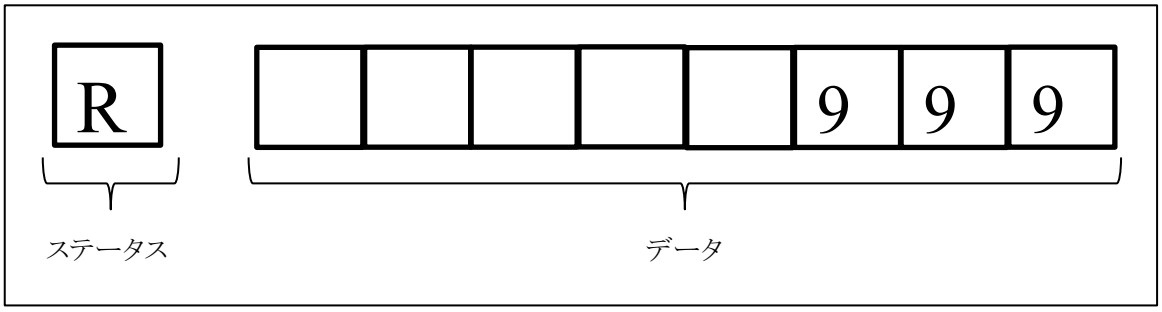

図 1-4 1 チャネルデータ構成(例)

表 1-1 はステータスの例になります. ステータスの詳細に関しては, SMS-55 の製造元へお問い合わせください.

表 1-1 ステータス(例)

ステータス	説明
_ (20H)	正常データ

* (2AH)	異常データ
R (52H)	停止中データ

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. SMS-55 とクライアント PC との接続

SMS-55 から本プロバイダに接続するには，SMS-55 から送信される RS422 のシリアルデータを UDP に変換する必要があります．本書では，RS422 から UDP への変換器として MOXA 製「NPort5150」を使用し接続確認をしております．

本プロバイダは，UDP サーバとして運用しているため接続先が SMS-55 かどうかは判定しておらず指定した受信ポートに送られてくるデータに対して解析処理を行っています．

3. コマンドリファレンス

3.1. メソッド/プロパティー覧

表 3-1 メソッド/プロパティー覧

カテゴリ	メソッド/プロパティ[1]		機能	参照
CaoWorkspace				
	AddController	M	コントローラに接続	P.10
CaoController				
	Variables	P	コントローラが保持する変数コレクションの取得	P.11
	AddVariable	M	変数オブジェクトの追加	P.11
	OnMessage	E	メッセージ受信イベント	P.12
CaoVariable				
	Value	P	値の取得/設定	P.12

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．SMS-55 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，該当すると SMS-55 との接続を行います．以下に，AddController メソッドの仕様を示します．

書式

```
AddController
(
"<コントローラ名>",          // コントローラ名(任意)
"CaoProv.JRCS.SMS-55",      // 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)
"<オプション>"               // オプション文字列(省略可能)
)
```

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．

[1] 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	必須	説明	値範囲
MyPort=<受信ポート番号>	-	SMS-55 からデータを受信する UDP ポート番号を指定します.	1-65535 デフォルト値:1024
ReceiveWait=<受信待機時間>	-	SMS-55 からデータ受信を開始し, 途中でデータが受信されなくなった場合の最大待機時間(ms)を指定します. ※データ受信中に最後に受信したデータから指定時間経過した場合, タイムアウトエラーメッセージを発行し受信データを破棄します.	0-ULONG_MAX デフォルト値:10000

使用例(C#)

```
// Engine オブジェクト
ORiN2.ManagedCAO.CCaoEngine engine = new ORiN2.ManagedCAO.CCaoEngine();
// Workspace オブジェクト
ORiN2.ManagedCAO.CCaoWorkspace workspace = engine.AddWorkspace("NewWrks", "");
// Controller オブジェクト
ORiN2.ManagedCAO.CCaoController controller= workspace.AddController("SMS-55",
                                                    "CaoProv.JRCS.SMS-55",
                                                    "",
                                                    "");
```

3.2.2. CaoController クラス

3.2.2.1. Variables プロパティ

コントローラが保持する, 変数コレクションを取得します.

使用例(C#)

```
// 変数コレクション取得
ORiN2.ManagedCAO.CCaoVariables variables = controller.Variables;
// 変数取得
ORiN2.ManagedCAO.CCaoVariable variable = variables[0];
```

3.2.2.2. 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 3.3.2 に示すもののみ使用できます.

以下に, AddVariable の仕様を示します.

書式

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

3.2.2.3. OnMessage イベント

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. 受け取れるイベントについては 3.4 を参照してください.

3.2.3. CaoVariable クラス

3.2.3.1. Value プロパティ

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.

3.3.1. システム変数

本プロバイダにはシステム変数のみ存在します.

システム変数

その変数を保持するオブジェクト内で唯一の情報にアクセスための変数です. 本プロバイダのシステム変数はすべて静的データとなります. システム変数は名前の先頭に"@"がついています.

3.3.2. CaoController クラス変数

変数名	説明	Value		参照
		get	put	
@MAKER_NAME	メーカー名を取得します.	○	-	P.12
@VERSION	DLL バージョンを取得します.	○	-	P.13

3.3.2.1. @MAKER_NAME

メーカー名の取得をします.

データ型

型説明	
VT_BSTR	メーカー名を取得します.

使用例(C#)

```
// 変数追加
ORiN2.ManagedCAO.CCaoVariable var = controller.AddVariable("@MAKER_NAME","");
// 値取得
string value = var.Value as string;
```

3.3.2.2. @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(C#)

```
// 変数追加
ORiN2.ManagedCAO.CCaoVariable var = controller.AddVariable("@VERSION","");
// 値取得
string value = var.Value as string;
```

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.

番号	説明
0	受信データ SMS-55 から正常に受信したデータになります.
101	受信中タイムアウトメッセージ
102	受信データ異常エラー 受信したデータが想定外の場合のエラーメッセージになります.
103	日付データ異常メッセージ 受信した日付データが変換できない場合のエラーになります.
104	チャネルデータ異常メッセージ 受信した 1 チャネルデータが 9 文字のデータで構成されていなかった場合のエラーになります.

3.4.1.1. 受信データメッセージ

受信したパケットデータを日付, チャネルデータのスタータス配列, チャネルデータのデータ配列のメッセージとして発行します.

Value プロパティデータ型

型説明			
VT_ARRAY | VT_VARIANT			
0	VT_DATE		日付
1	VT_ARRAY | VT_BSTR		受信データのステータス部
	0	VT_BSTR	0 番目のチャネルデータのステータス
	~	…	
	n	VT_BSTR	n 番目のチャネルデータのステータス
2	VT_ARRAY | VT_BSTR		受信データのデータ部
	0	VT_BSTR	0 番目のチャネルデータ
	~	…	
	n	VT_BSTR	n 番目のチャネルデータ

使用例(C#)

```
/// <summary>
///  受信時の処理メソッド例
/// </summary>
/// <param name="chData">受信データ</param>
private void OnReceivedData(object[] chData)
{
    CDateTime date = Convert.ToDateTime(chData[0]);//日付データ
    String[] status = chData[1] as String[];//ステータス部配列
    String[] data = chData[2] as String[];//データ部配列
}
```

4. SMS-55 プロバイダによるプログラミング

SMS-55 プロバイダでは，以下の手順でクライアント PC と SMS-55 を接続することができます．

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

SMS-55 に接続した後は，プロバイダは SMS-55 から送信されてくるチャネルデータをメッセージとして発行します．発行されたメッセージは，CaoProvMessage のイベントにより取得することが可能となります．

4.1. チャネルデータを受信するサンプルプログラミング

ここでは例として発行されたメッセージをのグローバル OM のデータレジスタの値を読み書きするサンプルプログラムを示します．表 4-1 にサンプルプログラムの要件を，図 4-1 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 4-1 サンプルプログラムの要件

要件	説明
接続先	UDP で接続する
接続元	受信ポート番号は 4001
処理内容	デバイスからデータを受信
	受信したデータを要素ごとに格納

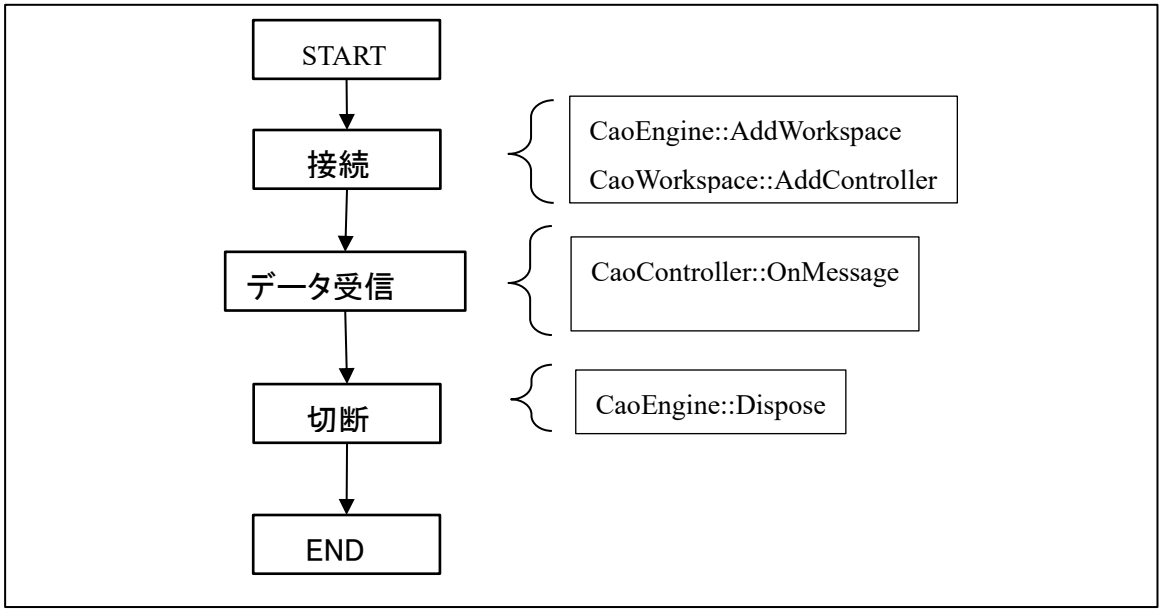

図 4-1 データ受信の流れ

以降の節から具体的なコードを示します．

4.1.1. 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 SMS55Sample.cs

```
// オブジェクト
private ORiN2.ManagedCAO.CCaoEngine m_caoEngine = null;
private ORiN2.ManagedCAO.CCaoWorkspace m_caoWorkspace = null;
private ORiN2.ManagedCAO.CCaoController m_caoController = null;

public void Main()
{
    // 接続
    this.Connect();

  // コントローラに Onmessage イベントを受信した際の動作を追加
    m_caoController.OnMessage += new ORiN2.ManagedCAO.OnMesssageEventHandler(O
nMessageEvent);
    // OnMessage イベントを受け付けるために 60 秒間待機
    Thread.Sleep(60000);
    // 切断
    this.Disconnect();
}

// 接続メソッド
private void Connect()
{
    // CaoEngine オブジェクトの生成
    this.m_caoEngine = new ORiN2.ManagedCAO.CCaoEngine();
    // CaoWorkspace オブジェクトの生成
    this.m_caoWorkspace = this.m_caoEngine.AddWorkspace("NewWrks", "");
    // CaoController オブジェクトの生成
    this.m_caoController = this.m_caoWorkspace.AddController("SMS-55",
                                                    "CaoProv.JRCS.SMS-55",
                                                    "",
                                                    "MyPort=4001");
}
```

```
// 切断メソッド
private void Disconnect()
{
    this.m_caoEngine.Dispose();
    this.m_caoEngine = null;
}

/// <summary>
/// OnMessage 受信イベント
/// </summary>
private void OnMessageEvent(object sender, ORiN2.ManagedCAO.OnMessageEventArgs e)
{
    If(e.Message.Number == 0)
    {
        // メッセージの内容を取得
        Object[] messageData = e.Message.Value as Object[];
        // 日付データ
        DateTime date = Convert.ToDateTime(messageData[0]);
        // チャネルデータ(ステータス)
        String[] statusArray = messageData[1] as String[];
        // チャネルデータ(データ)
        String[] statusArray = messageData[2] as String[];
    }
}
```

4.1.1.1. 接続

SMS-55 と接続するためには，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．
コントローラ接続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．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．

使用例(C#)

```
// CaoEngine オブジェクト用の変数
```

```
private ORiN2.ManagedCAO.CCaoEngine m_caoEngine = null;
// CaoWorkspace オブジェクト用の変数
private ORiN2.ManagedCAO.CCaoWorkspace m_caoWorkspace = null;
// CaoController オブジェクト用の変数
private ORiN2.ManagedCAO.CCaoController m_caoController = null;
```

(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.

使用例(C#)

```
// CaoEngine オブジェクトの生成
this.m_caoEngine = new ORiN2.ManagedCAO.CCaoEngine();
```

(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.

使用例(C#)

```
// CaoWorkspace オブジェクトの生成
this.m_caoWorkspace = this.m_caoEngine.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. SMS-55 プロバイダでは, MyPort オプションでデータを受信する UDP ポートを指定します. 以下にコード例を示します.

使用例(C#)

```
// CaoController オブジェクトの生成
this.m_caoController = this.m_caoWorkspace.AddController("SMS-55",
                                                          "CaoProv.JRCS.SMS-55",
                                                          "",
                                                          "MyPort=4001");
```

4.1.1.2. データの受信

(1) SMS-55 プロバイダでは, デバイスからの送信データを受信した際に OnMessage イベントが発行されます. OnMessage イベントを受信するためには CaoController オブジェクトの OnMessage イベントハンドラにイベント受信時の動作を設定します. 以下にコード例を示します.

使用例(C#)

```
// コントローラに Onmessage イベントを受信した際の動作を追加
m_caoController.OnMessage += new ORiN2.ManagedCAO.OnMesssageEventHandler(OnMessageEvent);
```

(2) OnMessage イベントが発行されると上記で登録したイベントが実行され，取得データは OnMessage の Value プロパティで取得することができます．以下にコードの例を示します．

使用例(C#)

```
/// <summary>
/// OnMessage 受信イベント
/// </summary>
private void OnMessageEvent(object sender, ORiN2.ManagedCAO.OnMessageEventArgs e)
{
    if(e.Message.Number == 0)
    {
        // メッセージの内容を取得
        Object[] messageData = e.Message.Value as Object[];
        // 日付データ
        DateTime date = Convert.ToDateTime(messageData[0]);
        // チャネルデータ(ステータス)
        String[] statusArray = messageData[1] as String[];
        // チャネルデータ(データ)
        String[] dataArray = messageData[2] as String[];
    }
}
```

4.1.1.3. 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．ただし，ORiN.ManagedCAO を使用した場合は明示的に削除する必要はありません．以下にコード例を示します．

使用例(C#)

```
// CaoEngine からすべてのオブジェクトを削除
this.m_caoEngine.Dispose();
// CaoEngine の消去
this.m_caoEngine = null;
```

5. SMS-55 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 5-1 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 5-1 独自エラーコード表

エラー番号	説明
0x80110001	MyPort オプションエラー MyPort オプション指定が間違っています.
0x80110002	ReceiveWait オプションエラー ReceiveWait オプション指定が間違っています.