

NX Context-based Data Management
System for Embedded device ユーザーズ
ガイド

IOT-7-0010

■ 対象製品

S-741E-10P 組込み装置向けデータ管理基盤 01-00 以降（適用 OS：Windows 10）

■ 輸出時の注意

本製品を輸出される場合には、外国為替及び外国貿易法の規制並びに米国輸出管理規則など外国の輸出関連法規をご確認の上、必要な手続きをお取りください。

なお、不明な場合は、弊社担当営業にお問い合わせください。

■ 商標類

HITACHI は、株式会社 日立製作所の商標または登録商標です。

Microsoft、Visual Studio、Windows は、米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

その他記載の会社名、製品名などは、それぞれの会社の商標もしくは登録商標です。

■ マイクロソフト製品の表記について

このマニュアルでは、マイクロソフト製品の名称を次のように表記しています。

表記	製品名
Visual Studio	Microsoft(R) Visual Studio(R) 2015
Windows 10	Microsoft(R) Windows(R) 10 IoT Enterprise 2016 LTSB (64bit)

■ 注意

このマニュアルの一部または全部を無断で転写したり複写したりすることは、固くお断りいたします。

このマニュアルの内容を、改良のため予告なしに変更することがあります。

システムの構築やプログラムの作成などは、このマニュアルの記載内容をよく読み、書かれている指示や注意を十分理解してから行ってください。誤操作によって、システムの故障が発生することがあります。

このマニュアルは、必要なときすぐに参照できるよう、手近なところに保管してください。

このマニュアルの記載内容について疑問点または不明点がございましたら、最寄りの弊社営業または SE までお知らせください。

お客様の誤操作に起因する事故発生や損害につきましては、弊社では責任を負いかねますのでご了承ください。

弊社提供ソフトウェアを改変して使用した場合に発生した事故や損害につきましては、弊社では責任を負いかねますのでご了承ください。

弊社提供以外のソフトウェアを使用した場合の信頼性については、弊社では責任を負いかねますのでご了承ください。

ファイルのバックアップ作業を日常業務に組み入れてください。ファイル装置の障害、ファイルアクセス中の停電、誤操作、その他何らかの原因によってファイルの内容を消失することがあります。このような事態に備え、計画的にファイルのバックアップを取っておいてください。

■ 発行

2021年3月（第1版）IOT-7-0010

■ 著作権

All Rights Reserved. Copyright (C) 2021, Hitachi, Ltd.

はじめに

このマニュアルは、組込み装置向けデータ管理基盤（NX Context-based Data Management System for Embedded device、以降 NX CDMS Embedded と略す）の機能や操作方法について説明したものです。

■ 対象読者

次に示す方を対象としています。

- NX CDMS Embedded を構築・運用する方
- NX CDMS Embedded を利用して、現場データとは独立して、現場データの拡張およびアプリケーションの拡張などを実施する方

■ マニュアルの構成

このマニュアルは、次に示す章と付録から構成されています。

第1章 概説

NX CDMS Embedded の概要について説明しています。

第2章 NX CDMS Embedded の機能

NX CDMS Embedded の機能について説明しています。

第3章 コマンド一覧

NX CDMS Embedded で提供する各種コマンドについて説明しています。

第4章 保存領域操作 API

保存領域に蓄積したデータの参照や削除を行う API について説明しています。

第5章 定義ファイル

NX CDMS Embedded で使用する定義ファイルについて説明しています。

第6章 Windows イベントログ

Windows イベントログについて説明しています。

第7章 保存領域操作 API を使ったアプリケーション作成

保存領域操作 API を使ったアプリケーションの作成について説明しています。

第8章 HX-DSM との連携

HX-DSM との連携について説明しています。

付録 A 用語解説

このマニュアル内で使用する用語について説明しています。

■ このマニュアルで使用する書式

このマニュアルで使用する書式を説明します。

書式	説明
文字列	可変の値を示します。 (例) 日付は <code>yyyyMMdd</code> の形式で指定します。
<code>NX_CDMS_EMB_DIR</code>	NX CDMS Embedded のインストールパスを示します。

書式	説明
	環境変数 NX_CDMS_EMB_DIR に設定されます。
[]	ウインドウ、ダイアログボックス、メニュー、ボタンなどの画面上の要素名を示します。
<>	キーボードのキーを示します。

■ このマニュアルで使用する KiB（キビバイト）などの単位表記

1KiB（キビバイト）、1MiB（メビバイト）、1GiB（ギビバイト）、1TiB（テビバイト）はそれぞれ 1,024 バイト、 $1,024^2$ バイト、 $1,024^3$ バイト、 $1,024^4$ バイトです。

目次

1	概説	1
1.1	概要	2
1.1.1	背景	2
1.1.2	NX CDMS Embedded とは	2
1.1.3	NX CDMS Embedded の時刻の取り扱い	5
1.2	システム要件	6
1.2.1	ハードウェア要件	6
1.2.2	ソフトウェア要件	6
1.3	構築から運用まで	7
1.3.1	概略フロー	7
1.3.2	初期構築フロー	9
1.3.3	障害対策フロー	13
1.3.4	定義変更フロー	15
1.3.5	プログラムプロダクト更新フロー	16
1.3.6	前提ソフトウェアの更新	18
2	NX CDMS Embedded の機能	23
2.1	サポート機能一覧	24
2.2	起動停止機能	26
2.3	データ収集蓄積機能	27
2.4	データ項目紐付け機能	32
2.5	保存領域操作機能	33
2.6	データ名称定義機能	34
2.7	RAS 機能	35
2.8	情報/状態表示機能	36
2.9	インストーラ/アンインストーラ機能	37
2.9.1	インストーラ	37
2.9.2	アンインストーラ	38
3	コマンド一覧	39
3.1	提供コマンド一覧	40
3.2	起動コマンド	41
3.2.1	コマンド形式	41
3.2.2	コマンドの説明	41
3.2.3	前提条件	41
3.2.4	戻り値	41

3.2.5	メッセージ一覧	41
3.3	停止コマンド	43
3.3.1	コマンド形式	43
3.3.2	コマンドの説明	43
3.3.3	前提条件	43
3.3.4	戻り値	43
3.3.5	メッセージ一覧	43
3.4	RAS コマンド	45
3.4.1	コマンド形式	45
3.4.2	コマンドの説明	45
3.4.3	前提条件	46
3.4.4	戻り値	46
3.4.5	メッセージ一覧	47
3.5	プログラムプロダクト情報表示コマンド	49
3.5.1	コマンド形式	49
3.5.2	コマンドの説明	49
3.5.3	前提条件	49
3.5.4	戻り値	49
3.5.5	メッセージ一覧	49
3.6	動作状態表示コマンド	51
3.6.1	コマンド形式	51
3.6.2	コマンドの説明	51
3.6.3	前提条件	51
3.6.4	戻り値	51
3.6.5	メッセージ一覧	51
3.7	DB セットアップコマンド	53
3.7.1	コマンド形式	53
3.7.2	コマンドの説明	53
3.7.3	前提条件	54
3.7.4	戻り値	54
3.7.5	メッセージ一覧	54
3.8	DB VACUUM コマンド	56
3.8.1	コマンド形式	56
3.8.2	コマンドの説明	56
3.8.3	前提条件	56
3.8.4	戻り値	56
3.8.5	メッセージ一覧	56
4	保存領域操作 API	59
4.1	保存領域操作 API 一覧	60

4.2	初期化 API	61
4.2.1	API の説明	61
4.2.2	インターフェイス	61
4.2.3	戻り値	61
4.3	終了処理 API	62
4.3.1	API の説明	62
4.3.2	インターフェイス	62
4.3.3	戻り値	62
4.4	定常監視データ読込 API	63
4.4.1	API の説明	63
4.4.2	インターフェイス	63
4.4.3	戻り値	64
4.5	ロググループ読込 API	65
4.5.1	API の説明	65
4.5.2	インターフェイス	65
4.5.3	戻り値	66
4.6	ロググループ削除 API	67
4.6.1	API の説明	67
4.6.2	インターフェイス	67
4.6.3	戻り値	67
4.7	ロググループ容量取得 API	69
4.7.1	API の説明	69
4.7.2	インターフェイス	69
4.7.3	戻り値	70
4.8	リターンコード一覧	71
5	定義ファイル	73
5.1	定義ファイル一覧	74
5.2	定義ファイルにおける注意事項	76
5.3	定義ファイルの共通仕様	77
5.4	定義ファイルの配置場所	78
5.5	データ用リングバッファ定義ファイル	79
5.5.1	ファイルの説明	79
5.5.2	ファイル名称	79
5.5.3	定義記述方法	79
5.6	データ項目紐付け定義ファイル	81
5.6.1	ファイルの説明	81
5.6.2	ファイル名称	81
5.6.3	定義記述方法	81
5.7	データ名称定義ファイル	83

5.7.1	ファイルの説明	83
5.7.2	ファイル名称	83
5.7.3	定義記述方法	83
5.8	トリガー用リングバッファ定義ファイル	85
5.8.1	ファイルの説明	85
5.8.2	ファイル名称	85
5.8.3	定義記述方法	85
5.9	イベントトリガー定義ファイル	88
5.9.1	ファイルの説明	88
5.9.2	ファイル名称	88
5.9.3	定義記述方法	88
5.10	DB 接続設定ファイル	90
5.10.1	ファイルの説明	90
5.10.2	ファイル名称	90
5.10.3	定義記述方法	90
6	Windows イベントログ	93
6.1	Windows イベントログ一覧	94
7	保存領域操作 API を使ったアプリケーション作成	99
7.1	インクルードファイル	100
7.2	DLL (Dynamic Link Library)	101
7.3	作成環境	102
7.4	保存領域操作 API 使用時の注意事項	103
8	HX-DSM との連携	105
8.1	データ用リングバッファの注意事項	106
8.1.1	データ用リングバッファへの書き込み	106
8.2	トリガー用リングバッファの注意事項	107
8.2.1	トリガー用リングバッファへの書き込み	107
8.2.2	トリガー用リングバッファからの読み込み	107
付録		109
付録 A	用語解説	110
索引		111

1

概説

この章では、NX CDMS Embedded の概要について説明します。

1.1 概要

ここでは、NX CDMS Embedded の概要について説明します。

1.1.1 背景

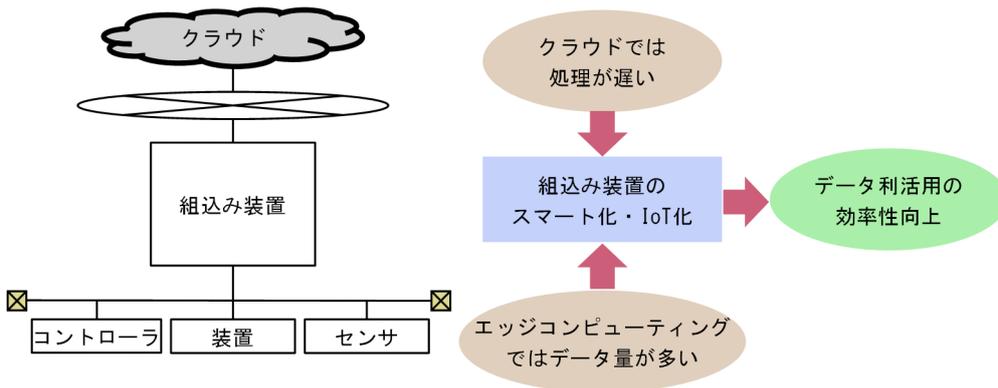
近年、競争が激化する工作機械業界で、その競争軸が従来の多機能化・機械性能向上から、故障診断やパラメータチューニングといった工作機械由来のデータを活用した新たなサービスにシフトしています。これら新サービスを実現するためには、リアルタイムでのデータの高速度処理が不可欠です。しかし、クラウドを利用した場合は処理スピードの遅さが課題となり、エッジコンピューティングの場合ではデータ量の多さが課題です。

1.1.2 NX CDMS Embedded とは

組み込み装置上で時系列データを収集し、蓄積することで、「組み込み装置のスマート化・IoT化を実現し、データ利活用の効率性の向上」を可能にするプラットフォームです。

NX CDMS Embedded が解決する課題を次の図に示します。

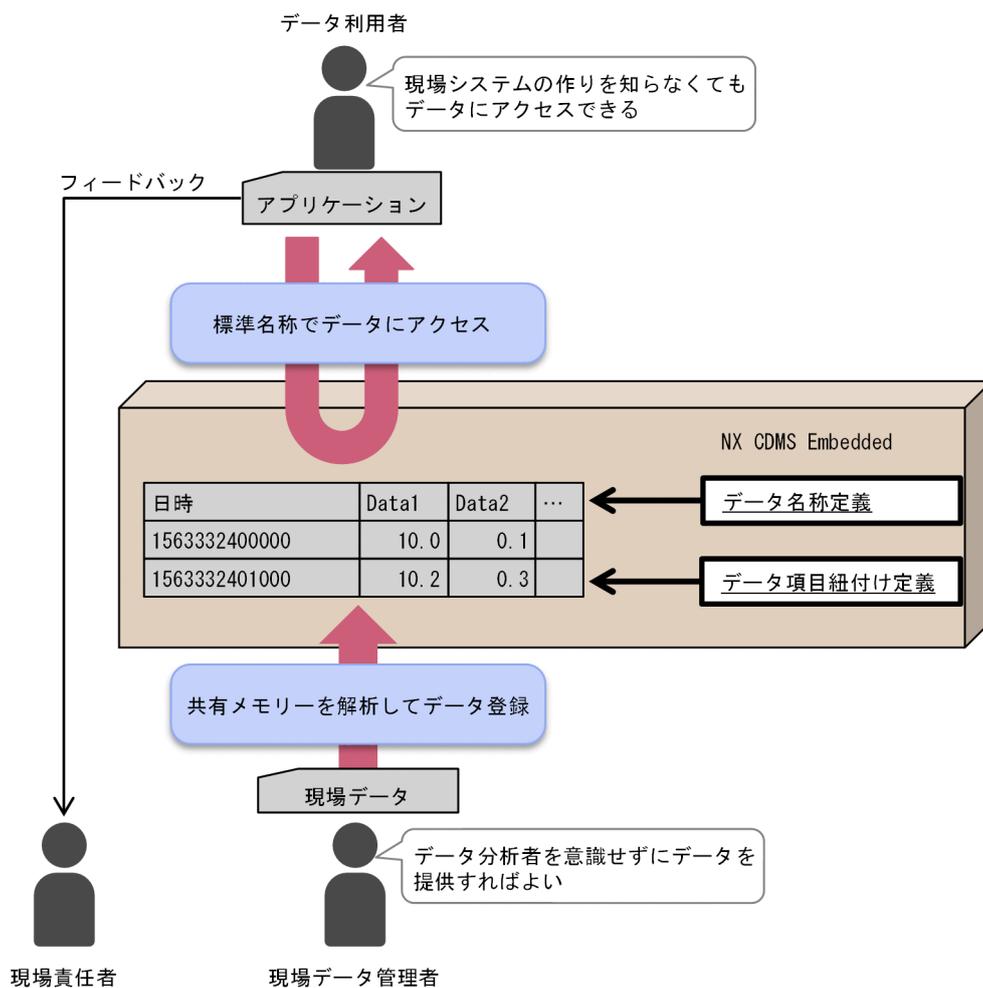
図 1-1 NX CDMS Embedded が解決する課題



NX CDMS Embedded では、現場データと独立で、現場データの拡張、アプリケーションの拡張を容易にします。

NX CDMS Embedded のメリットを次の図に示します。

図 1-2 NX CDMS Embedded のメリット



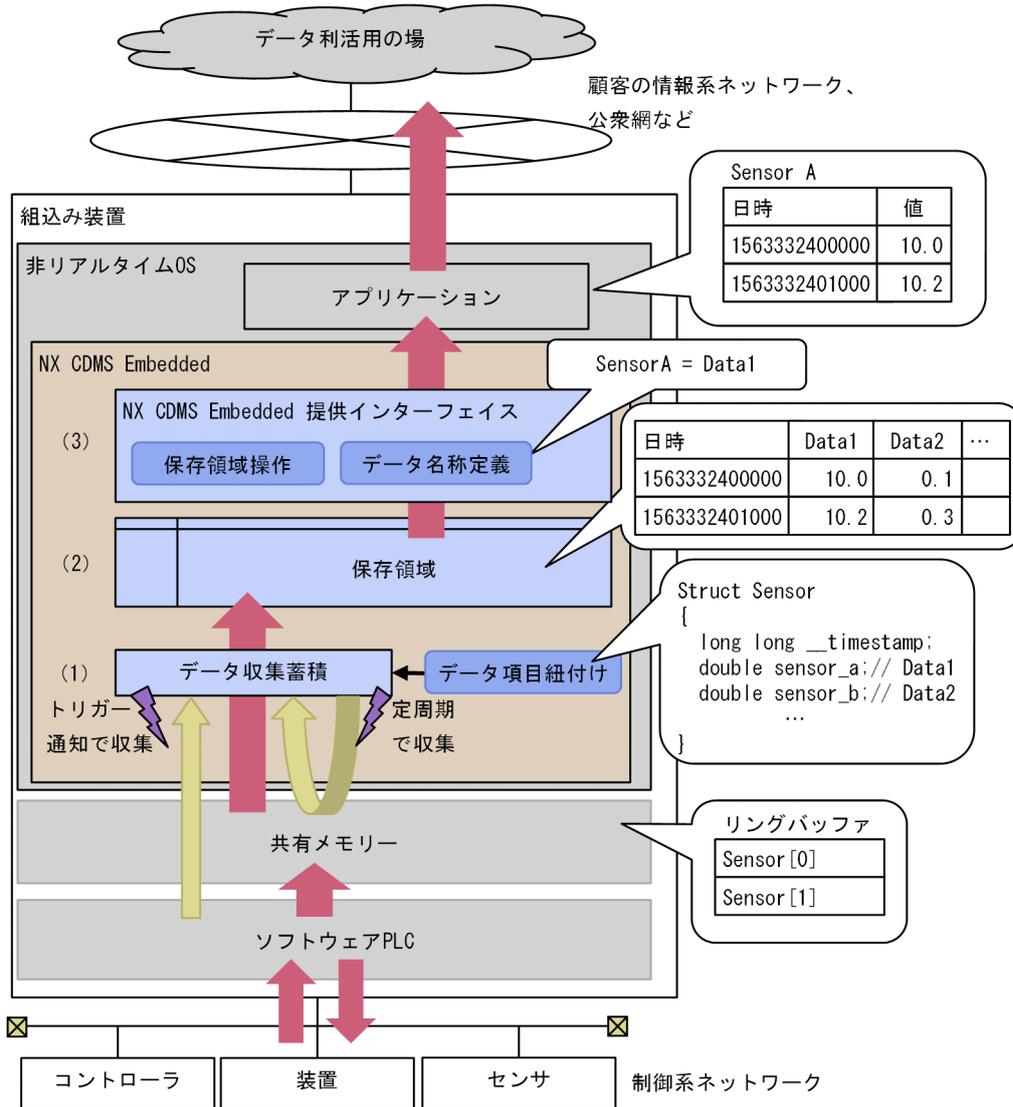
(凡例)

 : データの流れ

- 現場データ管理者は、現場の時系列データのフォーマットを定義するだけで、現場のデータを提供できます。
- データ利用者は、必要なデータ項目と標準名称を定義することで、必要なデータを入手でき、データをより使いやすくするとともに、データを活用したアプリケーションの横展開を容易にします。

NX CDMS Embedded が実現するシステム全体構成の例を次の図に示します。

図 1-3 NX CDMS Embedded が実現するシステム全体構成の例



(凡例)

- ➡ : データの流れ
- ➡ : 収集の流れ

(1) 共有メモリー上に書き込まれた時系列データを、定周期、またはトリガー通知によって収集します

トリガーにはあらかじめ、蓄積対象のデータ、サンプリング周期、取得期間を定義できます。

トリガーの通知は共有メモリーを介して行われ、トリガーが通知された場合、NX CDMS Embedded はトリガーの定義に基づいて時系列データを保存領域に蓄積します。なお、トリガーの種類には、共有メモリー上の時系列データをすべて保存領域に蓄積するエラートリガーとユーザーが任意に選択した時系列データを保存領域に蓄積するイベントトリガーの 2 種類があります。

(2) 保存領域には、定周期による収集、エラートリガーによる収集、イベントトリガーによる収集の3種類の領域があります

それぞれの領域に保存できるサイズの上限に達した場合、定周期による収集では、最古に登録したデータから順に上書きします。

保存できるサイズの上限については、「2.3 データ収集蓄積機能」の、保存領域に保存できるサイズの上限についての表を参照してください。

エラートリガーとイベントトリガーによる収集では、データの削除が実施されるまで保存は行なわれません。

NX CDMS Embedded が提供するインターフェイスによって、適宜、削除を実施してください。

(3) 保存領域に蓄積されたデータは、NX CDMS Embedded が提供するインターフェイスによって取得できます

保存領域上では各データは Data1、Data2…のように番号で管理するため、各番号に対し別名を定義することで、アプリケーションはその名称でデータを取得できます。

1.1.3 NX CDMS Embedded の時刻の取り扱い

NX CDMS Embedded での時刻の取り扱い方針を次の表に示します。

表 1-1 時刻の取り扱い

項番	項目	詳細
1	Windows のイベントログ出力	ローカルタイム (Windows の仕様に準拠します)。
2	NX CDMS Embedded が収集し、保存領域に格納するデータ内の時刻	UTC (1970/01/01 00:00:00.000 からのミリ秒で表記します)。
3	NX CDMS Embedded が提供するインターフェイスによって出力するデータ内の時刻	UTC (1970/01/01 00:00:00.000 からのミリ秒で表記します)。

1.2 システム要件

ここでは、NX CDMS Embedded のシステム要件について説明します。

1.2.1 ハードウェア要件

NX CDMS Embedded が前提とするハードウェアを次の表に示します。

表 1-2 前提ハードウェア

項番	分類	ハードウェア要件
1	アーキテクチャ	x64
2	CPU	クロック周波数：3.4 GHz 以上、コア数：2 コア以上*
3	RAM	2GB 以上*
4	SSD	10GB 以上*

注※

OS などの前提ソフトウェア、および共存させる他のソフトウェアのリソースは別に確保する必要があります。

1.2.2 ソフトウェア要件

NX CDMS Embedded が前提とするソフトウェアを次の表に示します。

表 1-3 前提ソフトウェア

項番	プログラム プロダクト	前提ソフトウェア
1	NX CDMS Embedded (型式：S-741E-10P)	<ul style="list-style-type: none"> • Microsoft(R) Windows(R) 10 IoT Enterprise 2019 LTSC (64bit) • RT-DSM (型式：S-763A-96P) * • PostgreSQL 11.10

注※

NX CDMS Embedded では、RT-DSM (型式：S-763A-96P) を HX-DSM と呼称します。
組込み装置に同梱のマニュアル等には、RT-DSM と記載されていますので、ご注意ください。

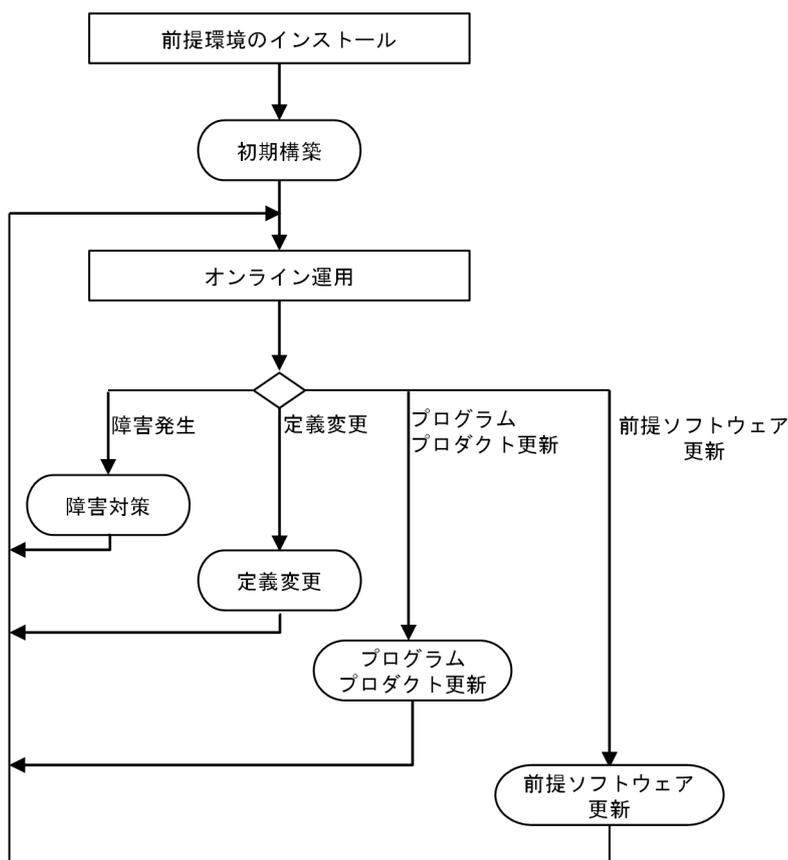
1.3 構築から運用まで

ここでは、NX CDMS Embedded の初期構築から運用まで、および保守に関する作業の概略について説明します。

1.3.1 概略フロー

初期構築から運用までの概略フローを次の図に示します。

図 1-4 構築から運用までの概略フロー



(1) 前提環境のインストール

最初に前提環境をインストールします。「1.2 システム要件」を参照し、前提となるハードウェア、ソフトウェアを準備し、各ハードウェア、ソフトウェアのマニュアルを参照してインストールしてください。

(a) PostgreSQL のインストール

次の手順で前提ソフトウェアのインストールおよび設定をします。

1. インストーラ [postgresql-11 *n1*-*n2*-windows-x64.exe] をダブルクリックします。Disk2 のメディアを使用してください。
n1、*n2* はマイナーバージョンを表します。
2. 表示された [Setup] 画面で、[Next] ボタンをクリックします。

- 3.表示された [Installation Directory] 画面で、[Installation Directory] にインストールするディレクトリを指定し、[Next] ボタンをクリックします。
- 4.表示された [Select Components] 画面で、[pgAdmin] と [Stack Builder] のチェックボックスにあるチェックを外し、[Next] ボタンをクリックします。
- 5.表示された [Data Directory] 画面で、[Data Directory] にデータを保存するディレクトリを指定し、[Next] ボタンをクリックします。
- 6.表示された [Password] 画面で、スーパーユーザー (postgres) のパスワードを指定します。
[Password] と [Retype password] に任意のパスワードを入力し、[Next] ボタンをクリックします。
- 7.表示された [Port] 画面で、[Port] に使用するポート番号を指定し、[Next] ボタンをクリックします。
- 8.表示された [Advanced Options] 画面で、[Locale] に「C」を指定し、[Next] ボタンをクリックします。
- 9.表示された [Pre Installation Summary] 画面で、[Next] ボタンをクリックします。
- 10.表示された [Ready to Install] 画面で、[Next] ボタンをクリックします。
- 11.表示された [Completing the PostgreSQL Setup Wizard] 画面で、[Finish] ボタンをクリックし、インストールを終了します。
- 12.環境変数の [PATH] に以下に示すパスを追加します。
[PostgreSQL のインストール先ディレクトリ] %bin

(b) PostgreSQL の設定

次の手順で PostgreSQL を設定します。

- 1.Windows の [スタート] ボタンをクリックし、スタートメニューの [Windows 管理ツール] - [サービス] をクリックします。
- 2.表示された [サービス] 画面で、[postgresql-x64-11] サービスの [状態] を確認します。
[実行中] の場合、[postgresql-x64-11] サービスを右クリックし、[停止] をクリックします。
[停止] の場合、次の手順に進んでください。
- 3.以下のフォルダに移動し、「postgresql.conf」ファイルを開きます。
フォルダ：「(a) PostgreSQL のインストール」の手順 5 で指定したデータ保存先ディレクトリ
- 4.以下の項目の設定を変更し、ファイルを保存します。

表 1-4 PostgreSQL の設定項目

設定項目	設定値
listen_addresses	'localhost'
max_connections	6
max_wal_senders	0
max_locks_per_transaction	400

- 5.Windows の [スタート] ボタンをクリックし、スタートメニューの [Windows 管理ツール] - [サービス] をクリックします。

6. 表示された [サービス] 画面で [postgresql-x64-11] サービスを右クリックし、[開始] をクリックします。

PostgreSQL が起動します。

(2) 初期構築

NX CDMS Embedded の初期構築を行います。

詳細については、「1.3.2 初期構築フロー」を参照してください。

(3) オンライン運用

NX CDMS Embedded は、動作定義に従ってデータの収集・蓄積を行います。

定常時はこの状態でシステムの運用を続けてください。

障害が発生した場合、システム増改築に合わせ NX CDMS Embedded の動作定義を変更する場合、プログラムプロダクトのバージョンアップのためにプログラムプロダクトを更新する場合は、それぞれの手順を実施してください。

(4) 障害対策

NX CDMS Embedded で障害が発生した場合は、障害の調査・対策を実施します。

詳細については、「1.3.3 障害対策フロー」を参照してください。

(5) 定義変更

NX CDMS Embedded の動作定義を変更する場合、動作定義を更新し再起動することで定義変更ができます。

詳細については、「1.3.4 定義変更フロー」を参照してください。

(6) プログラムプロダクト更新

プログラムプロダクトのバージョンアップをする場合は、「1.3.5 プログラムプロダクト更新フロー」の手順を実施してください。

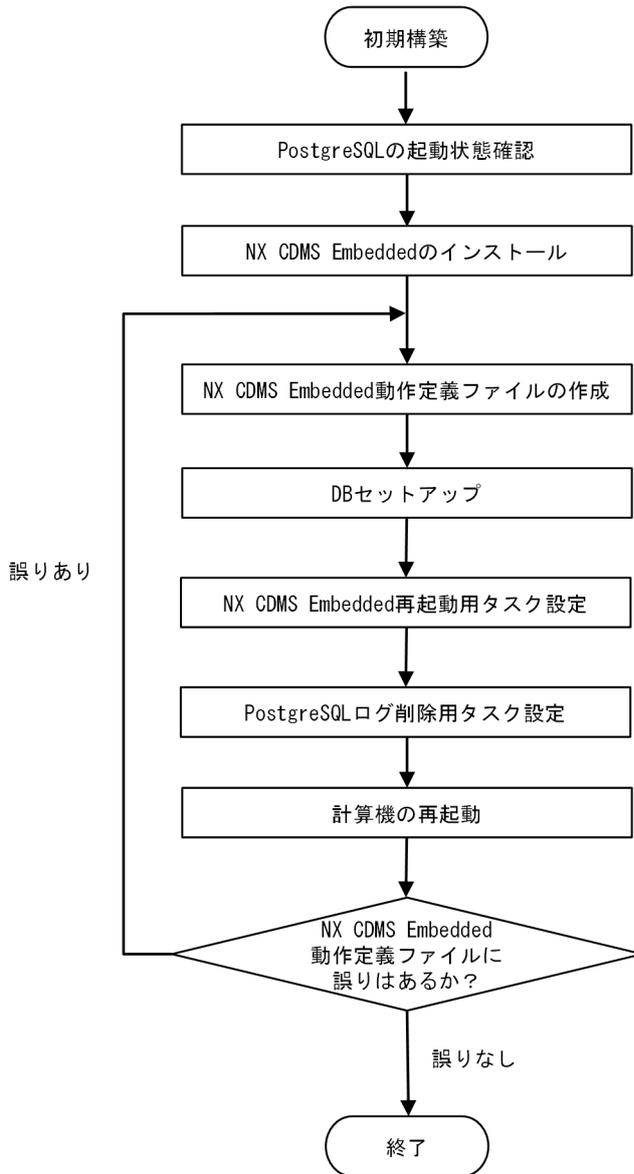
(7) 前提ソフトウェアの更新

前提ソフトウェアを更新する場合は、「1.3.6 前提ソフトウェアの更新」の手順を実施します。

1.3.2 初期構築フロー

NX CDMS Embedded のプログラムプロダクトをインストール後、最初の実施する初期構築作業の概略フローを次の図に示します。

図 1-5 初期構築の概略フロー



(1) PostgreSQL の起動状態確認

PostgreSQL が起動していることを確認します。

1. Windows の [スタート] ボタンをクリックし、スタートメニューの [Windows 管理ツール] - [サービス] をクリックします。
2. 表示された [サービス] 画面で、[postgresql-x64-11] サービスの [状態] が [実行中] と表示されていることを確認します。
[状態] が何も表示されていない場合、PostgreSQL は停止中です。以下の手順で起動してください。
[サービス] 画面で、[postgresql-x64-11] サービスを右クリックし、[開始] を選択します。

(2) NX CDMS Embedded のインストール

NX CDMS Embedded をインストールする手順を次に示します。

1. NX CDMS Embedded のインストール CD の [setup.exe] をダブルクリックします。Disk1 のメディアを使用してください。
2. 表示されたインストール開始画面で、[次へ] ボタンをクリックします。
3. 表示された [インストール先の選択] ダイアログボックスで [参照] ボタンをクリックし、インストール先を選択します。
4. 表示された [次へ] ボタンをクリックします。
5. [完了] ボタンをクリックします。
6. 次のプログラムプロダクト情報表示コマンドを実行し、インストールの成功を確認します。

```
cdms_emb_info.bat
```

プログラムプロダクト情報表示コマンドの詳細については、「3.5 プログラムプロダクト情報表示コマンド」を参照してください。

7. 表示された Windows の [スタート] ボタンを右クリックし、[ファイル名を指定して実行] をクリックします。
8. 表示された [ファイル名を指定して実行] ダイアログボックスで、実行するプログラム名称に [gpedit.msc] と入力して、[OK] ボタンをクリックします。
9. 表示された [ローカル グループ ポリシー エディター] 画面で、コンソールツリーの [コンピューターの構成] - [Windows の設定] - [スクリプト (スタートアップ/シャットダウン)] をクリックします。
10. 右側の結果ウィンドウに表示される [シャットダウン] をダブルクリックします。
11. 表示された [シャットダウンのプロパティ] ダイアログボックスで、[追加] ボタンをクリックします。
12. [スクリプトの追加] ダイアログボックスの [参照] ボタンをクリックします。
13. NX CDMS Embedded のインストールディレクトリの bin フォルダ下にある cdms_emb_stop.bat ファイルを選択し、[OK] ボタンをクリックします。
14. [シャットダウンのプロパティ] ダイアログボックスで、[適用] ボタンをクリックし、[OK] ボタンをクリックします。

(3) NX CDMS Embedded 動作定義ファイルの作成

NX CDMS Embedded の動作定義ファイルを作成します。

作成する定義ファイルについては、「5.1 定義ファイル一覧」を参照してください。

DB 接続設定ファイルに定義するユーザー名、パスワード、およびデータベース名称については、「(4) DB セットアップ」で定義に従って DB セットアップコマンドがユーザー、およびデータベースを作成します。

このため、作成したいユーザー名、パスワード、およびデータベース名称を定義してください。

(4) DB セットアップ

NX CDMS Embedded が収集したデータの格納用にデータベースをセットアップします。

「5.10 DB 接続設定ファイル」の定義に従って、ユーザー作成、データベース作成・設定をします。

データベースをセットアップする手順を次に示します。PostgreSQL が起動している状態でセットアップを実施してください。

1. 次の DB セットアップコマンドを実行します。

cdms_emb_dbsetup.exe

DB セットアップコマンドの詳細は、「3.7 DB セットアップコマンド」を参照してください。

DB セットアップコマンド起動時に、ユーザー名、パスワードを入力します。

表 1-5 DB セットアップコマンド起動時に入力するユーザー名とパスワード

項番	設定項目	設定内容
1	ユーザー名	postgres
2	パスワード	スーパーユーザーのパスワード (PostgreSQL インストール時に設定したパスワード)

データベース接続設定に誤りがある場合は、エラーを出力しますので、DB 接続設定ファイルの誤りを訂正してください。

DB セットアップコマンドを実施したあとに再度実行した場合は、何もする必要はありません。

(5) NX CDMS Embedded 再起動用タスク設定

1 週間に 1 回、NX CDMS Embedded のサービスを再起動するタスクをタスクスケジューラに設定します。

再起動を実施する曜日および時間については、運用に合わせて調整してください。

NX CDMS Embedded のサービスの再起動中はデータが欠損します。データの欠損を考慮して、タスクスケジューラのタイミングを調整してください。

再起動タスクをタスクスケジューラに設定する手順を次に示します。

1. Windows の [スタート] ボタンをクリックし、スタートメニューの [Windows 管理ツール] - [タスクスケジューラ] をクリックします。
2. [タスクスケジューラ] 画面で、メニューの [操作] - [タスクのインポート] をクリックします。
3. [開く] 画面で、NX CDMS Embedded のインストールディレクトリの etc フォルダ下の cdms_emb_restart_task.xml ファイルを選択し、[開く] ボタンをクリックします。
4. [タスクの作成] 画面で、[トリガー] タブをクリックします。
5. [トリガー] タブで [毎週] を選択し、[編集] ボタンをクリックします。
6. [トリガーの編集] ダイアログの [週間ごとの次の曜日] および [開始] の時間を運用に合わせて入力し、[OK] ボタンをクリックします。
7. [タスクの作成] 画面で、[OK] ボタンをクリックします。

<定期的な NX CDMS Embedded の再起動が必要な理由>

データベースのレコード追加や削除を繰り返すとデータベースにごみがたまり、ディスク使用量増加やスローダウンにつながります。

上記現象を防止するため、本製品では NX CDMS Embedded のサービス起動時にデータベース整理 (REINDEX) を実行する処理を組み込んでいます。

このため、タスクスケジューラで定期的に再起動を実行してください。

ただし、NX CDMS Embedded のサービス以外の再起動は不要です。

<保存領域操作 API を使用するアプリケーションを常駐させる場合の注意事項>

保存領域操作 API を使用するアプリケーションを常駐させる場合は、「7.4 保存領域操作 API 使用時の注意事項」の表 7-1 の項番 4 を参照してください。

(6) PostgreSQL ログ削除用タスク設定

PostgreSQL のログが 30 日分を超えた場合に、古いログを削除するタスクをタスクスケジューラに設定します。

削除用タスクをタスクスケジューラに設定する手順を、次に示します。

1. Windows の [スタート] ボタンをクリックします。
2. スタートメニューの [Windows 管理ツール] - [タスクスケジューラ] をクリックします。
3. 表示された [タスクスケジューラ] 画面で、メニューの [操作] - [タスクのインポート] をクリックします。
4. 表示された [開く] 画面で、NX CDMS Embedded のインストールディレクトリの etc フォルダに格納されている `cdms_emb_del_log_task.xml` ファイルを選択します。
5. [開く] ボタンをクリックします。
6. 表示された [タスクの作成] 画面で、[操作] タブをクリックします。
7. [操作] タブで [プログラムの開始] を選択し、[編集] ボタンをクリックします。
8. [操作の編集] ダイアログの [引数の追加 (オプション)] の文字列の `[forfiles /p "[PostgreSQL データディレクトリ]¥log" /d -30 /m postgresql-*.log /c "cmd /c del @file"]` 中の `[PostgreSQL データディレクトリ]` の部分を以下に置き換えます。
PostgreSQL のインストール時に設定したデータディレクトリまでのフルパス ([(a) PostgreSQL のインストール] の手順 5 で設定)
デフォルト設定でインストールした場合、`[C:¥Program Files¥PostgreSQL¥11¥data]` です。
9. [操作の編集] ダイアログの [OK] ボタンをクリックします。
10. [タスクの作成] 画面の [OK] ボタンをクリックします。

(7) 計算機再起動

定義に誤りがないことを確認した上で、計算機を再起動します。再起動後、NX CDMS Embedded が動作を開始します。

定義に誤りがある場合は、動作を停止します。エラーを出力しますので、定義ファイルの誤りを訂正してください。

NX CDMS Embedded の動作が開始できているかについては、次の動作状態表示コマンドを実行することで確認します。

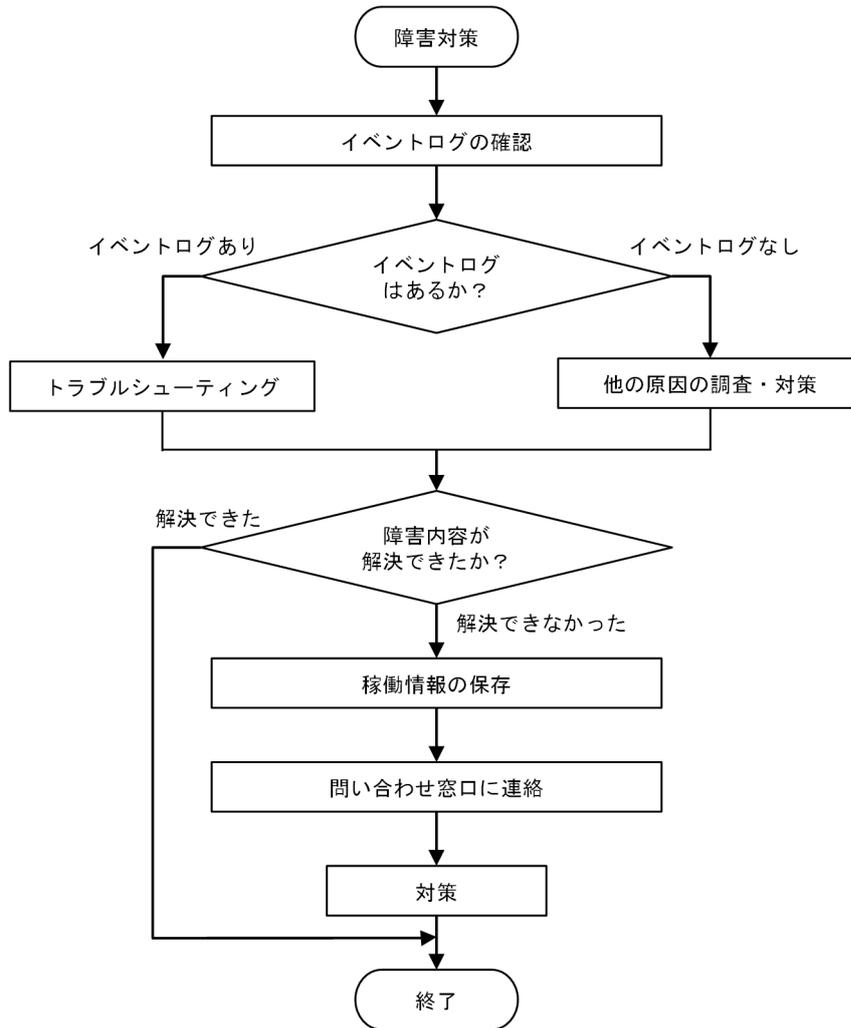
```
cdms_emb_stat.exe
```

動作状態表示コマンドの詳細については、「3.6 動作状態表示コマンド」を参照してください。

1.3.3 障害対策フロー

障害が発生した場合に行なう調査・対策・復旧作業の概略フローを次の図に示します。

図 1-6 障害対策の概略フロー



(1) イベントログの確認

NX CDMS Embedded の障害情報は、Windows イベントログに記録されます。

Windows イベントログを参照し、NX CDMS Embedded の障害情報を確認してください。

(2) トラブルシューティング

Windows イベントログに NX CDMS Embedded のイベントが記録されている場合は、「6 Windows イベントログ」を参照してください。そのイベントの意味とともに対処方法を記載していますので、その対処方法に従って対策してください。対策の結果、障害が取り除かれれば、障害対策は終了です。

(3) 他の原因の調査・対策

Windows イベントログにイベントが記録されていない場合、他の要因のおそれがあるため、他の原因を調査し、原因を取り除いてください。

(4) 稼働状態の保存、問い合わせ窓口に連絡、対策

次に示す場合、「2.7 RAS 機能」の手順に従い、NX CDMS Embedded の RAS ログを回収してください。

- 「6 Windows イベントログ」に記載の対策を実施しても障害が回復しない場合
- Windows イベントログに NX CDMS Embedded のイベントがなく、他の原因も見つからない場合

その後、問い合わせ窓口に連絡して指示に従い、対策してください。

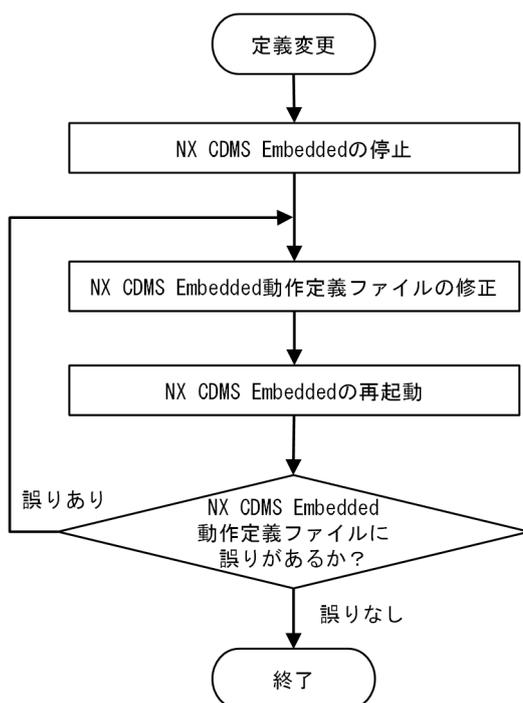
対策の結果、障害の原因が取り除かれれば、障害対策は終了です。

なお、システムを復旧するには計算機の再起動が必要です。ただし、障害の原因が取り除かれていない場合は、再度同じ障害が起きるおそれがあります。

1.3.4 定義変更フロー

システムの増改築やトラブルシューティングのために、運用中の NX CDMS Embedded の動作定義を変更する作業の概略フローを次の図に示します。

図 1-7 定義変更の概略フロー



(1) NX CDMS Embedded の停止

次の停止コマンドを実行し、NX CDMS Embedded を停止してください。

```
cdms_emb_stop.bat
```

停止コマンドの詳細については、「3.3 停止コマンド」を参照してください。

(2) NX CDMS Embedded 動作定義ファイルの修正

NX CDMS Embedded の定義ファイルを変更してください。

詳細については、「5 定義ファイル」を参照してください。

(3) NX CDMS Embedded の再起動

定義に誤りがないことを確認した上で、NX CDMS Embedded を再起動します。再起動後、NX CDMS Embedded が動作を開始します。

次の起動コマンドを実行し、NX CDMS Embedded を再起動してください。

```
cdms_emb_start.bat
```

開始コマンドの詳細については、「3.2 起動コマンド」を参照してください。

定義に誤りがある場合は、動作を停止します。出力されたエラーを確認して、定義ファイルの誤りを訂正してください。

NX CDMS Embedded の動作が開始できているかは、次の動作状態表示コマンドを実行し確認してください。

```
cdms_emb_stat.exe
```

動作状態表示コマンドの詳細については、「3.6 動作状態表示コマンド」を参照してください。

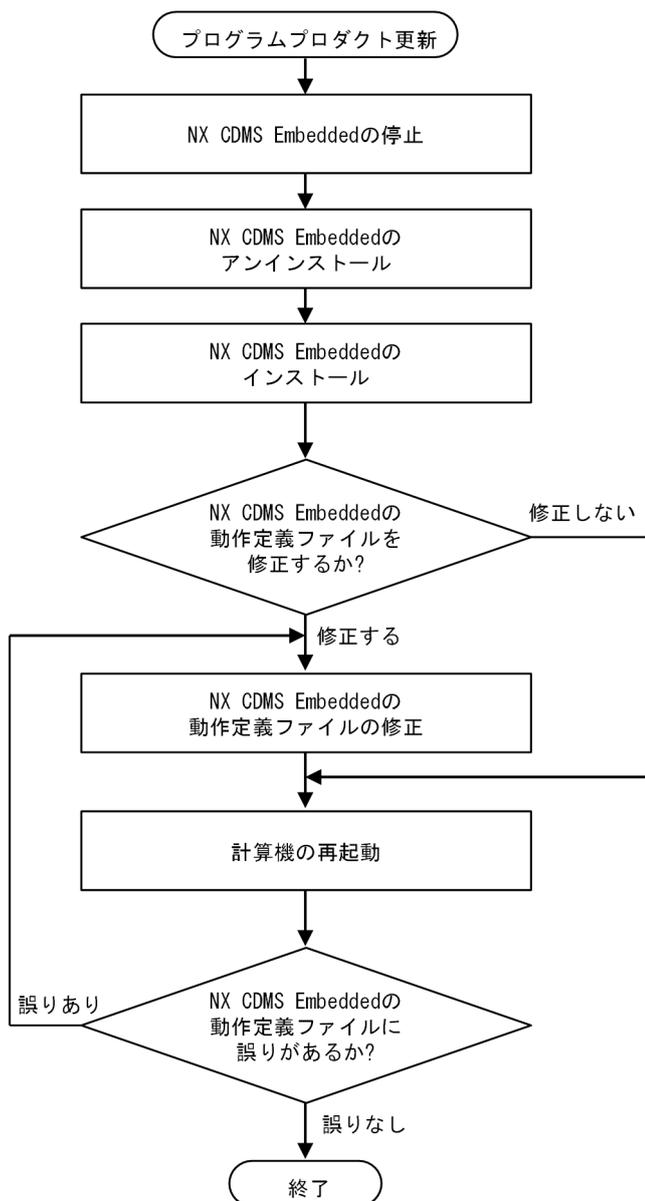
注

計算機を再起動することで、NX CDMS Embedded を再起動できます。

1.3.5 プログラムプロダクト更新フロー

運用中の NX CDMS Embedded に対し、バージョンアップなどのためにアンインストール、または再インストールする作業の概略フローを次の図に示します。

図 1-8 プログラムプロダクト更新の概略フロー



(1) NX CDMS Embedded の停止

次の停止コマンドを実行し、NX CDMS Embedded を停止してください。

```
cdms_emb_stop.bat
```

停止コマンドの詳細については、「3.3 停止コマンド」を参照してください。

(2) NX CDMS Embedded のアンインストール

NX CDMS Embedded をアンインストールする手順を次に示します。

1. Windows の [コントロール パネル] から「プログラムと機能」を選択します。
2. 表示された [プログラムと機能] 画面から「NX-CDMS-EMB」を選択します。

3. 右クリックして表示されるメニューから「アンインストール」を選択します。

アンインストールをするかどうかの確認ダイアログが表示された場合は、[はい] ボタンをクリックしてください。

NX CDMS Embedded がアンインストールされます。

(3) NX CDMS Embedded のインストール

NX CDMS Embedded をインストールする手順を次に示します。

1. NX CDMS Embedded のインストール CD の [setup.exe] をダブルクリックします。

[インストール開始] 画面が表示されます。

2. 表示された [インストール開始] 画面で、[次へ] ボタンをクリックします。

[インストール先の選択] ダイアログが表示されます。

3. 表示された [インストール先の選択] ダイアログで、[参照] ボタンをクリックしてインストール先を選択後、[次へ] ボタンをクリックします。

インストールが開始され、インストールが完了すると [インストール完了] 画面が表示されます。

4. 表示された [インストール完了] 画面で、[完了] ボタンをクリックします。

5. 次のプログラムプロダクト情報表示コマンドを実行し、インストールの成功を確認します。

```
cdms_emb_info.bat
```

プログラムプロダクト情報表示コマンドの詳細については、「3.5 プログラムプロダクト情報表示コマンド」を参照してください。

(4) NX CDMS Embedded の動作定義ファイルの修正

機能追加などによって定義項目が追加される場合があります。

追加した定義項目はデフォルト値で動作しますが、デフォルト値からの変更が必要な場合は、NX CDMS Embedded の動作定義ファイルを修正します。

修正する定義ファイルについては、「5 定義ファイル」を参照してください。

(5) 計算機再起動

定義に誤りがないことを確認した上で、計算機を再起動します。再起動後、NX CDMS Embedded が動作を開始します。

定義に誤りがある場合は、動作を停止します。出力されたエラーを確認して、定義ファイルの誤りを訂正してください。

NX CDMS Embedded の動作が開始できているかは、次の動作状態表示コマンドを実行し確認してください。

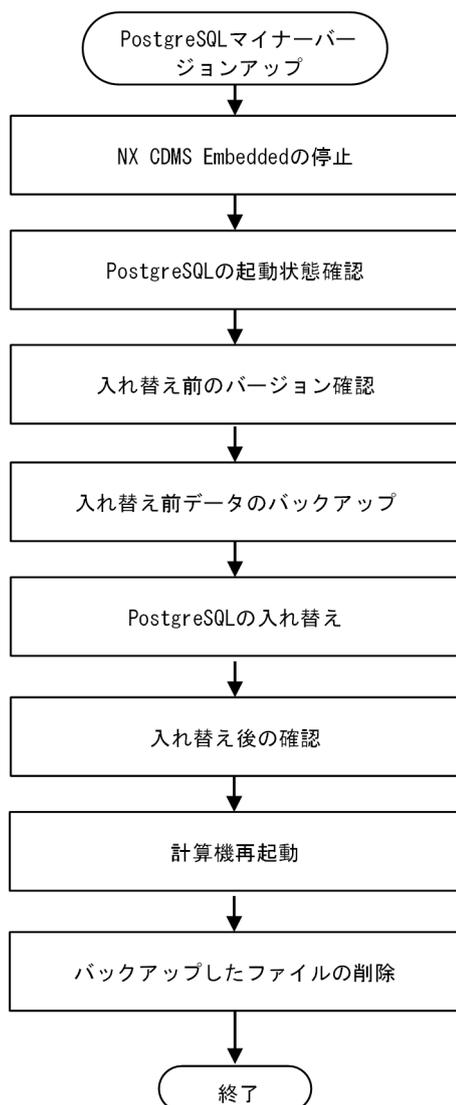
```
cdms_emb_stat.exe
```

動作状態表示コマンドの詳細については、「3.6 動作状態表示コマンド」を参照してください。

1.3.6 前提ソフトウェアの更新

運用中の NX CDMS Embedded に対し、PostgreSQL をマイナーバージョンアップする作業の概略フローを次に示します。

図 1-9 前提ソフトウェア更新の概略フロー



(1) NX CDMS Embedded の停止

次の停止コマンドを実行し、NX CDMS Embedded を停止してください。

```
cdms_emb_stop.bat
```

停止コマンドの詳細は、「3.3 停止コマンド」を参照してください。

(2) PostgreSQL の起動状態確認

PostgreSQL が起動していることを確認します。

1. Windows の [スタート] ボタンをクリックし、スタートメニューの [Windows 管理ツール] - [サービス] をクリックします。
2. [サービス] 画面で、[postgresql-x64-11] サービスの [状態] が [実行中] と表示されていることを確認します。
[状態] が何も表示されていない場合、PostgreSQL が停止しています。

3. PostgreSQL が停止している場合は、[サービス] 画面で [postgresql-x64-11] サービスを右クリックして、[開始] を選択します。

(3) 入れ替え前のバージョン確認

PostgreSQL のバージョンを確認します。

1. 次のコマンドを実行し、バージョンを確認してください。

```
psql -V
```

2. 結果が次のとおりに表示されることを確認してください。

```
psql(PostgreSQL) [入れ替え前のPostgreSQLのバージョン番号]
```

(4) 入れ替え前のデータのバックアップ

次の手順で PostgreSQL のデータベースのデータをバックアップします。

1. 次のコマンドを実行し、データベースのデータをバックアップしてください。

```
set PGUSER=postgres
set PGPASSWORD= [パスワード]
pg_dumpall -c -g > [バックアップ用ディレクトリ] %globals.backup
```

[パスワード] は、スーパーユーザーのパスワードです。

[バックアップ用ディレクトリ] は、データベースのデータのバックアップを格納するための任意のディレクトリです。

2. 次のコマンドを実行し、0 が表示されることを確認します。

```
echo %errorlevel%
```

3. 次のコマンドを実行し、データベースのデータをバックアップします。

```
pg_dump -Fc [データベース名] > [バックアップ用ディレクトリ] %cdms_emb_db.backup
```

[バックアップ用ディレクトリ] は、データベースのデータのバックアップを格納するための任意のディレクトリです。

[データベース名] は、DB 接続設定ファイルに定義しているデータベース名です。DB 接続設定ファイルの詳細については、「5.10 DB 接続設定ファイル」を参照してください。

4. 次のコマンドを実行し、0 が表示されることを確認します。

```
echo %errorlevel%
```

(5) PostgreSQL の入れ替え

PostgreSQL を新しいバージョンに入れ替えます。

新しいバージョンに入れ替える手順を次に示します。

1. Windows の [スタート] ボタンをクリックし、スタートメニューの [Windows 管理ツール] - [サービス] をクリックします。

2. 表示された [サービス] 画面で、[postgresql-x64-11] サービスの [状態] が [実行中] と表示されていることを確認します。

[状態] が [実行中] の場合、[postgresql-x64-11] サービスを右クリックし、[停止] を選択します。

[状態] が何も表示されていない場合は、手順 3 に進んでください。

3. 新しいバージョンの PostgreSQL をインストールします。
 インストーラ [postgresql-11.n1.n2-windows-x64.exe] をダブルクリックしてください。n1、n2 はマイナーバージョンです。
4. 表示された [Setup] 画面で、[Next] ボタンをクリックします。
5. 表示された [Select Components] 画面で、[pgAdmin] と [Stack Builder] のチェックボックスにあるチェックを外し、[Next] ボタンをクリックします。
6. 表示された [Existing Installation] 画面で、[Next] ボタンをクリックします。
7. 表示された [Existing Data Directory] 画面で、[Next] ボタンをクリックします。
8. 表示された [Pre Installation Summary] 画面で、[Next] ボタンをクリックします。
9. 表示された [Ready to Install] 画面で、[Next] ボタンをクリックします。
10. 表示された [Completing the PostgreSQL Setup Wizard] 画面で、[Finish] ボタンをクリックし、インストールを終了します。
11. 計算機を再起動します。

(6) 入れ替え後の確認

入れ替え後にデータが参照可能か確認します。

1. 次のコマンドを実行します。

```
psql -V
```

2. 結果が次のように表示されることを確認します。

```
psql(PostgreSQL) [新しいPostgreSQLのバージョン番号]
```

3. 次のコマンドを実行します。

```
set PGUSER=postgres
set PGPASSWORD=[パスワード]
```

```
psql -d [データベース名] -c "(SELECT 't_trg_mgt' as name, count(*) FROM cdms_emb_db.t_trg_mgt) UNION
(SELECT 't_err_trg_data_status_of_use_mgt' as name, count(*) FROM cdms_emb_db.t_err_trg_data_status_of_use_mgt) UNION
(SELECT 't_event_trg_data_status_of_use_mgt' as name, count(*) FROM cdms_emb_db.t_event_trg_data_status_of_use_mgt)"
```

[パスワード] は、スーパーユーザーのパスワードです。

[データベース名] は、DB 接続設定ファイルに定義しているデータベース名です。DB 接続設定ファイルの詳細については「5.10 DB 接続設定ファイル」を参照してください。

4. 結果が次のように表示されることを確認します。

name	count
t_trg_mgt	4
t_err_trg_data_status_of_use_mgt	31
t_event_trg_data_status_of_use_mgt	10000
(3 行)	

(7) 計算機再起動

定義に誤りがないことを確認した上で、計算機を再起動します。再起動後、NX CDMS Embedded が動作を開始します。

定義に誤りがある場合は、動作を停止します。エラーを出力しますので、定義ファイルの誤りを訂正してください。

NX CDMS Embedded の動作が開始できているかを確認するためには、次の動作状態表示コマンドを実行します。

```
cdms_emb_stat.exe
```

動作状態表示コマンドの詳細については、「3.6 動作状態表示コマンド」を参照してください。

(8) バックアップしたファイルの削除

バックアップしたファイルを削除します。

次のコマンドを実行して、削除してください。

```
del [バックアップ用ディレクトリ] %globals.backup  
del [バックアップ用ディレクトリ] %cdms_emb_db.backup
```

[バックアップ用ディレクトリ] は「(4) 入れ替え前のデータのバックアップ」で指定したデータベースのバックアップデータを格納するための任意のディレクトリです。

2

NX CDMS Embedded の機能

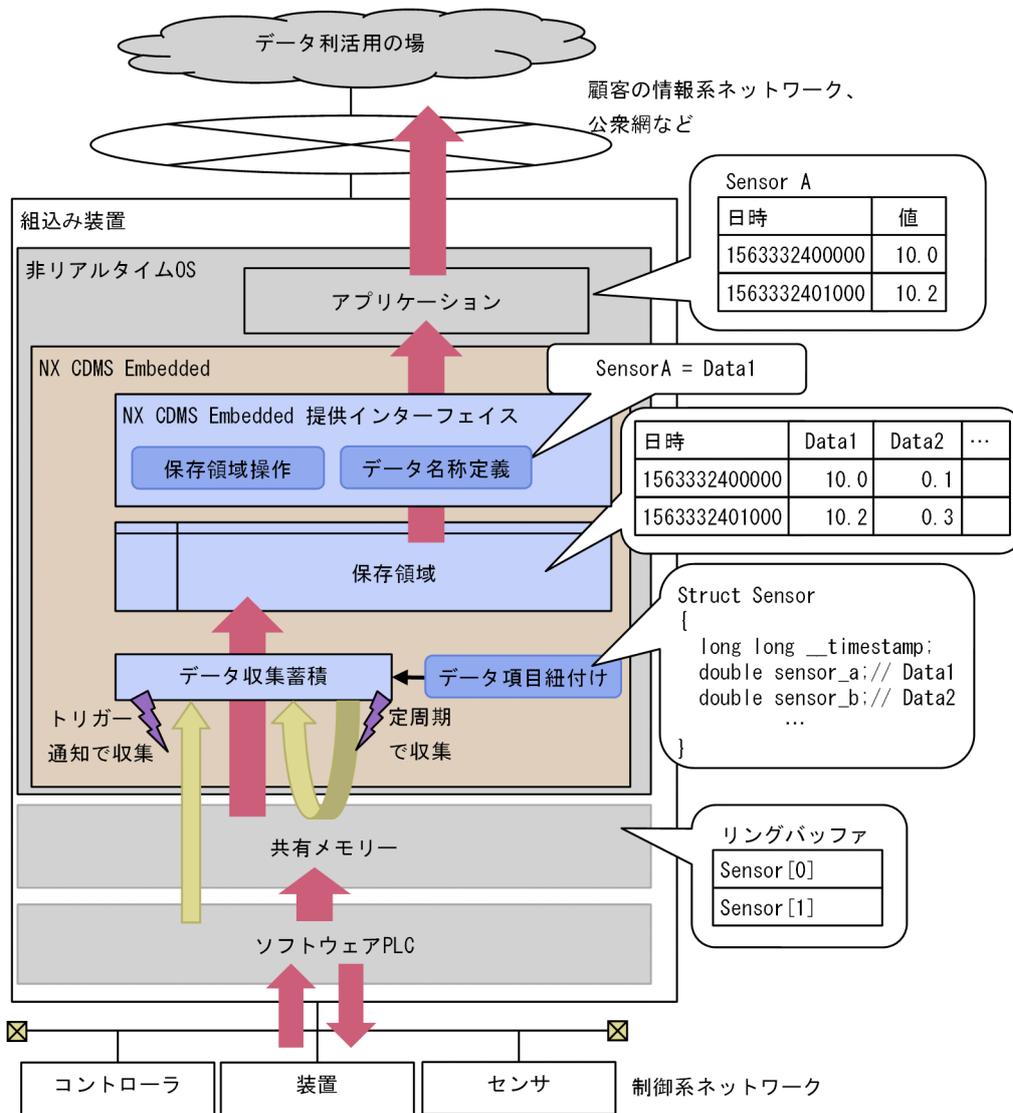
この章では、NX CDMS Embedded の機能について説明します。

2.1 サポート機能一覧

ここでは、NX CDMS Embedded のサポート機能について説明します。

NX CDMS Embedded が実現するシステム全体構成の例を次の図に示します。

図 2-1 NX CDMS Embedded が実現するシステム全体構成の例



(凡例)

- ➡ : データの流れ
- ➡ : 収集の流れ

NX CDMS Embedded がサポートする機能の一覧を次の表に示します。

表 2-1 NX CDMS Embedded サポート機能一覧

項番	機能	説明	参照項
1	起動停止機能	NX CDMS Embedded サービスの起動、および停止を行なうコマンドインターフェイスをサポートします。	「2.2 起動停止機能」
2	データ収集蓄積機能	共有メモリー上に書き込まれた時系列データを、定周期、または共有メモリーに設定されたトリガーの通知によって、保存領域に蓄積します。	「2.3 データ収集蓄積機能」
3	データ項目紐付け機能	共有メモリーの番地と保存領域での番号との紐付けを行ない、時系列データを管理します。	「2.4 データ項目紐付け機能」
4	保存領域操作機能	保存領域上のデータに対し、取得・削除を行なう API をサポートします。	「2.5 保存領域操作機能」
5	データ名称定義機能	保存領域上のデータに対し、別名称の定義を可能とします。	「2.6 データ名称定義機能」
6	RAS 機能	異常発生時のイベントログへの出力や障害情報の収集をします。	「2.7 RAS 機能」
7	情報／状態表示機能	NX CDMS Embedded のプログラムプロダクト情報、および NX CDMS Embedded の動作状態を表示するコマンドインターフェイスをサポートします。	「2.8 情報／状態表示機能」
8	インストーラ／アンインストーラ機能	プログラムプロダクトのインストール、およびアンインストールをサポートします。	「2.9 インストーラ／アンインストーラ機能」

2.2 起動停止機能

ここでは、NX CDMS Embedded の起動停止機能について説明します。

NX CDMS Embedded の起動停止は、計算機の起動停止時のサービスの自動起動、自動停止で実施します。

計算機の起動停止以外に、定義ファイル変更時などの NX CDMS Embedded を起動、停止するための起動コマンド、および、停止コマンドを提供します。

トリガーによるデータの収集中に NX CDMS Embedded を停止した場合、次回起動時にデータの収集を再開します。

データの収集が再開できるのは、計算機が停止した場合、または停止コマンドにより NX CDMS Embedded が正常に停止した場合です。

障害により NX CDMS Embedded が異常で停止した場合は再開できません。

NX CDMS Embedded が停止していた期間については、時刻を補間しデータは 0 が入ります。

サービス起動時の処理で NX CDMS Embedded が使用するテーブルに対し REINDEX を実行します。

起動コマンドの詳細については、「3.2 起動コマンド」を参照してください。

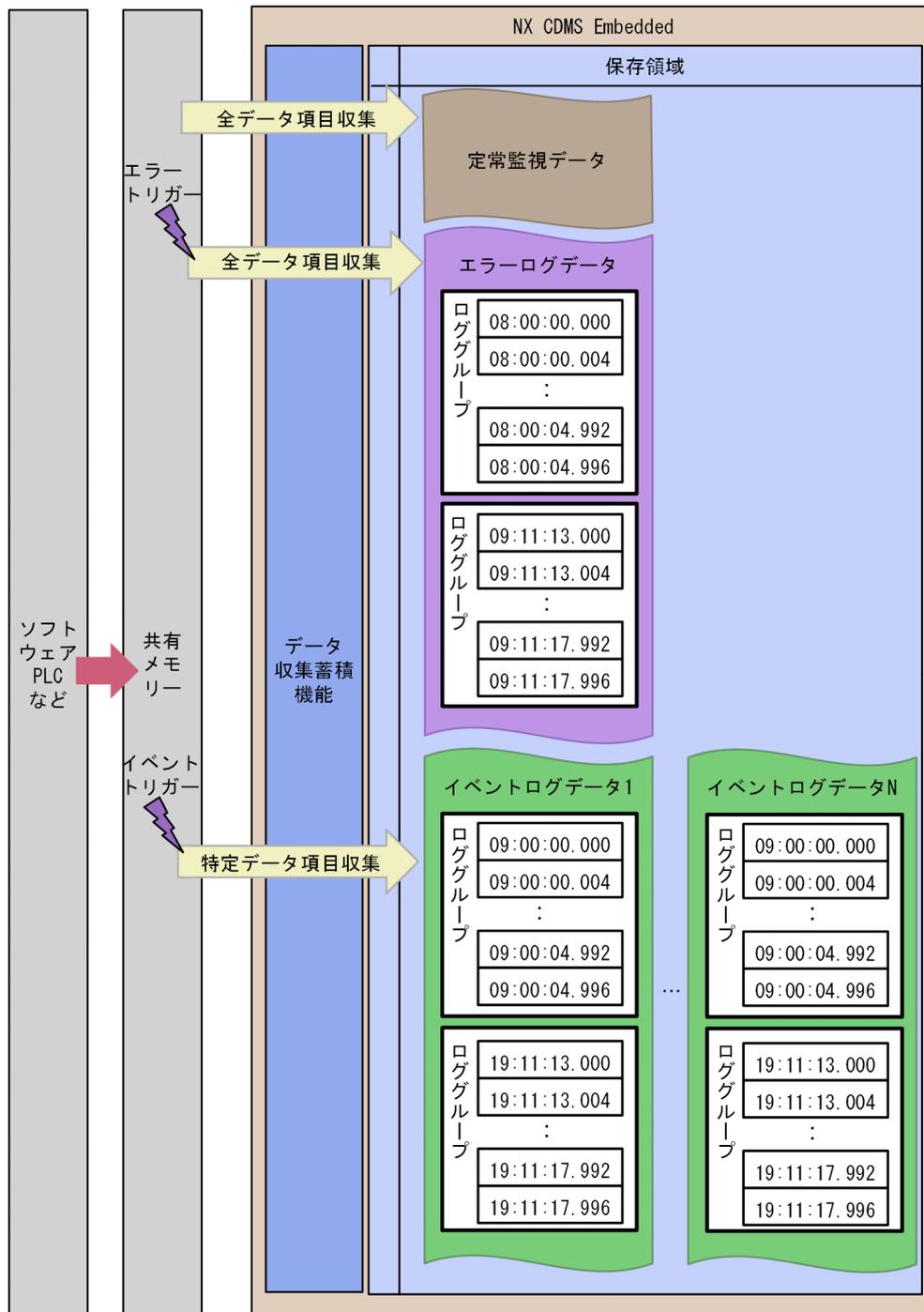
停止コマンドの詳細については、「3.3 停止コマンド」を参照してください。

2.3 データ収集蓄積機能

ここでは、NX CDMS Embedded のデータ収集蓄積機能について説明します。

NX CDMS Embedded のデータ収集蓄積機能は、共有メモリー上に書き込まれた時系列データを、定周期、またはトリガー通知によって収集します。

図 2-2 データ収集蓄積機能の概要図



(凡例)

… : 省略を表す

:

共有メモリーに書き込まれた時系列データを保存領域に蓄積する際、データを間引く間隔をサンプリング周期と呼びます。

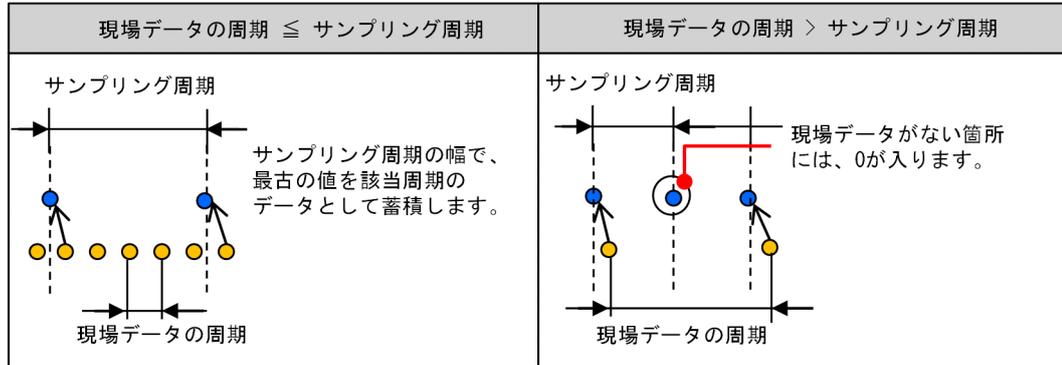
NX CDMS Embedded は時系列データを保存領域に蓄積する際、サンプリング周期の幅で最古の値を該当周期の値として蓄積します。

また、サンプリング周期が現場データの周期よりも小さい場合、現場データがないサンプリング周期の個所はデータの欠落として扱います。

そのため、サンプリング周期は現場データの周期以上とすることを推奨します。

サンプリング周期について、次の図に示します。

図 2-3 サンプリング周期



(凡例)

- : サンプリングデータ
- : 現場データ

共有メモリーに設定されたトリガー通知に関係なく、常時収集するデータを定常監視データと呼びます。

定常監視データはサンプリング周期を 1 秒として、すべてのデータ項目を収集します。

トリガーで収集するデータには、エラートリガーによるデータ収集と、イベントトリガーによるデータ収集があります。

エラートリガーによって収集するデータをエラーログデータと呼びます。イベントトリガーによって収集するデータをイベントログデータと呼びます。

トリガーが発生した時刻を基準にさかのぼってデータを取得する期間を Before 期間、トリガーが発生した時刻を基準にデータを取得し続ける期間を After 期間と呼びます。

トリガーによるデータ収集は、Before 期間と After 期間を合計した一定期間のデータを収集します。

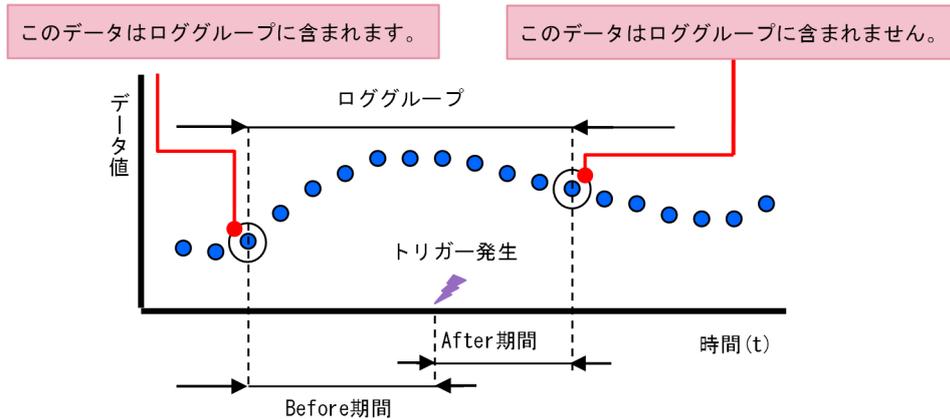
この通知によって収集した一定期間のデータの集まりをロググループと呼びます。

一定期間のデータを収集する際、ロググループの先頭時刻とデータの時刻が一致する場合、そのデータは該当ロググループに含まれます。

ロググループの末尾時刻とデータの時刻が一致する場合、そのデータは該当ロググループに含まれません。

ロググループについて、次の図に示します。

図 2-4 ロググループ



(凡例)

● : ログデータ

エラーログデータは全データを収集対象とし、収集する期間は Before 期間が 4 秒、After 期間が 1 秒の計 5 秒間です。

また、エラーログデータのサンプリング周期は 4 ミリ秒です。

イベントログデータは、特定のデータだけ収集します。

イベントログデータの収集する期間、取得するデータ、サンプリング周期は、イベントトリガー定義ファイルで定義します。

定義ファイルの詳細については、「5.9 イベントトリガー定義ファイル」を参照してください。

保存領域は、定常監視データ、エラーログデータとイベントログデータの 3 種類の領域に分かれています。

それぞれの領域に保存できるサイズの上限を次の表に示します。

表 2-2 保存領域に保存できるサイズの上限

保存領域	Windows
定常監視データ	40,000 レコード
エラーログデータ	38,750 レコード、または 31 ロググループ*
イベントログデータ	2,070,000 レコード、または 10,000 ロググループ*

注※

どちらか一方の上限まで保存できます。

上限に達した場合、定常監視データは、最古に登録したデータから順に上書きします。

エラーログデータとイベントログデータは、データの削除が実施されるまで保存は行なわれません。

不要になったデータをロググループ容量取得 API で確認し、定期的にロググループ削除 API で削除してください。

ロググループ削除 API、およびロググループ容量取得 API の詳細については、「4.6 ロググループ削除 API」、および「4.7 ロググループ容量取得 API」を参照してください。

なお、共有メモリーへの書き込みの際にデータが欠落していた場合、NX CDMS Embedded は欠落している個所を 0 埋めした状態で保存領域に蓄積します。

2.4 データ項目紐付け機能

ここでは、NX CDMS Embedded のデータ項目紐付け機能について説明します。

NX CDMS Embedded のデータ項目紐付け機能は、共有メモリーのデータ項目と保存領域のデータ項目の対応付けを管理します。

対応付けが変更された場合、保存領域はすべて削除します。

共有メモリーのデータ項目と保存領域のデータ項目の対応付けは、データ項目紐付け定義ファイルで定義します。

詳細については、「5.6 データ項目紐付け定義ファイル」を参照してください。

2.5 保存領域操作機能

ここでは、NX CDMS Embedded の保存領域操作機能について説明します。

NX CDMS Embedded の保存領域操作機能は、保存領域を操作する API を提供します。提供する API の一覧、および詳細については、「4 保存領域操作 API」を参照してください。

保存領域操作 API を使ったアプリケーションを作成し、NX CDMS Embedded で蓄積したデータの参照、および不要になったエラーログデータ、イベントログデータを削除してください。

エラーログデータ、イベントログデータは、それぞれ「2.3 データ収集蓄積機能」の保存領域に保存できるサイズの上限についての表に記載の上限まで蓄積し、上限を超えるとデータの収集を停止します。

保存領域操作 API を使用して、不要になったエラーログデータ、イベントログデータを削除するようにしてください。

2.6 データ名称定義機能

ここでは、NX CDMS Embedded のデータ名称定義機能について説明します。

NX CDMS Embedded のデータ名称定義機能は、保存領域操作 API で指定する取得対象のデータ名称と保存領域のデータ項目との対応付けを管理します。

保存領域操作 API で指定する取得対象のデータ名称と保存領域のデータ項目との対応付けは、データ名称定義ファイルで定義します。

詳細については、「5.7 データ名称定義ファイル」を参照してください。

2.7 RAS 機能

ここでは、NX CDMS Embedded の RAS 機能について説明します。

NX CDMS Embedded の RAS 機能は、障害発生時に解析に必要なログを出力します。

NX CDMS Embedded で出力するログは、イベントログだけです。

NX CDMS Embedded で出力するイベントログの詳細については、「6.1 Windows イベントログ一覧」を参照してください。

また、障害の原因を解析するために必要な情報を収集する RAS コマンドを提供します。

RAS コマンドの詳細については、「3.4 RAS コマンド」を参照してください。

2.8 情報／状態表示機能

ここでは、NX CDMS Embedded の情報／状態表示機能について説明します。

NX CDMS Embedded の情報／状態表示機能は、NX CDMS Embedded のプログラムプロダクト情報、および NX CDMS Embedded の動作状態を確認するコマンドを提供します。

プログラムプロダクト情報表示コマンドの詳細については、「3.5 プログラムプロダクト情報表示コマンド」を参照してください。

動作状態表示コマンドの詳細については、「3.6 動作状態表示コマンド」を参照してください。

2.9 インストーラ／アンインストーラ機能

ここでは、NX CDMS Embedded のインストーラ／アンインストーラ機能について説明します。

2.9.1 インストーラ

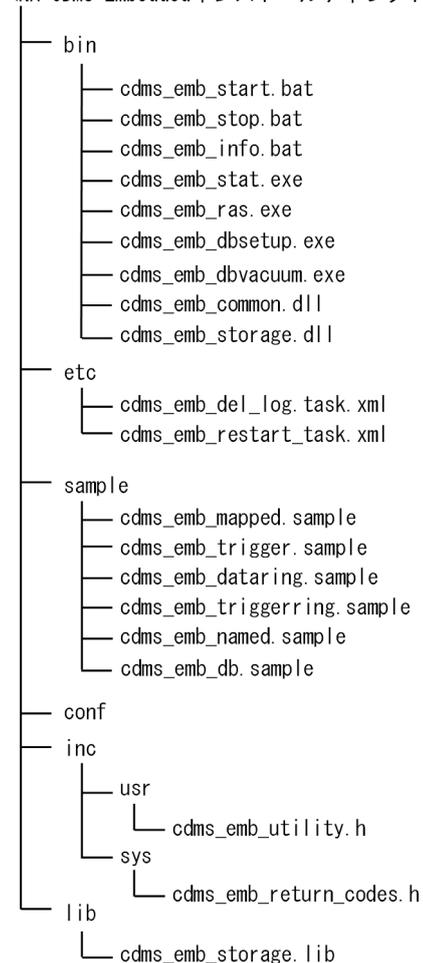
インストーラは、ファイルの作成、サービスの追加、および環境変数の設定を行います。

(1) ファイル一覧

インストーラで作成するファイルを次に示します。

図 2-5 ファイル一覧

<NX CDMS Embeddedインストールディレクトリ>



(2) サービス一覧

インストーラで作成するサービスを次の表に示します。

表 2-3 インストーラで追加するサービスの一覧

項番	サービス名	表示名	備考
1	NX_CDMS_EMB_Service	NX_CDMS_EMB_Service	以下のサービスに依存関係を設定します。

項番	サービス名	表示名	備考
			HX-DSM のサービス [HX_DSM_Manager] PostgreSQL のサービス [postgresql-x64-11]

(3) 環境変数一覧

インストーラで設定するシステム環境変数を次の表に示します。

表 2-4 インストーラで設定するシステム環境変数の一覧

項番	変数名	設定内容	備考
1	PATH	<i>NX CDMS Embedded</i> インストールディレクトリ¥bin	末尾に追加します。
2	NX_CDMS_EMB_DIR	<i>NX CDMS Embedded</i> インストールディレクトリ	デフォルト：C:¥NX-CDMS-EMB

2.9.2 アンインストーラ

アンインストーラは、インストーラで作成したファイル、サービス、レジストリを削除します。環境変数の再設定は行いません。

インストール後に、インストーラで作成したフォルダ配下にファイルが新たに作成された場合、該当するファイル、およびフォルダは削除されません。

3

コマンド一覧

この章では、NX CDMS Embedded で提供する各種コマンドについて説明します。

3.1 提供コマンド一覧

NX CDMS Embedded で提供するコマンドの一覧を次の表に示します。

表 3-1 提供コマンド一覧

項番	コマンド	コマンド名称	概要	参照先
1	起動コマンド	cdms_emb_start.bat	NX CDMS Embedded を起動するコマンド	[3.2 起動コマンド]
2	停止コマンド	cdms_emb_stop.bat	NX CDMS Embedded を停止するコマンド	[3.3 停止コマンド]
3	RAS コマンド	cdms_emb_ras.exe	障害解析に必要な情報を収集するコマンド	[3.4 RAS コマンド]
4	プログラムプロダクト情報表示コマンド	cdms_emb_info.bat	プログラムプロダクト情報を表示するコマンド	[3.5 プログラムプロダクト情報表示コマンド]
5	動作状態表示コマンド	cdms_emb_stat.exe	NX CDMS Embedded の動作状態を表示するコマンド	[3.6 動作状態表示コマンド]
6	DBセットアップコマンド	cdms_emb_dbsetup.exe	データベースを構築するコマンド	[3.7 DBセットアップコマンド]
7	DB VACUUM コマンド	cdms_emb_dbvacuum.exe	データベースの VACUUM を実施するコマンド	[3.8 DB VACUUM コマンド]

3.2 起動コマンド

3.2.1 コマンド形式

起動コマンドの形式を次に示します。

```
cdms_emb_start.bat
```

3.2.2 コマンドの説明

起動コマンドを実行すると、NX CDMS Embedded サービスを開始します。

3.2.3 前提条件

起動コマンドの前提条件を次の表に示します。

表 3-2 起動コマンドの前提条件

項番	項目	前提条件
1	NX CDMS Embedded 稼働状態	NX CDMS Embedded サービスが停止している状態で実行してください。
2	実行ユーザー	管理者権限があるユーザーで実行してください。
3	PostgreSQL 起動状態	PostgreSQL が起動している状態で実行してください

3.2.4 戻り値

起動コマンドの戻り値を次の表に示します。

表 3-3 起動コマンドの戻り値一覧

項番	戻り値	内容	条件
1	0	正常終了	NX CDMS Embedded サービスが開始しました。
2	1	多重起動エラー	NX CDMS Embedded サービスが停止していません。
3		管理者権限エラー	コマンド実行ユーザーに管理者権限が付与されていません。
4		起動エラー	NX CDMS Embedded サービスの開始に失敗しました。

3.2.5 メッセージ一覧

起動コマンドのメッセージ一覧を次の表に示します。

表 3-4 起動コマンドのメッセージ一覧

項番	メッセージテキスト	説明	対処方法
1	NX CDMS Embedded start success.	NX CDMS Embedded サービスの起動に成功しました。	—

3 コマンド一覧

項番	メッセージテキスト	説明	対処方法
2	NX CDMS Embedded is already running.	NX CDMS Embedded サービスがすでに起動中です。	—
3	This command needs to be executed as Administrator.	管理者権限が付与されていないユーザーで実行しています。	管理者権限が付与されているユーザーで実行してください。
4	NX CDMS Embedded start failed.	NX CDMS Embedded サービスの起動に失敗しました。	イベントログを確認して、「6 Windows イベントログ」に記載されている対処を行ってください。

(凡例)

—：対処不要

3.3 停止コマンド

3.3.1 コマンド形式

停止コマンドの形式を次に示します。

```
cdms_emb_stop.bat
```

3.3.2 コマンドの説明

停止コマンドを実行すると、NX CDMS Embedded サービスを停止します。

3.3.3 前提条件

停止コマンドの前提条件を次の表に示します。

表 3-5 停止コマンドの前提条件

項番	項目	前提条件
1	NX CDMS Embedded 稼働状態	NX CDMS Embedded サービスが開始している状態で実行してください。
2	実行ユーザー	管理者権限があるユーザーで実行してください。

3.3.4 戻り値

停止コマンドの戻り値を次の表に示します。

表 3-6 停止コマンドの戻り値一覧

項番	戻り値	内容	条件
1	0	正常終了	NX CDMS Embedded サービスが停止しました。
2	1	多重停止エラー	NX CDMS Embedded サービスが起動していません。
3		管理者権限エラー	コマンド実行ユーザーに管理者権限が付与されていません。

3.3.5 メッセージ一覧

停止コマンドのメッセージ一覧を次の表に示します。

表 3-7 停止コマンドのメッセージ一覧

項番	メッセージテキスト	説明	対処方法
1	NX CDMS Embedded stop success.	NX CDMS Embedded サービスの停止に成功しました。	—
2	NX CDMS Embedded is not running.	NX CDMS Embedded サービスが起動されていません。	—

3 コマンド一覧

項番	メッセージテキスト	説明	対処方法
3	This command needs to be executed as Administrator.	管理者権限が付与されていないユーザーで実行しています。	管理者権限が付与されているユーザーで実行してください。
4	NX CDMS Embedded stop failed.	NX CDMS Embedded サービスの停止に失敗しました。	イベントログを確認して、「6 Windows イベントログ」に記載されている対処を行ってください。

(凡例)

— : 対処不要

3.4 RAS コマンド

3.4.1 コマンド形式

RAS コマンドの形式を次に示します。

```
cdms_emb_ras.exe [-out path] [-level value]
```

オプションなし：

`NX_CDMS_EMB_DIR`¥log ディレクトリ下にコマンド実行時刻「yyyyMMddHHmmss」という名称のディレクトリを作成し、出力します。

実行時刻を表示する変数と内容を次の表に示します。

表 3-8 実行時刻表示文字と内容

項番	変数	内容
1	yyyy	西暦年
2	MM	月
3	dd	月に対する日
4	HH	時
5	mm	分
6	ss	秒

-out オプション：

path で指定した任意のディレクトリに出力します。path はフルパスで指定してください。ネットワークアドレスを含むパスは指定しないでください。

指定したディレクトリがない場合、指定したディレクトリを新規作成します。

指定したディレクトリがある場合、同名ファイルは上書き保存します。

-level オプション：

障害レベル (1,2) を指定します。障害レベルとは、障害解析に必要な情報ごとに設定されている値です。

このオプションは省略可能です。省略した場合、障害レベルを 1 として障害情報を収集します。

障害レベルと収集する情報の関係については、「3.4.2 コマンドの説明」を参照してください。

3.4.2 コマンドの説明

RAS コマンドを実行すると、NX CDMS Embedded の障害解析に必要な情報を収集します。

収集する対象の情報を次の表に示します。

表 3-9 コマンドの収集情報一覧

項番	収集対象	収集情報	障害レベル	収集対象の情報
1	OS	プロセス一覧	1	tasklist -V の実行結果

項番	収集対象	収集情報	障害レベル	収集対象の情報
2		環境変数一覧		set の実行結果
3		システム情報		systeminfo の実行結果
4		Windows イベントログ		wevtutil の実行結果
5	NX CDMS Embedded	NX CDMS Embedded のプログラムプロダクト情報	1	cdms_emb_info.bat の実行結果
6		NX CDMS Embedded の動作情報		cdms_emb_stat.exe の実行結果
7		NX CDMS Embedded の定義情報		<code>NX_CDMS_EMB_DIR</code> ¥conf 下の定義ファイル
8		NX CDMS Embedded の内部テーブル情報		NX CDMS Embedded のメモリーダンプ
9	PostgreSQL	システムログ	1	DB 接続設定ファイルの PostgreSQL のログフォルダ直下のファイル
10		データベースのダンプ		2

3.4.3 前提条件

RAS コマンドの前提条件を次の表に示します。

表 3-10 RAS コマンドの前提条件

項番	項目	前提条件
1	実行ユーザー	管理者権限があるユーザーで実行してください。
2	出力先空き容量	出力先ディレクトリに十分な空き容量を確保してください。 空き容量の目安 障害レベル 1 の場合：100MiB 障害レベル 2 の場合：1GiB
3	PostgreSQL	障害レベル 2 の場合、PostgreSQL が起動している必要があります。 PostgreSQL の起動状態の確認方法については、「(1) PostgreSQL の起動状態確認」の手順を参照してください。

3.4.4 戻り値

RAS コマンドの戻り値を次の表に示します。

表 3-11 RAS コマンドの戻り値一覧

項番	戻り値	内容	条件
1	0	正常終了	すべての障害情報の出力に成功しました。
2	1	オプションエラー	無効なオプションを指定しています。

項番	戻り値	内容	条件
3		出力エラー	ディレクトリの生成に失敗しました。 障害情報の出力に失敗しました。
4		管理者権限エラー	コマンド実行ユーザーに管理者権限が付与されていません。

3.4.5 メッセージ一覧

RAS コマンドのメッセージ一覧を次の表に示します。

表 3-12 RAS コマンドのメッセージ一覧

項番	メッセージテキスト	説明	対処方法
1	Write to <i>path</i> completed.	障害情報の出力に成功しました。 <i>path</i> : 出力先のフルパス	—
2	The parameter is incorrect. Usage: cdms_emb_ras.exe [-out path] [-level value]	オプションの指定が誤っています。	オプションの指定を見直しの上、実行してください。
3	Create <i>path</i> failed.	ディレクトリの生成に失敗しました。 <i>path</i> : 出力先のフルパス	<i>path</i> のパス名称を確認してください。
4	Write to <i>path</i> failed.	障害情報の出力に失敗しました。 <i>path</i> : 出力先のフルパス	<i>path</i> のアクセス権を確認してください。 <i>path</i> の空き容量を確認してください。
5	This command needs to be executed as Administrator.	管理者権限が付与されていないユーザーで実行しています。	管理者権限が付与されているユーザーで実行してください。
6	Configuration Error. (File= <i>name</i> , Section= <i>section</i> , Key= <i>key</i>)	DB 接続設定ファイルが誤っています。 <i>name</i> : ファイル名称 <i>section</i> : セクション名称 <i>key</i> : キー名称	該当の定義項目を修正したあと、再度実行してください。
7	PostgreSQL(pg_dump) Error. (<i>msg</i>)	PostgreSQL の pg_dump コマンドでエラーが発生しました。 <i>msg</i> : エラーメッセージ	pg_dump コマンドのエラーメッセージに従って対処してください。 対処できない場合は、RAS コマンドにて障害レベルが 1 の障害情報を収集の上、問い合わせ窓口へご連絡ください。

3 コマンド一覧

(凡例)

－：対処不要

3.5 プログラムプロダクト情報表示コマンド

3.5.1 コマンド形式

プログラムプロダクト情報表示コマンドの形式を次に示します。

```
cdms_emb_info.bat
```

3.5.2 コマンドの説明

プログラムプロダクト情報表示コマンドを実行すると、インストールされている NX CDMS Embedded プログラムプロダクトの型式名称、バージョン、および SI 回数を表示します。

3.5.3 前提条件

プログラムプロダクト情報表示コマンドの前提条件を次の表に示します。

表 3-13 プログラムプロダクト情報表示コマンドの前提条件

項番	項目	前提条件
1	実行ユーザー	管理者権限があるユーザーで実行してください。

3.5.4 戻り値

プログラムプロダクト情報表示コマンドの戻り値を次の表に示します。

表 3-14 プログラムプロダクト情報表示コマンドの戻り値一覧

項番	戻り値	内容	条件
1	0	正常終了	プログラムプロダクト情報の表示に成功しました。
2	1	管理者権限エラー	コマンド実行ユーザーに管理者権限が付与されていません。

3.5.5 メッセージ一覧

プログラムプロダクト情報表示コマンドのメッセージ一覧を次の表に示します。

表 3-15 プログラムプロダクト情報表示コマンドのメッセージ一覧

項番	メッセージテキスト	説明	対処方法
1	[NX CDMS Embedded Information] [P.P. Name] [Model] [Version] [SI] NX-CDMS-EMB <i>model ver</i> [<i>si</i>]	プログラムプロダクト情報の表示に成功しました。 <i>model</i> : 型式 <i>ver</i> : バージョン情報 <i>si</i> : SI 回数	—
2	This command needs to be executed as Administrator.	管理者権限が付与されていないユーザーで実行しています。	管理者権限が付与されているユーザーで実行してください。

3 コマンド一覧

(凡例)

－：対処不要

3.6 動作状態表示コマンド

3.6.1 コマンド形式

動作状態表示コマンドの形式を次に示します。

```
cdms_emb_stat.exe
```

3.6.2 コマンドの説明

動作状態表示コマンドを実行すると、NX CDMS Embedded の動作状態を表示します。

3.6.3 前提条件

動作状態表示コマンドの前提条件を次の表に示します。

表 3-16 動作状態表示コマンドの前提条件

項番	項目	前提条件
1	実行ユーザー	管理者権限があるユーザーで実行してください。

3.6.4 戻り値

動作状態表示コマンドの戻り値を次の表に示します。

表 3-17 動作状態表示コマンドの戻り値一覧

項番	戻り値	内容	条件
1	0	正常終了	動作状態の表示に成功しました。
2	1	管理者権限エラー	コマンド実行ユーザーに管理者権限が付与されていません。

3.6.5 メッセージ一覧

動作状態表示コマンドのメッセージ一覧を次の表に示します。

表 3-18 動作状態表示コマンドのメッセージ一覧

項番	メッセージテキスト	説明	対処方法
1	[NX CDMS Embedded Status Information] [Name] [Status] NX-CDMS-EMB <i>status</i>	動作状態の表示に成功しました。 <i>status</i> ：動作状態 STOP：停止中 INITIAL：初期化中 RUN：正常動作中 ERROR：異常発生中	—
2	This command needs to be executed as Administrator.	管理者権限が付与されていないユーザーで実行しています。	管理者権限が付与されているユーザーで実行してください。

3 コマンド一覧

(凡例)

－：対処不要

3.7 DB セットアップコマンド

3.7.1 コマンド形式

DB セットアップコマンドの形式を次に示します。

```
cdms_emb_dbsetup.exe
```

3.7.2 コマンドの説明

NX CDMS Embedded のデータベース構築を実施します。

DB セットアップコマンドは、NX CDMS Embedded が使用するデータベースに接続用のユーザーやデータベースを作成します。

このため、コマンド実行時にスーパーユーザーのユーザー名(postgres)とパスワードを確認します。

入力表示フォーマットは、次のとおりです。

```
# cdms_emb_dbsetup <Enter>
User : user
Password : password
```

userにはスーパーユーザーのユーザー名 [postgres] を入力し、passwordにはスーパーユーザーのパスワード (PostgreSQL インストール時に設定したパスワード) を入力してください。

(1) 機能

DB セットアップコマンドは、DB 接続定義ファイルの定義に従って以下の作成や設定を行い、NX CDMS Embedded のデータベースを構築します。

下記の定義項目については、「5.6.3 定義記述方法」を参照してください。

- ユーザー作成
DB 接続定義ファイルのユーザー情報で、NX CDMS Embedded の DB 接続用ユーザーを作成します。
<作成するユーザー名> : (定義項目 : User)
<作成するパスワード> : (定義項目 : Password)
- データベース作成
DB 接続定義ファイルのデータベース名称で、データベースを作成します。
<作成するデータベース名> : (定義項目 : DbName)
<データベースを所有するユーザー名> : (定義項目 : User)
- スキーマ作成
データベースのスキーマを作成します。
<作成するスキーマ名> : cdms_emb_db
- タイムゾーン設定
データベースにタイムゾーンを設定します。
<設定するタイムゾーン> : UTC

- テーブル作成
NX CDMS Embedded のデータベースで用いるテーブルを作成します。

3.7.3 前提条件

DB セットアップコマンドの前提条件を次の表に示します。

表 3-19 DB セットアップコマンドの前提条件

項番	項目	前提条件
1	実行ユーザー	管理者権限があるユーザーで実行してください。
2	PostgreSQL 起動状態	PostgreSQL が起動している状態で実行してください。

3.7.4 戻り値

DB セットアップコマンドの戻り値を次の表に示します。

表 3-20 DB セットアップコマンドの戻り値一覧

項番	戻り値	項目	前提条件
1	0	正常終了	コマンドが正常終了しました。
2	1	異常終了	コマンドが異常終了しました。

3.7.5 メッセージ一覧

メッセージの一覧を次の表に示します。

表 3-21 DB セットアップコマンドのメッセージ一覧

項番	メッセージ テキスト	説明	対処方法
1	DB setup success.	DB セットアップに成功しました。	—
2	Configuration Error. (File= <i>name</i> , Section= <i>section</i> , Key= <i>key</i>)	DB 接続設定ファイルが誤っています。 <i>name</i> : ファイル名称 <i>section</i> : セクション名称 <i>key</i> : キー名称	該当の定義項目を修正したあと、再度実行してください。
3	PostgreSQL(<i>psql</i>) Error. (Message= <i>msg</i>)	PostgreSQL の <i>psql</i> コマンドでエラーが発生しました。 <i>msg</i> : 詳細メッセージ	表示されているメッセージ内容に従って対処を実施してください。対処できない場合は RAS コマンドにて障害レベルが 1 の障害情報を収集の上、問い合わせ窓口へご連絡ください。

項番	メッセージ テキスト	説明	対処方法
4	DB setup has already completed.	DB セットアップはすでに完了しました。	初期構築以降の手順を実施してください。
5	DB setup failed. (エラーの詳細情報)	DB セットアップに失敗しました。	RAS コマンドにて障害レベルが1の障害情報を収集の上、問い合わせ窓口へご連絡ください。
6	This command needs to be executed as Administrator.	管理者権限が付与されていないユーザーで実行しています。	管理者権限が付与されているユーザーで実行してください。

3.8 DB VACUUM コマンド

3.8.1 コマンド形式

DB VACUUM コマンドを次に示します。

```
cdms_emb_dbvacuum.exe
```

3.8.2 コマンドの説明

NX CDMS Embedded が使用するデータベースの VACUUM を実施します。

DB VACUUM コマンドを実行すると、不要なディスク使用領域を解放し、ディスク空き容量を確保できます。ディスク容量が圧迫されているときに実行してください。

3.8.3 前提条件

DB VACUUM コマンドの前提条件を次の表に示します。

表 3-22 DB VACUUM コマンドの前提条件

項番	項目	前提条件
1	NX CDMS Embedded 稼働状態	NX CDMS Embedded サービスが停止している状態で実行してください。
2	実行ユーザー	管理者権限があるユーザーで実行してください。
3	PostgreSQL 起動状態	PostgreSQL が起動している状態で実行してください。

3.8.4 戻り値

DB セットアップコマンドの戻り値を次の表に記載します。

表 3-23 DB VACUUM コマンドの戻り値一覧

項番	戻り値	項目	前提条件
1	0	正常終了	コマンドが正常終了しました。
2	1	異常終了	コマンドが異常終了しました。

3.8.5 メッセージ一覧

メッセージの一覧を次の表に示します。

表 3-24 DB VACUUM コマンドのメッセージ一覧

項番	メッセージ テキスト	説明	対処方法
1	DB vacuum success.	データベースの VACUUM に成功しました。	—

項番	メッセージ テキスト	説明	対処方法
2	Configuration Error. (File= <i>name</i> , Section= <i>section</i> , Key= <i>key</i>)	DB 接続設定ファイルが誤っています。 <i>name</i> : ファイル名称 <i>section</i> : セクション名称 <i>key</i> : キー名称	該当の定義項目を修正したあと、再度実行してください。
3	PostgreSQL(psql) Error. (Message= <i>msg</i>)	PostgreSQL の psql コマンドでエラーが発生しました。 <i>msg</i> : 詳細メッセージ	表示されているメッセージ内容に従って対処を実施してください。対処できない場合は RAS コマンドにて障害レベルが 1 の障害情報を収集の上、問い合わせ窓口へご連絡ください。
4	DB vacuum failed. (エラーの詳細情報)	データベースの VACUUM に失敗しました。	RAS コマンドにて障害レベルが 1 の障害情報を収集の上、問い合わせ窓口へご連絡ください。
5	This command needs to be executed as Administrator.	管理者権限が付与されていないユーザーで実行しています。	管理者権限が付与されているユーザーで実行してください。
6	NX CDMS Embedded is already running.	NX CDMS Embedded サービスがすでに起動中です。	NX CDMS Embedded を停止してからコマンドを実行してください。

4

保存領域操作 API

この章では、保存領域に蓄積したデータの参照や削除を行う API について説明します。

4.1 保存領域操作 API 一覧

NX CDMS Embedded は、保存領域に蓄積したデータの参照、および削除する API を提供します。

提供する API の一覧を次の表に示します。

表 4-1 提供する API の一覧

項番	API 名	機能概要	参照先
1	初期化 API	保存領域操作 API の使用に係る初期化処理を行います。	[4.2 初期化 API]
2	終了処理 API	保存領域操作 API の使用に係る終了処理を行います。	[4.3 終了処理 API]
3	定常監視データ読込 API	定常監視データから値を読み込みます。	[4.4 定常監視データ読込 API]
4	ロググループ読込 API	ロググループの値を読み込みます。	[4.5 ロググループ読込 API]
5	ロググループ削除 API	ロググループを削除します。	[4.6 ロググループ削除 API]
6	ロググループ容量取得 API	ロググループの容量を取得します。	[4.7 ロググループ容量取得 API]

4.2 初期化 API

ここでは、初期化 API について説明します。

4.2.1 API の説明

初期化 API は、保存領域操作 API の使用に係る初期化処理を行います。

初期化処理が未実行の場合は初期化処理を実行し、実行済みの場合は何も行いません。

4.2.2 インターフェイス

```
int CDMS_EMB_WinInitialize(void);
```

4.2.3 戻り値

初期化 API の戻り値を次の表に示します。説明や対処方法については、「4.8 リターンコード一覧」を参照してください。

表 4-2 初期化 API の戻り値

項番	リターンコード	値
1	CDMS_EMB_SUCCESS	0
2	E_CDMS_EMB_NOTRUNNING	-2
3	E_CDMS_EMB_MAXTHREAD	-16
4	E_CDMS_EMB_GETENV	-102
5	E_CDMS_EMB_MEMORY	-103
6	E_CDMS_EMB_POSTGRESQL_MAXCONNECTION	-302

4.3 終了処理 API

ここでは、終了処理 API について説明します。

4.3.1 API の説明

終了処理 API は、保存領域操作 API の使用に係る終了処理を行います。

終了処理が未実行の場合は終了処理を実行し、実行済みの場合は何も行いません。

4.3.2 インターフェイス

```
int CDMS_EMB_WinFinalize(void);
```

4.3.3 戻り値

終了処理 API の戻り値を次の表に示します。説明や対処方法については、「4.8 リターンコード一覧」を参照してください。

表 4-3 終了処理 API の戻り値

項番	リターンコード	値
1	CDMS_EMB_SUCCESS	0

4.4 定常監視データ読込 API

ここでは、定常監視データ読込 API について説明します。

4.4.1 API の説明

定常監視データ読込 API は、定常監視データからデータ名および時刻を指定してデータを読み込みます。

4.4.2 インターフェイス

```
int CDMS_EMB_WinReadNormalLog(
    const char** targets,
    size_t count_targets,
    int64_t begin_timestamp,
    int64_t end_timestamp,
    size_t max_results,
    int32_t flag_prioritize,
    char** results,
    size_t* count_results
);
```

次の表に定常監視データ読込 API の引数一覧を示します。

表 4-4 定常監視データ読込 API の引数一覧

項番	引数	種類	説明
1	targets	in	取得対象とするデータ名の配列。
2	count_targets	in	取得対象とするデータ名の件数。
3	begin_timestamp	in	開始時刻 (0、または西暦 3000 年 1 月 1 日 0 時 0 分 0 秒 0 ミリ秒以降を指定した場合、無限遠の過去時刻を表すものとします)。
4	end_timestamp	in	終了時刻 (0、または西暦 3000 年 1 月 1 日 0 時 0 分 0 秒 0 ミリ秒以降を指定した場合、無限遠の未来時刻を表すものとします)。
5	max_results	in	読み込み件数の上限値。
6	flag_prioritize	in	読み込み可能なデータの件数が読み込み件数の上限値を超える場合に、開始時刻側か終了時刻側のどちらに寄せて読み込むかを指定します。 <ul style="list-style-type: none"> • BEGIN_TIME を指定した場合、開始時刻側に寄せてデータを読み込みます。 • END_TIME を指定した場合、終了時刻側に寄せてデータを読み込みます。
7	results	out	データ名とタイムスタンプに対応する値の二次元配列。データ時刻の昇順でソートします。 二次元配列のイメージを「表 4-5 定常監視データ読込 API の引数 results の二次元配列イメージ」に示します。
8	count_results	out	結果の件数。

定常監視データ読込 API の引数 results の二次元配列イメージを次の表に示します。

表 4-5 定常監視データ読込 API の引数 results の二次元配列イメージ

件数	データ時刻	データ名 1	...	データ名 Y
1 件目	2019/08/01 10:00:00.000	123	...	aaa
...				
X 件目	2019/08/01 10:00:10.000	456	...	ccc

(凡例)

...：省略を表す。

X の値が結果の件数 (count_results)、Y の値がデータ名の件数 (count_targets) になります。各データは、一つ当たり 8 バイトとして扱います。

4.4.3 戻り値

定常監視データ読込 API の戻り値を次の表に示します。説明や対処方法については、「4.8 リターンコード一覧」を参照してください。

表 4-6 定常監視データ読込 API の戻り値

項番	リターンコード	値
1	CDMS_EMB_SUCCESS	0
2	E_CDMS_EMB_NOTREADY	-1
3	E_CDMS_EMB_NOTRUNNING	-2
4	E_CDMS_EMB_NORECORDS	-13
5	E_CDMS_EMB_INVALIDARGUMENT	-14
6	E_CDMS_EMB_INTERNAL	-101
7	E_CDMS_EMB_GET_ENV	-102
8	E_CDMS_EMB_MEMORY	-103
9	E_CDMS_EMB_POSTGRESQL	-301
10	E_CDMS_EMB_POSTGRESQL_DISKFULL	-303
11	E_CDMS_EMB_POSTGRESQL_MEMORY	-304
12	E_CDMS_EMB_POSTGRESQL_NOTRUNNING	-305

4.5 ロググループ読込 API

ここでは、ロググループ読込 API について説明します。

4.5.1 API の説明

ロググループ読込 API は、基点時刻を指定して、指定時刻に最も近いロググループを読み込みます。

4.5.2 インターフェイス

```
int CDMS_EMB_WinReadLogGroup(
    int32_t      flag_target,
    int64_t      begin_timestamp,
    int32_t      flag_direction,
    char**       results,
    size_t*      count_results
);
```

次の表にロググループ読込 API の引数一覧を示します。

表 4-7 ロググループ読込 API の引数一覧

項番	引数	種類	説明
1	flag_target	in	取得対象とするトリガー。 ERROR_TRIGGER, EVENT_TRIGGER_X (X は 1~16 が入る) の いずれかのマクロを指定します。
2	begin_timestamp	in	ロググループを取得する際の基点となる時刻 (西暦 3000 年 1 月 1 日 0 時 0 分 0 秒 0 ミリ秒以降を指定した場合、無限遠の未来時刻を表す ものとします)。
3	flag_direction	in	基点となる時刻より未来の時刻で近いロググループを取得するか、過 去の時刻で近いロググループを取得するか指定します。 <ul style="list-style-type: none"> • POSITIVE を指定した場合、基点の時刻より未来の時刻のログ グループを取得します。 • NEGATIVE を指定した場合、基点の時刻より過去の時刻のログ グループを取得します。
4	results	out	読み込んだデータの二次元配列。データ時刻の昇順でソートします。 二次元配列のイメージを「表 4-8 ロググループ読込 API の引数 results の二次元配列イメージ」に示します。
5	count_results	out	結果の件数。

ロググループ読込 API の引数 results の二次元配列イメージを次の表に示します。

表 4-8 ロググループ読込 API の引数 results の二次元配列イメージ

件数	データ時刻	Data1	...	DataN
1 件目	2019/08/01 10:00:00.000	123	...	aaa
...				
X 件目	2019/08/01 10:00:10.000	456	...	ccc

(凡例)

…：省略を表す。

X の値が結果の件数 (count_results) です。Data1 から DataN には、取得したデータ項目を、次に示す順にソートして格納します。

- ERROR_TRIGGER を対象とした場合、データ項目紐づけ定義ファイルの Data セクション順 (昇順)
- EVENT_TRIGGER_X を対象とした場合、イベントトリガー定義ファイルの該当イベントトリガーセクションの TargetData キーに記載した順

各データは、一つ当たり 8 バイトとして扱います。

4.5.3 戻り値

ロググループ読み API の戻り値を次の表に示します。説明や対処方法については、「4.8 リターンコード一覧」を参照してください。

表 4-9 ロググループ読み API の戻り値

項番	リターンコード	値
1	CDMS_EMB_SUCCESS	0
2	E_CDMS_EMB_NOTREADY	-1
3	E_CDMS_EMB_NOTRUNNING	-2
4	E_CDMS_EMB_NORECORDS	-13
5	E_CDMS_EMB_INVALIDARGUMENT	-14
6	E_CDMS_EMB_INTERNAL	-101
7	E_CDMS_EMB_MEMORY	-103
8	E_CDMS_EMB_POSTGRESQL	-301
9	E_CDMS_EMB_POSTGRESQL_DISKFULL	-303
10	E_CDMS_EMB_POSTGRESQL_MEMORY	-304
11	E_CDMS_EMB_POSTGRESQL_NOTRUNNING	-305

4.6 ロググループ削除 API

ここでは、ロググループ削除 API について説明します。

4.6.1 API の説明

ロググループ削除 API は、ロググループ ID を指定して、ロググループを削除します。

ロググループ ID は、ロググループ容量取得 API で削除対象のロググループを確認して指定してください。

同一ロググループ ID に対して、ロググループ削除 API を複数の処理から同時実行しないでください。

4.6.2 インターフェイス

```
int CDMS_EMB_WinDeleteLogGroup(
    int32_t      flag_target,
    uint64_t     loggroup_id
);
```

次の表にロググループ削除 API の引数一覧を示します。

表 4-10 ロググループ削除 API の引数一覧

項番	引数	種類	説明
1	flag_target	in	削除対象とするトリガー。 DELETE_ERROR_TRIGGER または DELETE_EVENT_TRIGGER のいずれかのマクロを指定します。
2	loggroup_id	in	削除対象とするロググループ ID。

4.6.3 戻り値

ロググループ削除 API の戻り値を次の表に示します。説明や対処方法については、「4.8 リターンコード一覧」を参照してください。

表 4-11 ロググループ削除 API の戻り値

項番	リターンコード	値
1	CDMS_EMB_SUCCESS	0
2	E_CDMS_EMB_NOTREADY	-1
3	E_CDMS_EMB_NOTRUNNING	-2
4	E_CDMS_EMB_IDNOTFOUND	-11
5	E_CDMS_EMB_GATHERING	-12
6	E_CDMS_EMB_INVALIDARGUMENT	-14
7	E_CDMS_EMB_POSTGRESQL	-301
8	E_CDMS_EMB_POSTGRESQL_DISKFULL	-303
9	E_CDMS_EMB_POSTGRESQL_MEMORY	-304

4 保存領域操作 API

項番	リターンコード	値
10	E_CDMS_EMB_POSTGRESQL_NOTRUNNING	-305

4.7 ロググループ容量取得 API

ここでは、ロググループ容量取得 API について説明します。

4.7.1 API の説明

ロググループ容量取得 API は、ログ容量の一覧を取得します。

4.7.2 インターフェイス

```
int CDMS_EMB_WinLogStatistics(
    const int32_t      flag_target,
    const uint64_t     max_results,
    uint64_t*         total_results,
    struct loggroup_info* results,
    uint64_t*         count_results
);
struct loggroup_info {
    uint64_t      loggroup_id;
    int64_t       timestamp;
    size_t        size;
};
```

次の表にロググループ容量取得 API の引数一覧を示します。

表 4-12 ロググループ容量取得 API の引数一覧

項番	引数	種類	説明
1	flag_target	in	取得対象とするトリガー。 ERROR_TRIGGER または EVENT_TRIGGER_X (X は 1~16 が入る) のいずれかのマクロを指定します。
2	max_results	in	ログ容量取得結果として確保した件数。
3	total_results	out	取得対象に指定したトリガーが持つロググループ ID の総数。
4	results	out	ログ容量取得結果の 1 次元配列。 構造体 loggroup_info を配列で、最大ログ容量取得結果として確保した件数分格納します。 構造体 loggroup_info の詳細は、「表 4-13 ロググループ容量取得 API の取得結果の構造」を参照してください。 ロググループ ID の昇順 (トリガー発生時刻が古い順) にソートします。
5	count_results	out	結果の件数。

ログ容量取得結果の 1 次元配列の構造を次の表に示します。

表 4-13 ロググループ容量取得 API の取得結果の構造

項番	引数	説明
1	loggroup_id	ロググループ ID。
2	timestamp	トリガー発生時刻。
3	size	ロググループの容量 (レコード数)。

4.7.3 戻り値

ロググループ容量取得 API の戻り値を次の表に示します。説明や対処方法については、「4.8 リターンコード一覧」を参照してください。

表 4-14 ロググループ容量取得 API の戻り値

項番	リターンコード	値
1	CDMS_EMB_SUCCESS	0
2	E_CDMS_EMB_NOTREADY	-1
3	E_CDMS_EMB_NOTRUNNING	-2
4	E_CDMS_EMB_NORECORDS	-13
5	E_CDMS_EMB_INVALIDARGUMENT	-14
6	E_CDMS_EMB_INTERNAL	-101
7	E_CDMS_EMB_POSTGRESQL	-301
8	E_CDMS_EMB_POSTGRESQL_DISKFULL	-303
9	E_CDMS_EMB_POSTGRESQL_MEMORY	-304
10	E_CDMS_EMB_POSTGRESQL_NOTRUNNING	-305

4.8 リターンコード一覧

保存領域操作 API が戻り値として返すリターンコードの一覧を次の表に示します。

表 4-15 保存領域 API のリターンコード一覧

項番	リターンコード	値	説明	対処方法
1	CDMS_EMB_SUCCESS	0	正常終了しました。	対処は不要です。
2	E_CDMS_EMB_NOTREADY	-1	保存領域の初期化が未完了です。	保存領域の初期化 API (4.2 初期化 API) をコールしてから、再度実行してください。
3	E_CDMS_EMB_NOTRUNNING	-2	NX CDMS Embedded のサービスが停止しています。	起動コマンド (3.2 起動コマンド) を実行してから、再度実行してください。
4	E_CDMS_EMB_IDNOTFOUND	-11	引数で指定したロググループ ID が見つかりません。	指定するロググループ ID は、ロググループ容量取得 API (4.7 ロググループ容量取得 API) で取得した ID を指定してください。
5	E_CDMS_EMB_GATHERING	-12	引数で指定したロググループ ID はデータ収集中です。	指定するロググループ ID は、ロググループ容量取得 API (4.7 ロググループ容量取得 API 参照) で取得した ID を指定してください。
6	E_CDMS_EMB_NO RECORDS	-13	引数で指定した条件のレコードが見つかりません。	引数を見直し、再度実行してください。
7	E_CDMS_EMB_INVALID_ARGUMENT	-14	引数に誤りがあります。	引数を見直し、再度実行してください。
8	E_CDMS_EMB_READING	-15	ファイルを読み込み中です。	ロググループ読込 API ([4.5 ロググループ読込 API] 参照) やロググループ容量取得 API ([4.7 ロググループ容量取得 API] 参照) の実行が完了したあと、再度実行してください。
9	E_CDMS_EMB_MAXTHREADS	-16	同時に実行可能な最大スレッド数をオーバーしました。	同時に実行可能な最大スレッド数の範囲で API を実行してください。
10	E_CDMS_EMB_INTERNAL	-101	内部エラーが発生しました。	RAS コマンドを使用して障害レベルが、1 と 2 の障害情報を収集の上、問い合わせ窓口へご連絡ください。
11	E_CDMS_EMB_GETENV	-102	環境変数が見つかりません。NX CDMS Embedded が正常にインストールされていません。	NX CDMS Embedded をアンインストール後、再度インストールしてください。
12	E_CDMS_EMB_MEMORY	-103	API 実行のためのメモリーが不足しています。	NX CDMS Embedded の実行に必要なメモリーを確保してから、再度実行してください。

項番	リターンコード	値	説明	対処方法
13	E_CDMS_EMB_POSTGRESQL	-301	PostgreSQL でエラーが発生しました。	RAS コマンドにて障害レベルが 1 と 2 の障害情報を収集の上、問い合わせ窓口へご連絡ください。
14	E_CDMS_EMB_POSTGRESQL_MAXCONNECTION	-302	PostgreSQL の最大接続数をオーバーしました。	PostgreSQL に接続する不要なアプリケーションを終了してから、再度実行してください。
15	E_CDMS_EMB_POSTGRESQL_DISKFULL	-303	PostgreSQL が使用するディスクの空き容量が不足しています。	PostgreSQL のデータ保存先ディレクトリ※が存在するディスクの空き容量を確保してから、再度実行してください。
16	E_CDMS_EMB_POSTGRESQL_MEMORY	-304	PostgreSQL を実行するためのメモリーが不足しています。	本製品の実行に必要なメモリーを確保してから、再度実行してください。
17	E_CDMS_EMB_POSTGRESQL_NOTRUNNING	-305	PostgreSQL が停止しています。	PostgreSQL を起動してから再度実行してください。

注※

「(a) PostgreSQL のインストール」の手順 5 で指定したディレクトリ

5

定義ファイル

この章では、NX CDMS Embedded で使用する定義ファイルについて説明します。

5.1 定義ファイル一覧

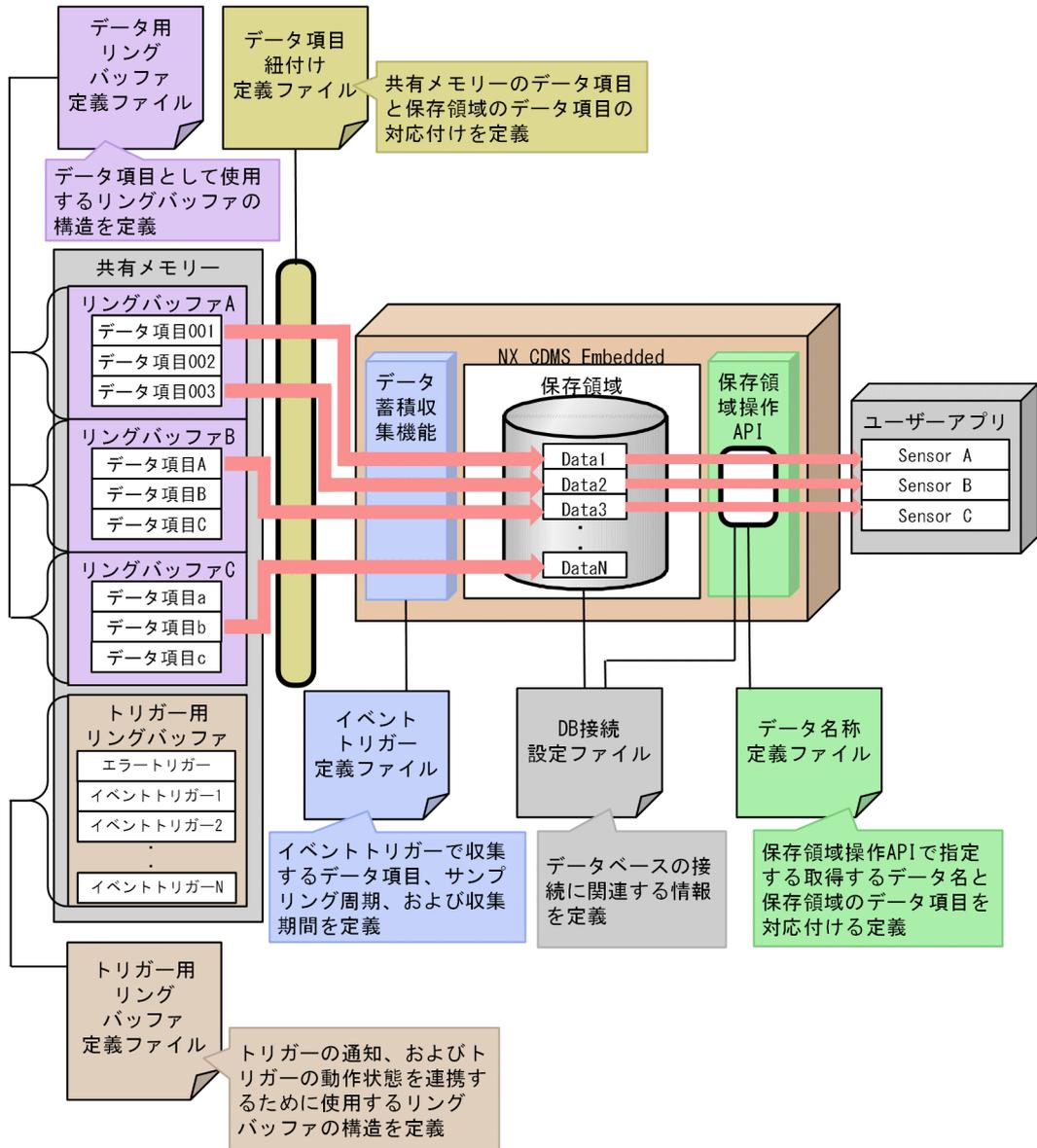
NX CDMS Embedded で使用する定義ファイルの一覧を次の表に示します。

表 5-1 定義ファイルの一覧

項番	定義ファイル名	定義内容	参照先
1	データ用リングバッファ定義ファイル	データ項目として使用する共有メモリーのリングバッファの構造を定義します。	[5.5 データ用リングバッファ定義ファイル]
2	データ項目紐付け定義ファイル	共有メモリーのデータ項目と保存領域のデータ項目の対応付けを定義します。	[5.6 データ項目紐付け定義ファイル]
3	データ名称定義ファイル	保存領域操作 API で指定する取得対象のデータ名称と保存領域のデータ項目の対応付けを定義します。	[5.7 データ名称定義ファイル]
4	トリガー用リングバッファ定義ファイル	トリガーの通知、およびトリガーの動作状態を連携するために使用するリングバッファの構造を定義します。	[5.8 トリガー用リングバッファ定義ファイル]
5	イベントトリガー定義ファイル	イベントトリガーで収集するデータ項目、サンプリング周期、および収集期間を定義します。	[5.9 イベントトリガー定義ファイル]
6	DB 接続設定ファイル	データベースの接続に関連する情報を定義します。	[5.10 DB 接続設定ファイル]

各定義ファイルの定義内容の概要を次の図に示します。

図 5-1 定義ファイルの定義内容の概要説明



5.2 定義ファイルにおける注意事項

定義ファイルの注意事項を次の表に示します。

表 5-2 定義ファイルの注意事項

項番	制約内容	対処方法
1	NX CDMS Embedded で使用する定義ファイルは先頭 1 行目がコメント行でなければなりません。コメント行でない場合、定義エラー扱いとなる場合があります。	定義ファイルは、サンプルファイルを編集して使用してください（「5.4 定義ファイルの配置場所」を参照）。なお、ユーザーが定義ファイルを最初からエディターで作成する場合は、定義ファイルの先頭 1 行目をコメント行としてください。 (Windows のメモ帳(notepad.exe)で UTF-8 ファイルを作成した場合、無条件で BOM 付きファイルとなります。この形式の場合、一部の API ではエンコードを正しく取り扱えない問題が発生しますが、先頭 1 行目をコメント行とすることで回避できます。)
2	範囲外の数値、使用不可の文字を指定した場合、NX CDMS Embedded の起動は失敗します。	範囲外の数値、使用不可の文字は指定しないでください。
3	データ用リングバッファ定義ファイルとトリガー用リングバッファ定義ファイルの定義内容は、HX-DSM の定義ファイルの定義内容と一致させてください。	—
4	イベントトリガー定義ファイルとデータ名称定義ファイルのデータ名称は一致させてください。	—

5.3 定義ファイルの共通仕様

定義ファイルの共通仕様を次の表に示します。

表 5-3 定義ファイルの共通仕様

項番	内容
1	定義ファイルは、Windows の ini ファイルの仕様に準拠します。
2	重複した同名のセクションを定義した場合は、行番号の小さいセクションを優先します。
3	セクション内で同名の定義項目を定義した場合は、行番号の小さい定義項目を優先します。
4	コメント行にする場合は、行の先頭に";"をつけてください。 行の途中に";"をつけてもコメント扱いとなりません。
5	セクションでは、定義項目に対し"="でパラメータを指定します。
6	各定義項目に対し、使用可能文字以外の文字を指定した場合および範囲を超えて指定した場合、NX CDMS Embedded の起動は失敗します。
7	定義ファイルのファイルフォーマットについては、次の表を参照してください。

表 5-4 定義ファイルのファイルフォーマット

項番	項目	内容
1	文字コード	UTF-8
2	改行コード	CRLF

5.4 定義ファイルの配置場所

サンプルの定義ファイルの配置場所を示します。

NX_CDMS_EMB_DIR%sample%

作成した定義ファイルの配置場所を示します。

NX_CDMS_EMB_DIR%conf%

5.5 データ用リングバッファ定義ファイル

5.5.1 ファイルの説明

データ用リングバッファ定義ファイルでは、データ項目として使用するリングバッファの構造を定義します。

ファイルが存在しない場合、NX CDMS Embedded の起動は失敗します。

5.5.2 ファイル名称

データ用リングバッファ定義ファイルの名称を示します。

```
cdms_emb_dataring.conf
```

5.5.3 定義記述方法

データ用リングバッファ定義ファイルの記述形式を説明します。

(1) 記述形式

各リングバッファについて定義する「RingBuffer」セクションから構成されます。

RingBuffer セクションの定義の詳細については、「(2) RingBuffer セクション」を参照してください。

```
;Caution!! Don't delete this message.
[RingBuffer1]
BuffName=リングバッファ名称
BlockSize=ブロックサイズ
Hash=ハッシュコード

[RingBuffer2]
RingBuffer2の定義
.
.
.
[RingBufferN]
RingBufferNの定義
```

参考

- 「RingBuffer」セクションに付加する番号に対する定義順番の制限はありません。番号は連番にする必要はありません。
定義位置の場所に関わらず、「RingBuffer1、RingBuffer2、・・・、RingBufferN」の順番で定義項目を取り込みます。
-

(2) RingBuffer セクション

各リングバッファに関わる定義を行ないます。

RingBuffer セクションの名称に「1～100」の範囲内の番号（10進数表記）を付加して、セクション名としてください。

RingBuffer セクションの名称の番号が範囲外の場合、該当する RingBuffer セクションは無視します。

セクションの定義や必須の定義項目がない場合、NX CDMS Embedded の起動は失敗します。

表 5-5 RingBuffer セクション定義項目

項番	定義項目	説明	設定要否
1	BuffName	HX-DSM で定義したリングバッファ名称を記述します。 使用不可文字を指定した場合はエラーとします。 <ul style="list-style-type: none">文字数：1～31 文字（32 文字目以降は無視します）使用可能文字：英数字、"_", "-"	必須
2	BlockSize	HX-DSM で定義したリングバッファにおける 1 ブロックあたりのサイズを記述します。 <ul style="list-style-type: none">範囲：8～2147483647	必須
3	Hash	HX-DSM で定義したハッシュコードを記述します。 <ul style="list-style-type: none">範囲：0x00000000～0xFFFFFFFF（0x の後に 16 進文字列）	必須

5.6 データ項目紐付け定義ファイル

5.6.1 ファイルの説明

データ項目紐付け定義ファイルでは、共有メモリーのデータ項目と保存領域のデータ項目の対応付けを定義します。

ファイルが存在しない場合、NX CDMS Embedded の起動は失敗します。

5.6.2 ファイル名称

データ項目紐付け定義ファイルの名称を示します。

```
cdms_emb_mapped.conf
```

5.6.3 定義記述方法

データ項目紐付け定義ファイルの記述形式を説明します。

(1) 記述形式

データ項目全体について定義する「Common」セクションと、各データ項目について定義する「Data」セクションから構成されます。

各セクション定義の詳細については、「(2) Common セクション」および「(3) Data セクション」を参照してください。

```
;Caution!! Don't delete this message.
[Common]
HashCode=ハッシュコード
[Data1]
BuffName=リングバッファ名称
DataTopPtr=データの相対先頭アドレス
DataSize=データのサイズ
[Data2]
Data2の定義
.
.
.
[DataN]
DataNの定義
```

参考

- 「Data」セクションに付加する番号に対しての定義順番の制限はありません。番号は連番にする必要はありません。
定義位置の場所に関わらず、「Data1、Data2、・・・、DataN」の順番で定義項目を取り込みます。
-

(2) Common セクション

データ項目全体に関わる定義を行ないます。

セクションの定義や必須の定義項目がない場合、NX CDMS Embedded の起動は失敗します。

表 5-6 Common セクション定義項目

項番	定義項目	説明	設定要否
1	HashCode	ハッシュコードを記述します。 ハッシュコードを変更すると、保存領域はすべて削除されます。 Data セクション内を編集した場合は、必ずハッシュコードを別の値に変更してください。 <ul style="list-style-type: none"> 範囲：0x00000000～0xFFFFFFFF (0x の後に 16 進文字列) 	必須

(3) Data セクション

各データ項目に関わる定義を行いません。

Data セクションの名称に「1～900」の範囲内の番号（10 進数表記）を付加して、セクション名としてください。

Data セクション名称の番号が範囲外の場合、該当する Data セクションは無視します。

セクションを定義して必須の定義項目がない場合、NX CDMS Embedded の起動は失敗します。

表 5-7 Data セクション定義項目

項番	定義項目	説明	設定要否
1	BuffName	HX-DSM で定義したリングバッファ名称を記述します。 使用不可文字を指定した場合はエラーとします。 <ul style="list-style-type: none"> 文字数：1～31 文字（32 文字目以降は無視します） 使用可能文字：英数字、"_"、"-" 	必須
2	DataTopPtr	リングバッファのブロック内におけるデータの相対先頭アドレスを記述します。 各ブロックにおいて、先頭 8 バイトは時刻の格納領域になりますので、8 バイト目以降を指定してください。 <ul style="list-style-type: none"> 範囲：8～2147483647 	必須
3	DataSize	データのサイズを記述します。 <ul style="list-style-type: none"> 範囲：1～8 	必須

5.7 データ名称定義ファイル

5.7.1 ファイルの説明

データ名称定義ファイルでは、保存領域操作 API で指定する取得対象のデータ名称と保存領域のデータ項目との対応付けを定義します。

ファイルが存在しない場合、NX CDMS Embedded の起動は失敗します。

5.7.2 ファイル名称

データ名称定義ファイルの名称を示します。

```
cdms_emb_named.conf
```

5.7.3 定義記述方法

データ名称定義ファイルの記述形式を説明します。

(1) 記述形式

データ名称について定義する「Aliases」セクションから構成されます。

Aliases セクション定義の詳細については、「(2) Aliases セクション」を参照してください。

```
;Caution!! Don't delete this message.
[Aliases]
データ名称=Data1
データ名称=Data1
データ名称=Data2
.
.
.
データ名称=Data900
```

参考

- 1つのデータ項目に対し、複数のデータ名称を定義することができます。
-

(2) Aliases セクション

データ名称に関わる定義を行ないます。

データ名称を定義しない場合、データ名称にはデータ項目紐付け定義ファイルの Data セクション名を使用します。

表 5-8 Aliases セクション定義項目

項番	定義項目	説明	設定要否
1	データ名称	左辺： 任意のデータ名称を指定します。 データ名称は NX CDMS Embedded 内で一意となるようにしてください。 • 文字数：1～31 文字	任意

5 定義ファイル

項番	定義項目	説明	設定要否
		<ul style="list-style-type: none"> • 使用可能文字：英数字、"_"、"-" 右辺： <ul style="list-style-type: none"> データ項目紐付け定義ファイルの Data セクション名を指定します。 データ項目紐付け定義ファイルに定義していないセクション名を指定した場合、NX CDMS Embedded の起動は失敗します。 <ul style="list-style-type: none"> • 範囲：Data1～Data900 	

5.8 トリガー用リングバッファ定義ファイル

5.8.1 ファイルの説明

トリガー用リングバッファ定義ファイルでは、トリガーの通知、およびトリガーの動作状態を連携するために使用するリングバッファの構造を定義します。

ファイルが存在しない場合、NX CDMS Embedded の起動は失敗します。

5.8.2 ファイル名称

トリガー用リングバッファ定義ファイルの名称を示します。

```
cdms_emb_triggerring.conf
```

5.8.3 定義記述方法

トリガー用リングバッファ定義ファイルの記述形式を説明します。

(1) 記述形式

トリガー用リングバッファ定義ファイルは、次のセクションから構成されます。

「TriggerStatus」セクション

トリガーの動作状態を連携するために使用するリングバッファの構造を定義します。

「ErrTrigger」セクション

エラートリガーを通知するために使用するリングバッファの構造を定義します。

「EvtTrigger」セクション

イベントトリガーを通知するために使用するリングバッファの構造を定義します。

各セクションの構造を定義する詳細については、「(2) TriggerStatus セクション」、「(3) ErrTrigger セクション」、および「(4) EvtTrigger セクション」を参照してください。

```
;Caution!! Don't delete this message.
[TriggerStatus]
BuffName=TRIGGER_STAT
BlockSize=32
Hash=ハッシュコード
[ErrTrigger]
BuffName=ERR_TRIGGER
BlockSize=16
Hash=ハッシュコード
[EvtTrigger1]
BuffName=EVT_TRIGGER_1
BlockSize=16
Hash=ハッシュコード
[EvtTrigger2]
EvtTrigger2の定義
.
.
.
[EvtTriggerN]
EvtTriggerNの定義
```

参考

- 「EvtTrigger」セクションに付加する番号に対しての定義順番の制限はありません。番号は連番にする必要はありません。

定義位置の場所に関わらず、「EvtTrigger1、EvtTrigger2、・・・、EvtTriggerN」の順番で定義項目を取り込みます。

(2) TriggerStatus セクション

トリガーの動作状態を連携するために使用するリングバッファの構造に関わる定義を行ないます。

セクションの定義や必須の定義項目がない場合、NX CDMS Embedded の起動は失敗します。

表 5-9 TriggerStatus セクション定義項目

項番	定義項目	説明	設定要否
1	BuffName	リングバッファ名称は、「TRIGGER_STAT」固定です。	必須
2	BlockSize	リングバッファにおける1ブロックあたりのサイズは、「32」固定です。	必須
3	Hash	HX-DSM で定義したハッシュコードを記述します。 <ul style="list-style-type: none"> 範囲：0x00000000～0xFFFFFFFF (0xの後に16進文字列) 	必須

(3) ErrTrigger セクション

エラートリガーを通知するために使用するリングバッファの構造に関わる定義を行ないます。

セクションの定義や必須の定義項目がない場合、NX CDMS Embedded の起動は失敗します。

表 5-10 ErrTrigger セクション定義項目

項番	定義項目	説明	設定要否
1	BuffName	リングバッファ名称は、「ERR_TRIGGER」固定です。	必須
2	BlockSize	リングバッファにおける1ブロックあたりのサイズは、「16」固定です。	必須
3	Hash	HX-DSM で定義したハッシュコードを記述します。 <ul style="list-style-type: none"> 範囲：0x00000000～0xFFFFFFFF (0xの後に16進文字列) 	必須

(4) EvtTrigger セクション

イベントトリガーを通知するために使用するリングバッファの構造に関わる定義を行ないます。

EvtTrigger セクションの名称に「1～16」の範囲内の番号（10進数表記）を付加して、セクション名としてください。

EvtTrigger セクション名称の番号が範囲外の場合、該当 EvtTrigger セクションは無視します。

セクションを定義して必須の定義項目がない場合、NX CDMS Embedded の起動は失敗します。

表 5-11 EvtTrigger セクション定義項目

項番	定義項目	説明	設定要否
1	BuffName	リングバッファ名称は、「EVT_TRIGGER_N」固定です。	必須

項番	定義項目	説明	設定要否
		N は、EvtTrigger セクションの名称に付加した番号 (10 進数表記) と同じ番号を指定してください。	
2	BlockSize	リングバッファにおける 1 ブロックあたりのサイズは、「16」固定です。	必須
3	Hash	HX-DSM で定義したハッシュコードを記述します。 <ul style="list-style-type: none">範囲：0x00000000~0xFFFFFFFF (0x の後に 16 進文字列)	必須

5.9 イベントトリガー定義ファイル

5.9.1 ファイルの説明

イベントトリガー定義ファイルでは、イベントトリガーで収集するデータ項目、サンプリング周期、および収集期間を定義します。

ファイルが存在しない場合、NX CDMS Embedded の起動は失敗します。

5.9.2 ファイル名称

イベントトリガー定義ファイルの名称を示します。

```
cdms_emb_trigger.conf
```

5.9.3 定義記述方法

イベントトリガー定義ファイルの記述形式を説明します。

(1) 記述形式

各イベントトリガーについて定義する「EventTrigger」セクションから構成されます。

EventTrigger セクション定義の詳細については、「(2) EventTrigger セクション」を参照してください。

```
;Caution!! Don't delete this message.
[EventTrigger1]
TargetData=データ名称
SampCyc=サンプリング周期
BeforeTime=Before期間
AfterTime=After期間
Hash=ハッシュコード
[EventTrigger2]
EventTrigger 2の定義
:
:
[EventTrigger N]
EventTrigger Nの定義
```

参考

- 「EventTrigger」セクションに付加する番号に対しての定義順番の制限はありません。番号は連番にする必要はありません。
位置の場所に関わらず、「EventTrigger 1、EventTrigger 2、・・・、EventTrigger N」の順番で定義項目を取り込みます。
-

(2) EventTrigger セクション

各イベントトリガーに関わる定義を行ないます。

EventTrigger セクションの名称に「1～16」の範囲内の番号（10進数表記）を付加して、セクション名としてください。

EventTrigger セクション名称の番号が範囲外の場合、該当 EventTrigger セクションは無視します。

セクションを定義して必須の定義項目がない場合、NX CDMS Embedded の起動は失敗します。

表 5-12 EventTrigger セクション定義項目

項番	定義項目	説明	設定要否
1	TargetData	<p>イベントトリガーの対象としたいデータ項目を、データ名称定義ファイルで定義したデータ名称で記述します。</p> <p>存在しないデータ名称を指定した場合、NX CDMS Embedded の起動は失敗します。</p> <ul style="list-style-type: none"> 区切り文字："," データ数：1~16 1 データ名称あたりの文字数：1~31 文字 使用可能文字：英数字、"_","-" 	必須
2	SampCyc	<p>データを間引く間隔（ミリ秒単位）を記述します。</p> <ul style="list-style-type: none"> 範囲：1~2147483647 	必須
3	BeforeTime	<p>トリガーが発生した時刻を基準に遡ってデータを取得する期間（ミリ秒単位）を記述します。</p> <ul style="list-style-type: none"> 範囲：0~4000 	必須
4	AfterTime	<p>トリガーが発生した時刻を基準にデータを取得し続ける期間（ミリ秒単位）を記述します。</p> <ul style="list-style-type: none"> 範囲：0~2147483647 	必須
5	Hash	<p>ハッシュコードを記述します。ハッシュコードを変更すると、該当するイベントトリガーのデータは削除されます。</p> <p>EventTrigger セクション内を編集した場合は、必ずハッシュコードを別の値に変更してください。</p> <ul style="list-style-type: none"> 範囲：0x00000000~0xFFFFFFFF (0x の後に 16 進文字列) 	必須

5.10 DB 接続設定ファイル

5.10.1 ファイルの説明

DB 接続設定ファイルでは、データベースの接続に関する情報を定義します。

ファイルが存在しない場合、デフォルト値で動作します。

5.10.2 ファイル名称

DB 接続設定ファイルの名称を示します。

```
cdms_emb_db.conf
```

5.10.3 定義記述方法

DB 接続設定ファイルの記述形式を説明します。

(1) 記述形式

データベース接続について定義する「Common」セクションから構成されます。

Common セクション定義の詳細については、「(2) Common セクション」を参照してください。

```

;Caution!! Don't delete this message.
[Common]
Port=ポート番号
DbName=データベース名称
User=ユーザー名
Password=パスワード
DbLogPath=PostgreSQLのログフォルダのパス

```

(2) Common セクション

データベースの接続に関する定義を行いません。セクションを定義して必須の定義項目がない場合、デフォルト値で動作します。

この定義ファイルに指定したユーザー名、パスワードで、DB セットアップコマンド実行時に、データベースにユーザーを作成します。

DB セットアップコマンド実行後に、指定したユーザーのパスワードを変更した場合は、この定義ファイルのパスワードも変更してください。

定義ファイル変更の手順については、「1.3.4 定義変更フロー」を参照してください。

表 5-13 Common セクション定義項目

項番	定義項目	説明	設定要否	デフォルト値
1	Port	サーバホストでの接続用のポート番号を記述します。 <ul style="list-style-type: none"> 範囲：1～65535 	任意	5432
2	DbName	データベースの名称を記述します。 <ul style="list-style-type: none"> 範囲：63 文字以内 	任意	cdms_emb_db

項番	定義項目	説明	設定要否	デフォルト値
		使用可能文字 (半角英文字、半角数字 (0~9)、半角記号 (_ (アンダースコア))) 先頭使用可能文字 (半角英文字、半角記号 (_ (アンダースコア)))		
3	User	データベースへ接続するユーザー名を記述します。 <ul style="list-style-type: none"> 範囲：63 文字以内 使用可能文字 (半角英文字、半角数字 (0~9)、半角記号 (_ (アンダースコア))) 先頭使用可能文字 (半角英文字、半角記号 (_ (アンダースコア)))	任意	cdms_emb
4	Password	サーバがパスワードによる認証を必要とした場合に使用されるパスワードを記述します。 <ul style="list-style-type: none"> 範囲：255 文字以内 	任意	password
5	DbLogPath	PostgreSQL のログフォルダのパスを記述します。 RAS コマンドで PostgreSQL のログファイルを収集する際に使用します。 <ul style="list-style-type: none"> 範囲：259 文字以内 	任意	C:¥Program Files ¥PostgreSQL ¥11¥data ¥log

6

Windows イベントログ

この章では、Windows イベントログについて説明します。

6.1 Windows イベントログ一覧

Windows イベントログに出力されるメッセージ内容と対処方法について、次の表に示します。

表 6-1 Windows イベントログ一覧

項番	イベント ID	レベル	内容
1	100	情報	<p>【メッセージテキスト】 Started NX CDMS Embedded.</p> <p>【説明】 NX CDMS Embedded が起動しました。</p> <p>【対処方法】 —</p>
2	101	情報	<p>【メッセージテキスト】 Stopped NX CDMS Embedded.</p> <p>【説明】 NX CDMS Embedded が停止しました。</p> <p>【対処方法】 —</p>
3	102	情報	<p>【メッセージテキスト】 Warning recovered.</p> <p>【説明】 データが欠落した状態から回復しました。</p> <p>【対処方法】 —</p>
4	103	情報	<p>【メッセージテキスト】 Storage area deleted.(Area=<i>name</i>)</p> <p>【説明】 定義ファイルの変更を検知したので、保存領域を削除しました。 <i>name</i> : All または EventTriggerN (N : 1 ~ 16)</p> <p>【対処方法】 —</p>
5	600	警告	<p>【メッセージテキスト】 Warning occurred.(RingBuffer=<i>name</i>)</p> <p>【説明】 データの欠落を検知しました。 <i>name</i> : リングバッファ名称</p> <p>【対処方法】 共有メモリーにデータが書き込まれていません。共有メモリーにデータを書き込んでいるか確認してください。</p>
6	601	警告	<p>【メッセージテキスト】 Storage area full.(Area=<i>type</i>)</p> <p>【説明】 保存領域の容量が上限に達しました。</p>

項番	イベント ID	レベル	内容
			<p><i>type</i> : Error、または Event</p> <p>【対処方法】 「4.6 ロググループ削除 API」を参照の上、該当する保存領域のデータを削除してください。</p>
7	975	エラー	<p>【メッセージテキスト】 PostgreSQL Error.(Connection failed.(Port=<i>portnum</i>, Message=<i>msg</i>))</p> <p>【説明】 PostgreSQL との接続に失敗しました。 <i>portnum</i> : 接続を試行したポート番号 <i>msg</i> : 詳細メッセージ</p> <p>【対処方法】 表示されているメッセージ内容に従って対処を実施してください。対処できない場合は RAS コマンドにて障害レベルが 1 の障害情報を収集の上、問い合わせ窓口へご連絡ください。RAS コマンドは、「3.4 RAS コマンド」を参照してください。</p>
8	976	エラー	<p>【メッセージテキスト】 PostgreSQL Error.(There is not enough storage available.)</p> <p>【説明】 PostgreSQL が使用するディスクの空き容量が不足しています。</p> <p>【対処方法】 PostgreSQL のデータ保存先ディレクトリ※が存在するディスクの空き容量を確保し、再起動してください。</p>
9	977	エラー	<p>【メッセージテキスト】 PostgreSQL Error.(There is not enough memory available.)</p> <p>【説明】 PostgreSQL を実行するためのメモリーが不足しています。</p> <p>【対処方法】 本製品の実行に必要なメモリーを確保し、再起動してください。</p>
10	978	エラー	<p>【メッセージテキスト】 PostgreSQL Error.(PostgreSQL is down.)</p> <p>【説明】 PostgreSQL が停止しています。</p> <p>【対処方法】 PostgreSQL を起動し、再起動してください。</p>
11	979	エラー	<p>【メッセージテキスト】 PostgreSQL Error.(Message=<i>msg</i>)</p> <p>【説明】 PostgreSQL にてエラーが発生しました。 <i>msg</i> : 詳細メッセージ</p>

項番	イベント ID	レベル	内容
			<p>【対処方法】</p> <p>RAS コマンドにて障害レベルが 1 と 2 の障害情報を収集の上、問い合わせ窓口へご連絡ください。RAS コマンドは、「3.4 RAS コマンド」を参照してください。</p>
12	994	エラー	<p>【メッセージテキスト】</p> <p>Configuration was inconsistent with HX-DSM. (File=<i>name</i>, BuffName=<i>buff</i>, HX-DSM=<i>size</i>)</p> <p>【説明】</p> <p>定義ファイルと HX-DSM のブロックサイズが不整合です。</p> <p><i>name</i> : ファイル名称</p> <p><i>buff</i> : リングバッファ名称</p> <p><i>size</i> : HX-DSM のブロックサイズ</p> <p>【対処方法】</p> <p>該当の定義項目を修正したあと、再起動してください。</p>
13	995	エラー	<p>【メッセージテキスト】</p> <p>Internal Process exit abnormally.(エラーの詳細情報)</p> <p>【説明】</p> <p>内部プロセスの異常終了を検知しました。</p> <p>【対処方法】</p> <p>他のエラーの Windows イベントログが出力されているか確認してください。</p> <p>出力がない場合は、RAS コマンドで障害レベルが 1 の障害情報を収集したあと、問い合わせ窓口へご連絡ください。RAS コマンドは、「3.4 RAS コマンド」を参照してください。</p>
14	996	エラー	<p>【メッセージテキスト】</p> <p>HX-DSM Error.(Func=<i>name</i>, Rtn=<i>num</i>)</p> <p>【説明】</p> <p>HX-DSM でエラーが発生しました。</p> <p><i>name</i> : 関数名称</p> <p><i>num</i> : 関数の戻り値</p> <p>【対処方法】</p> <p>HX-DSM のマニュアルを参照の上、対処してください。</p>
15	997	エラー	<p>【メッセージテキスト】</p> <p>Configuration Error.(File=<i>name</i>, Section=<i>section</i>, Key=<i>key</i>)</p> <p>【説明】</p> <p>定義ファイルが誤っています。</p> <p><i>name</i> : ファイル名称</p> <p><i>section</i> : セクション名称</p> <p><i>key</i> : キー名称</p> <p>【対処方法】</p> <p>該当の定義項目を修正したあと、再起動してください。</p>
16	998	エラー	<p>【メッセージテキスト】</p> <p>API Error.(Func=<i>name</i>, Return=<i>value</i>)</p>

項番	イベント ID	レベル	内容
			<p>【説明】 保存領域操作 API でエラーが発生しました。 <i>name</i> : 関数名 <i>value</i> : リターンコード</p> <p>【対処方法】 <i>value</i> の値は、「4.8 リターンコード一覧」を参照して対処してください。なお、本メッセージは初期化 API を一度も実行していない場合、出力されません。</p>
17	999	エラー	<p>【メッセージテキスト】 Internal Error.(エラーの詳細情報)</p> <p>【説明】 内部処理で異常が発生しました。</p> <p>【対処方法】 RAS コマンドで障害レベルが 1 の障害情報を収集したあと、問い合わせ窓口へご連絡ください。 RAS コマンドは、「3.4 RAS コマンド」を参照してください。</p>

注※

「(1) 前提環境のインストール」の「(a) PostgreSQL のインストール」の手順 5 で指定したディレクトリ (凡例)

- : 対処不要

7

保存領域操作 API を使ったアプリケーション作成

この章では、保存領域操作 API を使ったアプリケーションの作成について説明します。

7.1 インクルードファイル

アプリケーション作成に必要なインクルードファイルの配置場所を次に示します。

```
NX_CDMS_EMB_DIRinc%usr%cdms_emb_utility.h  
NX_CDMS_EMB_DIRinc%sys%cdms_emb_return_codes.h  
NX_CDMS_EMB_DIRlib%cdms_emb_storage.lib
```

ヘッダファイルの内容は、変更しないでください。

7.2 DLL (Dynamic Link Library)

アプリケーションの動作に必要な DLL の配置場所を、次に示します。

`NX_CDMS_EMB_DIR%bin%cdms_emb_storage.dll`

7.3 作成環境

NX CDMS Embedded のインクルードファイルおよび DLL は、Visual Studio 2015 で作成しています。

保存領域操作 API を使用したアプリケーションは、Visual Studio 2015 以降を使用して作成してください。

7.4 保存領域操作 API 使用時の注意事項

保存領域操作 API 使用時の注意事項を次の表に示します。

表 7-1 保存領域操作 API 使用時の注意事項一覧

項番	制約内容	対処方法
1	<p>計算機の電源 OFF 操作を含めて、保存領域操作 API を使用するアプリケーションを終了する場合は、必ず終了処理 API を実行してから終了してください。</p> <p>保存領域操作 API を使用するアプリケーションを強制終了すると保存領域がロールバックされる場合があります。</p>	<p>終了処理 API を実行してから、アプリケーションを終了してください。</p>
2	<p>マルチスレッドで使用する場合、スレッド毎に初期化 API、および終了処理 API の実行が必要です。</p>	<p>スレッド毎にスレッド開始時に初期化 API、スレッド終了時に終了処理 API を実行してください。</p>
3	<p>マルチスレッドで同時に API を実行可能な最大スレッド数はすべてのアプリケーションの合計で 5 スレッドまでです。</p>	<p>マルチスレッドで同時に API を実行する場合は、すべてのアプリケーションの合計で 5 スレッドまでにしてください。</p>
4	<p>保存領域操作 API を使用するアプリケーションを常駐させる場合、NX CDMS Embedded の再起動を考慮した対応が必要です。</p>	<p>保存領域操作 API を使用するアプリケーションを常駐させる場合、NX CDMS Embedded の再起動に合わせて事前に停止してください。</p> <p>アプリケーションを事前に停止できない場合、再起動中に保存領域操作 API をコールすると E_CDMS_EMB_NOTRUNNING(-2)が返るため、エラーとなった API をリトライしてください。</p>
5	<p>NX CDMS Embedded の起動に合わせてアプリケーションを起動する場合、NX CDMS Embedded を起動してから、保存領域操作 API が使用可能になるまでのタイムラグがあります。</p> <p>NX CDMS Embedded の起動コマンド終了後にアプリケーションを起動しても、NX CDMS Embedded の初期化が終わるまでは保存領域操作 API の初期化 API がエラー (E_CDMS_EMB_NOTRUNNING -2)になります。</p>	<p>動作状態表示コマンドで初期化が完了するのを確認してから起動するか、保存領域操作 API が E_CDMS_EMB_NOTRUNNING(-2)のリターンコードの場合、エラーとなった API をリトライしてください。</p>

8

HX-DSM との連携

この章では、HX-DSM との連携について説明します。

HX-DSM の定義方法および詳細については、HX-DSM のマニュアル(HIOT-DSM-01) を参照してください。

8.1 データ用リングバッファの注意事項

データ用リングバッファの定義で、データ用リングバッファは、すべてデータ用リングバッファに書き込むアプリケーション側が、Write 権限を持つようにしてください。

また、データ用リングバッファのブロック数には、「16384」を指定してください。

8.1.1 データ用リングバッファへの書き込み

データ用リングバッファへの書き込み時、ブロックの先頭 8 バイトには時刻を格納してください。

時刻には、マシン時刻を UTC (1970/01/01 00:00:00.000 からのミリ秒) で格納してください。

データ用リングバッファに書き込む際の構造体の例を、次に示します。

```
Struct Sensor
{
    long long _timestamp; // 時刻を格納する変数
    double sensor_a;     // データを格納する変数
    double sensor_b;     // データを格納する変数
    :
    :
}
```

同時刻のデータをリングバッファに書き込んだ場合、先にリングバッファへ書き込んだデータを優先し、後にリングバッファへ書き込んだデータを無視します。

データの書き込みを行わないリングバッファを、データ用リングバッファ定義ファイルに定義しないでください。データの書き込みが行われないデータ用リングバッファが存在する場合、NX CDMS Embedded の処理が遅延し、全データが欠損となる恐れがあります。

8.2 トリガー用リングバッファの注意事項

エラートリガーを通知するために使用するリングバッファ（リングバッファ名称：ERR_TRIGGER）、およびイベントトリガーを通知するために使用するリングバッファ（リングバッファ名称：EVT_TRIGGER1～EVT_TRIGGER16）は、トリガー用リングバッファに書き込むアプリケーション側が、Write 権限を持つようにしてください。

トリガー用リングバッファの定義で、トリガーの動作状態を連携するために使用するリングバッファ（リングバッファ名称：TRIGGER_STAT）は、Widows 側が Write 権限を持つようにしてください。

また、トリガー用リングバッファのブロック数には、「4」を指定してください。

8.2.1 トリガー用リングバッファへの書き込み

ERR_TRIGGER および EVT_TRIGGER1～EVT_TRIGGER16 への書き込み時、ブロックの先頭 8 バイトには時刻を格納してください。時刻には、マシン時刻を UTC（1970/01/01 00:00:00.000 からのミリ秒）で格納してください。

また、メンバには unsigned char 型の変数を一つ定義してください。

ERR_TRIGGER や EVT_TRIGGER1～EVT_TRIGGER16 に書き込む際の構造体の例を、次に示します。

```
Struct ERR_TRIGGER
{
    long long timestamp; // 時刻を格納する変数
    unsigned char err_trigger; // トリガーのON状態を示す変数
}
```

トリガーを動作させるには、unsigned char 型変数に 0 以外を指定してください。トリガーが停止している場合だけ、動作を開始します。すでに動作中のトリガーに対し、再度、unsigned char 型変数に 0 以外を指定しても、取得期間の延長は行いません。

トリガーの停止は、取得期間の経過か保存領域の容量が上限に達したときになります。unsigned char 型変数に 0 を指定しても、動作中のトリガーは停止しません。

また、イベントトリガーは 16 個まで定義できますが、同時に動作するのは 8 個までとなります。上限を超えて通知された場合、EvtTrigger セクションの名称の番号順に 8 個まで動作し、9 個目以降は無視します。

8.2.2 トリガー用リングバッファからの読み込み

TRIGGER_STAT の読み込み時、ブロックの先頭 8 バイトは時刻として読み込んでください。

時刻には、マシン時刻を UTC（1970/01/01 00:00:00.000 からのミリ秒）で格納します。

また、メンバにはエラートリガーとイベントトリガー数分の unsigned char 型変数を格納します。

TRIGGER_STAT を読み込む際の構造体の例を、次に示します。

```
Struct TRIGGER_STAT
{
    long long timestamp; // 時刻を格納する変数
    unsigned char err_trigger; // エラートリガーの動作状態を示す変数
    unsigned char evt_trigger1; // イベントトリガー1の動作状態を示す変数
    :
    :
    unsigned char evt_trigger16; // イベントトリガー16の動作状態を示す変数
}
```

unsigned char 型変数が 0 以外の場合、そのトリガーが動作中であることを示しています。0 の場合、そのトリガーは停止していることを示しています。

また、イベントトリガーの動作状態を示す変数の数は、イベントトリガー定義ファイルに定義した数と同数です。

なお、NX CDMS Embedded では上記構造体を 1 秒周期で TRIGGER_STAT に書き込みます。

付録

付録 A 用語解説

(英字)

After 期間

トリガーが発生した時刻を基準にデータを取得し続ける期間。

Before 期間

トリガーが発生した時刻を基準に遡ってデータを取得する期間。

(あ行)

イベントログデータ

イベントトリガーによって収集するデータ。

エラーログデータ

エラートリガーによって収集するデータ。

(さ行)

サンプリング周期

データを間引く間隔。

(た行)

定常監視データ

トリガー通知に関係なく、常時収集するデータ。

トリガー

イベントトリガーとエラートリガーの総称。

(ら行)

ロググループ

トリガーによって収集した一定期間のデータの集まり。

索引

A

APIの説明 61-63, 65, 67, 69

D

DB 接続設定ファイル 90
DLL (Dynamic Link Library) 101

H

HX-DSM との連携 105

N

NX CDMS Embedded とは 2
NX CDMS Embedded の機能 23
NX CDMS Embedded の時刻の取り扱い 5
NX CDMS Embedded の動作定義ファイルの修正
18

P

PostgreSQL のインストール 7
PostgreSQL の設定 8

R

RAS 機能 35
RAS コマンド 45

W

Windows イベントログ 93
Windows イベントログ一覧 94

あ

アンインストーラ 38

い

イベントトリガー定義ファイル 88
インクルードファイル 100
インストーラ 37
インストーラ/アンインストーラ機能 37
インターフェイス 61-63, 65, 67

か

概説 1

概要 2
概略フロー 7

き

記述形式 81
起動コマンド 41
起動停止機能 26

こ

構築から運用まで 7
コマンド一覧 39
コマンド形式 41, 43, 45, 49, 51
コマンドの説明 41, 43, 45, 49, 51

さ

作成環境 102
サポート機能一覧 24

し

システム要件 6
終了処理 API 62
障害対策フロー 13
情報/状態表示機能 36
初期化 API 61
初期構築フロー 9

せ

前提条件 41, 43, 46, 51
前提ソフトウェアの更新 18

そ

ソフトウェア要件 6

て

定義記述方法 79, 81, 83, 85, 88, 90
定義ファイル 73
定義ファイル一覧 74
定義ファイルにおける注意事項 76
定義ファイルの共通仕様 77
定義ファイルの配置場所 78
定義変更フロー 15
提供コマンド一覧 40

停止コマンド 43
定常監視データ読込 API 63
データ項目紐付け機能 32
データ項目紐付け定義ファイル 81
データ収集蓄積機能 27
データ名称定義機能 34
データ名称定義ファイル 83
データ用リングバッファ定義ファイル 79
データ用リングバッファの注意事項 106
データ用リングバッファへの書き込み 106

と

動作状態表示コマンド 51
トリガー用リングバッファからの読み込み 107
トリガー用リングバッファ定義ファイル 85
トリガー用リングバッファの注意事項 107
トリガー用リングバッファへの書き込み 107

は

ハードウェア要件 6
背景 2

ふ

ファイルの説明 79, 81, 83, 85, 88, 90
ファイル名称 79, 81, 83, 85, 88, 90
プログラムプロダクト更新フロー 16
プログラムプロダクト情報表示コマンド 49

ほ

保存領域操作 API 59
保存領域操作 API 一覧 60
保存領域操作 API 使用時の注意事項 103
保存領域操作 API を使ったアプリケーション作成 99
保存領域操作機能 33

め

メッセージ一覧 41, 43, 47, 49, 51

も

戻り値 41, 43, 46, 49, 51, 61, 62, 64, 66, 67, 70

よ

用語解説 110

り

リターンコード一覧 71

ろ

ロググループ削除 API 67
ロググループ容量取得 API 69
ロググループ読込 API 65

ソフトウェア添付資料 付録1 マニュアル正誤表

本書に、「NX Context-based Data Management System for Embedded device ユーザーズガイド」(IoT-7-0010)の訂正内容を記載します。

No.	対象ページ	対象箇所	誤	正
1	P. 46	表 3-10 項番 2	障害レベル 2 の場合 : 1GiB	障害レベル 2 の場合 : 4GiB