

INDUSTRIAL
EDGE
SOLUTION
WITH
HARD REALTIME
CAPABILITIES

RT-edge

Micronet.Co,

マイクロネット
Micronet

INDUSTRIAL REALTIME EDGE COMPUTERS

User's Manual

RT-edge basic software user's manual

ユーザーズマニュアル






株式会社マイクロネット

<http://www.mnc.co.jp>

TEL: +81(0)299-90-1733

FAX: +81(0)299-90-8557

本書で使用するマークについて

	ノート：操作方法や手順等の補足情報や注釈を説明しています。
	情報：製品を利用する上で有効な豆知識となる説明をしています。
	警告：製品仕様上注意が必要な事象について説明しています。

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

INtime は、米国 TenAsys Corporation の登録商標です。

TenAsys®, INtime®, eVM® and iRMX® are registered trademarks in USA of the TenAsys Corporation.

その他、本書に記載されている会社名、商品名は、各社の商標または登録商標です。

本書の内容を無断で転載することは禁止されています。

本書の内容に関しては、予告なしに変更することがあります。あらかじめご了承ください。

製品ご利用上の注意

(製品ご利用前に必ずお読みください)

本ソフトウェアのご使用の際には本マニュアルをお読みいただくとともに、安全に対し十分に注意を払い、正しく取り扱うようお願いいたします。

機能拡張における注意事項



本製品ではご利用になる機能をパラメータ設定によるカスタマイズや、開発キットを利用し新たに機能を開発することによる拡張が可能です。パラメータ設定によりカスタマイズした場合、または開発機能を組み込み、搭載したシステムを再配布する前には必ず十分デバッグを行っていただくようお願いいたします。



本製品にて提供される機能利用により、産業通信により接続される運転中の PLC、ロボット、サーボ制御モーター等装置データにアクセスすることが可能です。制御用メッセージや、レジスタの取り扱いは接続装置に依存するため、装置のマニュアルを十分お読みいただき、ご理解いただいた上で行っていただくようお願いいたします。

ネットワーク経由により遠隔地からの機器・装置に対し制御を行う場合、ノイズその他を原因としたデータ通信異常により機器・装置側のトラブルに対応できない場合もあります。

ネットワークセキュリティ上の注意事項



本製品を組み込んだシステム運用の際、外部機器からの不正アクセス、DoS 攻撃、コンピュータウィルス等、サイバー攻撃に対し、製品に接続するアクセス先機器やシステムセキュリティを保つため、コンピュータへのアンチウィルスソフトウェアの導入、ファイヤーウォール、VPN ルーター設置等の対策をご検討ください。

システム設計上の注意事項



本製品を搭載した産業用 PC において強制電源 OFF する操作を行わないでください。強制電源 OFF 操作によりシステムデータ構造の破損や不備が生じ、不安定な挙動や誤動作が発生する可能性があります。

本ソフトウェアの適用について

本ソフトウェアをご使用いただく際、ソフトウェアにおける不具合・不備が生じたが発生した場合でも重大な事故に至らない用途であること、および故障、不具合発生時には製品外部にバックアップ、もしくはフェールセーフ機能が設けられていることを条件とさせていただきます。

マイクロネット社(以下 当社)は、当社製品の品質、性能、安全に係る一切の責任(債務不履行責任、瑕疵担保責任、品質保証責任、不法行為責任、製造物責任を含むがそれらに限定されません)を負わないものとします。

目次

1. 概要	6
1.1. RT-edge の概要	6
1.1.1. RT-edge とは	6
1.1.2. サービスコンテナ	7
1.1.3. RTCD	9
1.1.4. RT-edge API	9
1.2. 開発から運用までの流れ	10
1.3. 環境構築の流れ	11
2. 利用可能なコンテナ	12
2.1. AI 画像解析	13
2.1.1. AI 画像解析	13
2.2. C2C インターフェース(W)	13
2.2.1. MC	13
2.2.2. FINS	13
2.3. C2C インターフェース(R)	13
2.3.1. CC-Link IE Control	13
2.3.2. EtherNet/IP	13
2.3.3. FL-net	13
2.4. SCADA/HMI	14
2.4.1. DDE サーバー	14
2.5. 外部通信インターフェース	14
2.5.1. MQTT	14
2.5.2. FTPS	14
2.5.3. TCP/IP	14
2.6. WEB サーバー	14
2.6.1. WEB サーバー	14
2.7. 挙動監視	15
2.7.1. 挙動監視	15
2.8. I-I/O インターフェース(W)	15
2.8.1. Modbus RTU	15
2.9. I-I/O インターフェース(R)	15
2.9.1. EtherCAT	15
2.9.2. RS-232C	15
2.9.3. Modbus TCP	15
2.10. ソフトウェア PLC	16
2.10.1. INplc	16
2.11. 仮想マイコン	16
2.11.1. 仮想マイコン	16
2.12. SDAT	16
2.12.1. SDAT	16
2.13. 保守機能	17
2.13.1. 保守機能	17
3. 仕様	18
3.1. 動作環境	18

3.2. 開発環境	18
3.3. 基本仕様	19
3.4. RT-edge アーキテクチャ	21
3.5. RT-edge オブジェクト	22
3.5.1. ①タグ(EgTag)オブジェクト	23
3.5.2. ②データセット(EgDataset)オブジェクト	26
3.5.3. ③コレクタ(EgCollector)オブジェクト	26
3.5.4. ④メールボックス(EgMailbox)オブジェクト	27
3.5.5. ⑤TagRef/⑥CollectorRef オブジェクト	27
3.6. RT-edge フレームワーク	28
3.6.1. RT-edge フレームワーク構造図	30
3.6.2. EgService 間メッセージ通信	31
3.6.3. RT-edge フレームワークで登録する EgTag 情報(サービスコンテナインジケータ)	31
3.6.4. タグトリガー機能	32
3.7. ECI ファイル	33
3.7.1. ファイル定義	34
3.7.2. 基本構造	35
3.7.3. 構成要素	38
3.7.4. EgBoot 固有設定	46
4. ファイル情報	47
4.1. 実行環境	47
4.2. 開発環境	48
5. 開発環境・実行環境の構築	49
5.1. 環境構築の流れ	49
5.2. 開発環境の構築	50
5.2.1. 必要ソフトウェアのインストール	50
5.2.2. 実行環境/開発環境用ファイルの配置	50
5.2.3. Visual Studio ランタイムライブラリのインストール	51
5.2.4. サンプル/サービス作成用テンプレートのインストール	51
5.3. 実行環境の構築	52
5.3.1. 必要ソフトウェアのインストール	52
5.3.2. 実行環境/開発環境用ファイルの配置	52
5.3.3. Visual Studio ランタイムライブラリのインストール	53
5.3.4. 不要ファイルの削除	53
5.4. 開発環境・実行環境の削除	54
5.5. 製品動作設定(実行環境設定)	54
5.5.1. 必要ソフトウェアのインストール	54
5.5.2. 実行ファイルの配置	54
5.5.3. ECI ファイルの編集	55
5.5.4. RT-edge 起動スクリプトについて	57
5.5.5. RT-edge システム動作確認	65
5.5.6. RT-edge システム終了手順	65
5.6. サンプルプログラムを使つての動作確認	67
5.7. 製品動作設定(開発環境設定)	67
6. トラブルシューティング	68
6.1. エラーが発生して起動できない(INtime サービスコンテナ)	68
6.2. エラーが発生して起動できない(.NET サービスコンテナ)	69
6.3. タグの生成に失敗する	69

7. 付録	70
7.1. RT-edge オブジェクトブラウザ	70
7.1.1. Tags 表示タブ	71
7.1.2. Containers 表示タブ	71
7.1.3. Collectors 表示タブ	71
7.1.4. Datasets 表示タブ	72
7.1.5. Mailboxes 表示タブ	72
7.1.6. 更新周期設定	72
7.1.7. CSV 出力	73
7.2. EgLog ログサービスコンテナ	74
7.2.1. 機能	75
7.2.2. サービスコンテナパラメータ	76
7.2.3. サービスコンテナ動作設定	77
7.3. EgTime 時刻サービスコンテナ	78
7.3.1. 機能	78
7.3.2. サービスコンテナパラメータ	79
7.3.3. サービスコンテナ動作設定	80
7.4. EgShDown サービスコンテナ	81
7.4.1. 機能	81
7.4.2. サービスコンテナの実行	81

はじめに

この度は産業用リアルタイム・エッジソフトウェアプラットフォーム「RT-edge」をご利用いただきまして有難うございます。本マニュアルは、本ソフトウェアのご使用方法についてご理解いただくためのマニュアルです。ご使用の前に本マニュアルをお読みいただき、製品の構造、機能等十分にご理解の上、正しくご使用いただくようお願いいたします。

用語解説

本ドキュメントにおいて使用される用語・略称について説明します：

表 1 用語集

用語	説明
RT-edge	エッジコンピューティングを軸とする IT の情報処理と、FA における装置・機器の制御を融合し、密度の高い高頻度データ利用を可能とするソフトウェアプラットフォームです。FA で要求されるハードリアルタイム制御を組み込むことで、情報処理と機器・装置制御を可能とするエッジコントローラを構成することができます。
RT-edge 基本ソフトウェア	RT-edge 機能の核となる機能・ライブラリを実装するパッケージソフトウェア製品です。
IoT ゲートウェイ	IoT において、端末とインターネットを介した遠隔サーバー(クラウド)がデータのやりとりをする際、中継する役割を担う機能です。サーバーや送信経路であるインターネット負荷の軽減をします。
IT システム	オンプレミスもしくはクラウドを活用した業務システムやアプリケーションです。
INtime	INtime for Windows: Windows と協調動作可能なリアルタイムカーネル拡張ソフトウェアです。INtime Distributed RTOS(dRTOS): Windows OS を必要とせず、スタンドアロンで動作するリアルタイム OS です。
RTA	RealTime Application: リアルタイムアプリケーションの略称。INtime 上で動作するロードブルプロセスの拡張子です。INtime 上で動作するロードブルアプリケーションは、RTA という拡張子を持ちます。
RSL	Realtime Shared Library: リアルタイム共有ライブラリの略称。INtime 上でアプリケーションがロード可能なライブラリです。Windows 上で使用される DLL(Dynamic Link Library)のようなものです。RTA から使用されるライブラリインタフェース等は、こちらを使用して作成することができます。
API	Application Programming Interface: アプリケーションプログラミングインタフェースの略称。RT-edge ではデバイスへのアクセスインタフェースとして API ライブラリを提供しています。
NTX	INtime's Windows NT extension API: INtime 用 Windows NT 拡張 API の略称。NTX 関数は Windows プログラムが INtime リアルタイム環境上で実行するリアルタイムプログラムと通信を可能とする関数セットです。
OPC	主に産業オートメーション分野においてデータ交換を目的とした相互運用標準規格です。
OPC UA	OPC UA(OPC Unified Architecture の略) 異なるプラットフォーム間のデータ交換を可能とした信頼性のある産業用通信データ交換標準。インダストリー4.0 の RAMI モデルに採用された規格です。
エッジアプリケーション	RT-edge 内コンテナにより集積されたデータ(RTCD)を活用、処理実行するソフトウェアです。
エッジコンピューティング	RT-edge 内で稼働する制御コンテナソフトウェアにより装置・機器から収集した高密度なデータをリアルタイムに収集、分析、フィードバックします。
オンプレミス	サーバーやソフトウェア等の情報システム、アプリケーション等のソフトウェアを管理する施設内に設置して運用することです。
クラウド	サーバーやストレージ等のインフラやソフトウェアを必要とせず、必要な IT リソースが、インターネットを通じてオンデマンドで得られる形態、サービスです。
産業用 PC	高信頼性、耐環境性、長期供給等の特徴をもつ産業用途の PC。
データ収集	診断、分析を行う対象となるデータを集積する処理です。
データ加工	集積されたデータを利用しやすい形に変更する処理です。
産業機器通信インタフェース	各種フィールドバス経由で機器、装置との通信、もしくは直接入出力デバイスの制御を行うインタフェースです。本インタフェースを介し、センサー値の参照やアクチュエータ制御が可能です。
サービスコンテナ/EgService	RT-edge システムを構成するコンテナ機能内のインタフェースや、プロセス(rta/exe)を指します。

用語	説明
タグ/EgTag	瞬時値データ値 1 つを示すオブジェクトです。ユニーク名とグローバルなスコープを持ち、全ての EgService から読み書きが許されたオブジェクトです。タグは生成時にデータ型が確定され変更はできません。
リンクタグ	同一名称のタグを重複生成した場合に自動的に別名称で生成されるタグを指します。通常のタグと同様、グローバルなスコープを持ち、全ての EgService から読み書きが許されたオブジェクトです。一つのタグに対し、異なるプロパティ情報を定義したい場合に使用します。
データセット/EgDataset	タグ 1 つ以上の組み合わせでデータ並び順(データ構造)を定義する名前付きオブジェクトです。
コレクタ/EgCollector	データセットに定義されたデータ構造に従って、同時刻のバイナリデータ列で生成し、データレコードとしてメールボックスに送信する機能です。
メールボックス/EgMailBox	時系列なデータセット、または時系列メッセージを FIFO で蓄えることができ、また受信イベントとして処理できるオブジェクトです。
タグ参照/TagRef	タグの参照として使用するオブジェクトです。タグの名前を保持し値は保持しません。RT-edge コンテナ設定情報(ECI)ファイルでデータセットの収集用タグとして定義することや、サービスコンテナ内のオブジェクトとして定義することでサービスコンテナのメンバ変数として使用することができます。
コレクタ参照/CollectorRef	コレクタの参照として使用するオブジェクトです。コレクタの名前を保持しそれ以外のオブジェクトは保持しません。RT-edge コンテナ設定情報(ECI)ファイルでサービスコンテナ内のオブジェクトとして定義することでサービスコンテナ内のメンバ変数として使用することができます。
タグトリガー/TagTrigger	タグトリガーとは特定のタグの値が変化した場合に、サービス側でメッセージ通知を受けることができる機能です。サービスの ECI ファイルでタグトリガーとして登録したいタグ名を記載することで、タグ値の変更通知を受け取ることができます。
メッセージ	メッセージとは、一般的にある人や機器、ソフトウェアなどから、別の主体へ伝えようとする内容や、それを一定の形式のデータとして表現したものをさします。 RT-edge では、メールボックスで扱われる 1 レコード分のデータ、またはサービス間のコマンド、応答の電文のことを意味します。
RT-edge フレームワーク	フレームワークとは、一般的にアプリケーション開発時の土台として機能させるソフトウェアや仕組みのことをさします。 RT-edge では、アプリケーションが API を組み合わせて実装する中で、よく利用する処理についてマクロ化、自動化したもので RT-edge コンテナ設定情報(ECI)ファイルの記述により自動処理させることができます。
RT-edge コンテナ設定情報(ECI)	サービスコンテナが RTCD に展開する入出力データ定義の他、RT-edge フレームワークが、オブジェクト生成やサービスコンテナ等自動処理するための定義設定情報(XML 型式)です。
入力	RT-edge システムを中心に見た場合、外部の情報を RT-edge システムへ取り込む方向性のデータの流れを意味します。
出力	RT-edge システムを中心に見た場合、RT-edge システムが持つデータを外部に書き出す方向性のデータの流れを意味します。
RTCD	Realtime Common Data の略称。RT-edge システム上で最もベースとなる共有データ構造機能です。RT-edge サービスコンテナの生成する入出力データタグのコレクション。例えば、センサーや装置から収集したデータをアプリケーション間で受け渡しを行う場合、またはアプリケーション間でメッセージのやり取りを行う場合等、アプリケーション間でデータの受け渡しを行うケースで利用されます。 RTCD には入出力データの瞬時値が格納されるだけでなく、入出力データを処理した加工データや、アプリケーションや各サービスコンテナが独自に提供したデータも格納されます。

関連資料

表 2 RT-edge 関連資料

名称	ファイル名	内容
RT-edge API リファレンス	DOCRTEEDGEAPI.pdf	RT-edge API の使用方法が記載されています。
RT-edge コンテナ作成マニュアル	DOCRTEEDGESESRV.pdf	サービスコンテナの構造、サンプルプロジェクトを利用した作成方法等について記載されています。
インストール手順書	インストール手順書.pdf	RT-edge 実行/開発環境のインストール手順が記載されています。

1. 概要

1.1. RT-edge の概要

1.1.1. RT-edge とは

RT-edge は、リアルタイム OS「INtime®」やソフトウェア PLC「INplc」を利用するユーザーに向けた開発支援フレームワークです。その主な特徴としては以下の点が挙げられます：

① 効率的な開発プロセス：

RT-edge は、産業システムや産業装置の制御ソフトウェア開発を効率化するために設計されています。

② コンテナ技術の活用：

独立したソフトウェアモジュールである「コンテナ」を用いることで、開発プロセスを単純化し、再利用性を高めます。

③ シームレスなデータ統合：

システム内のデータをタグ機能により一元管理し、情報処理と制御処理の統合を容易に行えるようにします。

1.1.2. サービスコンテナ

RT-edge の処理ターゲットは、エッジコンピューティングを軸とした IT 情報処理(IT-Process)と、ミリ秒精度のハードリアルタイム性を要求される FA 制御(FA-Control)に分類され、ターゲットの機能に特化した専門処理サービスをコンテナ(サービスコンテナ)と呼びます。

IT 情報処理ターゲットは上位層にあり、主に外部システムからの要求指示の受付や、外部システムへのデータ公開、通信等を担う要素となります。IT 情報処理サービスコンテナは、制御システムのコンソール画面や外部システムから WEB ブラウザ経由でのアクセス機能、制御データ情報を外部クラウドストレージに保存する機能等、上位システムとの接続・インターフェースを提供します。

一方、FA 制御ターゲットは下位層に位置し、主に通信やハードウェアへの直接 I/O 入出力等により装置・機器制御を担う要素です。FA 制御サービスコンテナは、産業用フィールドバスやコントローラ通信プロトコルによるロボット制御、計測機器からのデータロギング、デジタルパルス出力等、装置・機器へのアクセスを提供します。

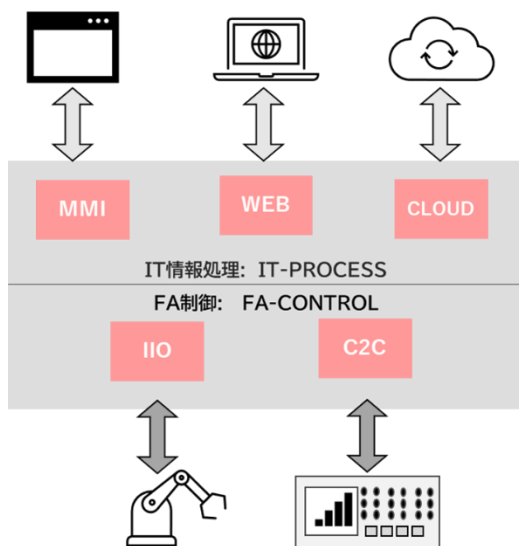


図 1 ターゲットとコンテナ

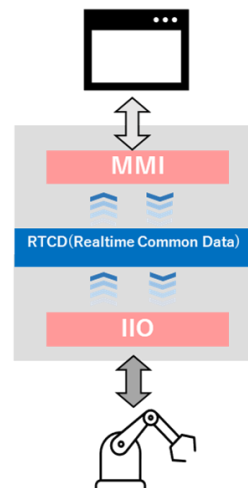


図 2.コンテナの役割



サービスコンテナ例

WEB: IT 情報処理ターゲット内には、IIS(Internet Information Service)を介し、制御情報をインターネット上に公開する WEB サービスコンテナ

IIO: ロボットアーム制御に特化した産業 I/O サービスコンテナ

サービスコンテナはターゲットに特化した入出力データをシステム内でグローバルにアクセス可能なタグ情報としてリンクし、このタグ情報のコレクションを RTCD(Realtime Common Data)とよばれる共有データ構造に展開します。

サービスコンテナは、タグ情報コレクションや、動作・挙動を決定するパラメータ設定と、ター

ゲット処理に特化した一つ以上の実行処理の集合体です:

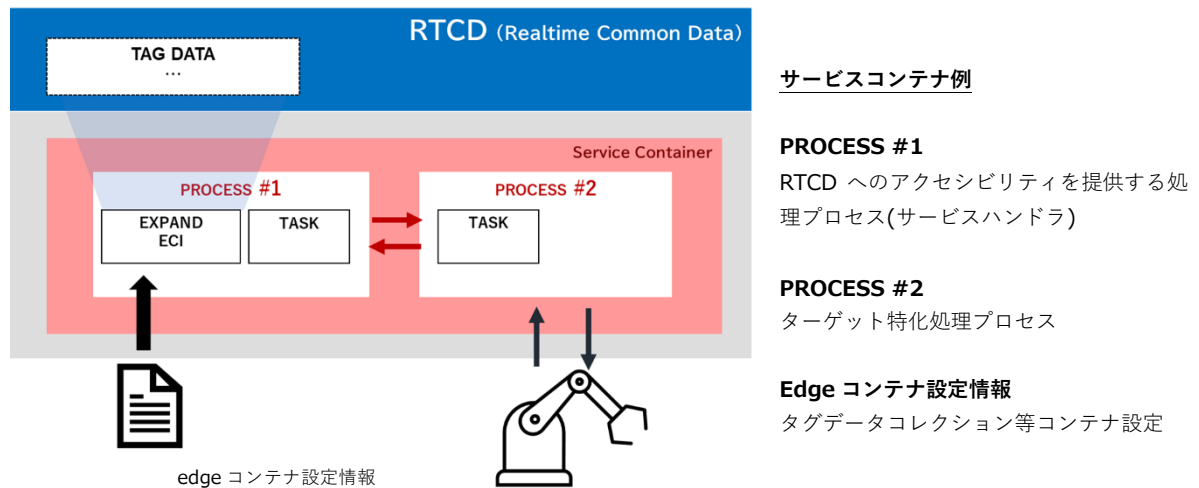


図 3 サービスコンテナの構造

サービスコンテナは標準で利用可能なサービスコンテナ（標準コンテナ）の他に、独自の機能・処理を組み込んだサービスコンテナ（カスタムコンテナ）を作成することもできます。カスタムコンテナの作成には edge API を使用します。

1.1.3. RTCD

RT-edge の IT 情報処理(IT-Process)と、FA 制御(FA-Control)を繋ぐインターフェースとして、RTCD(Realtime Common Data)とよばれる共有データ構造があります。

RTCD には、タグ情報と紐づいたデータや、タグ情報の集合体を 1 レコードしたコレクション、サービスコンテナ間通信用のデータエリア等が含まれており、これらの情報は、全てのサービスコンテナに公開され、全てのサービスコンテナからアクセスする事が可能です。

例えば、センサーや装置から収集したデータをサービスコンテナ間で受け渡しを行う場合、データとタグを紐づけ、タグ情報として RTCD へ展開する事によって、サービスコンテナ間は特別な手続きを必要とする事なく、タグ情報にあるデータを読み書きする事が可能になります。

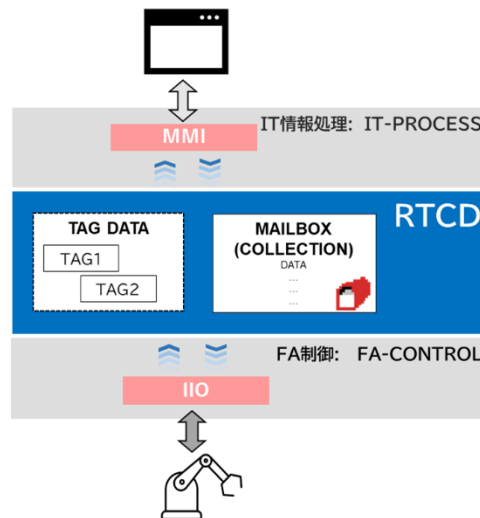


図 4 RTCD とコンテナ

1.1.4. RT-edge API

RT-edge の機能を提供する API です。RTCD に展開されたタグ情報等へ edge API を使用してアクセスすることができます。また、この API を使用して独自のサービスコンテナ(カスタムコンテナ)を作成する事ができます。

API の詳細は「RT-edge API リファレンスマニュアル」を参照してください。

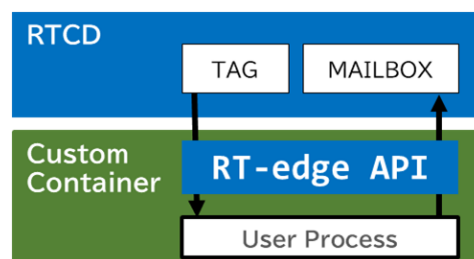
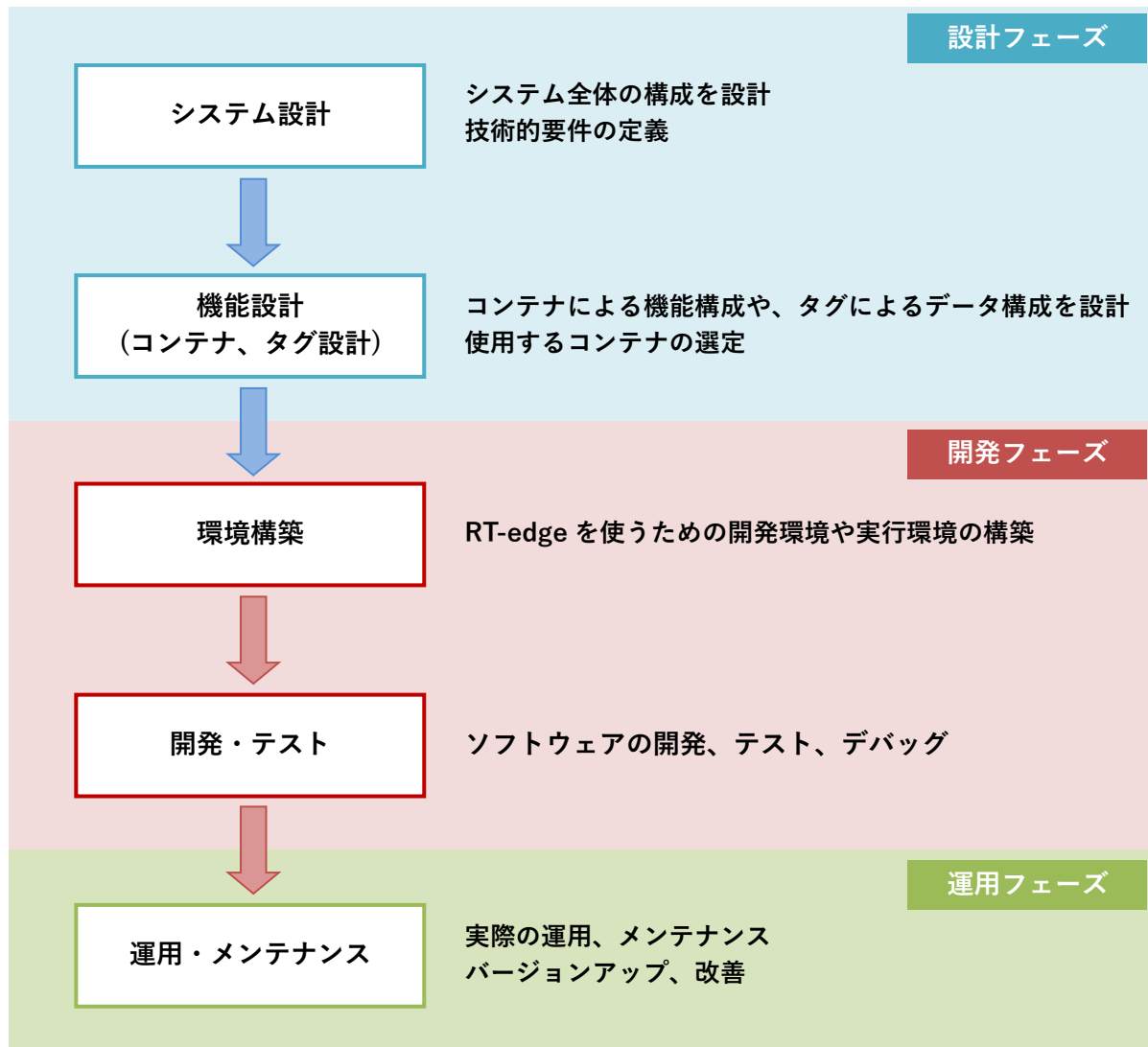


図 5 RT-edge API とカスタムコンテナ

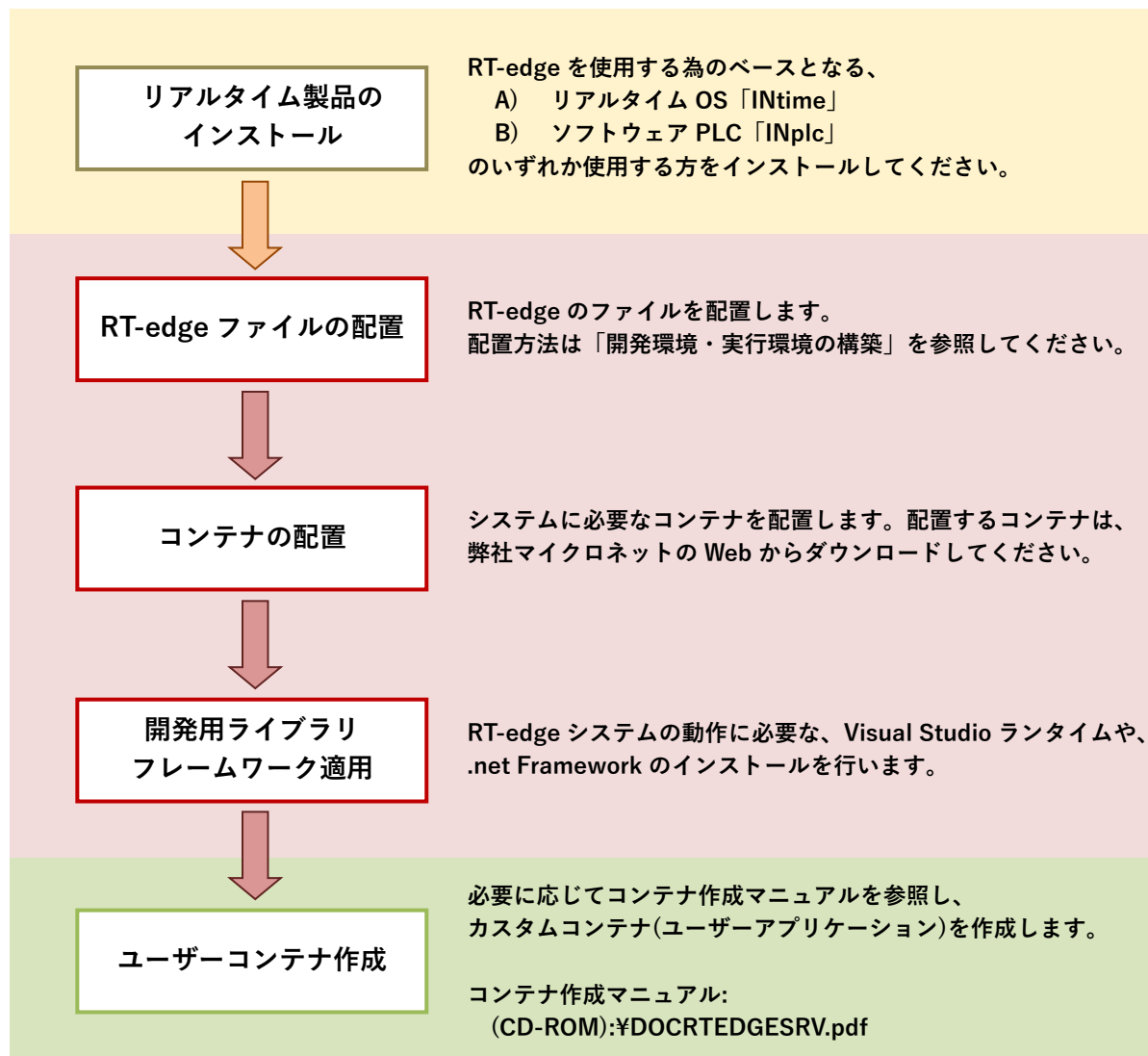
1.2. 開発から運用までの流れ

RT-edge による、開発から運用までの流れは次のようになります。



1.3. 環境構築の流れ

RT-edge を使った環境構築の流れは次のようになります。



2. 利用可能なコンテナ

利用できるコンテナは、以下の表 3 利用可能なコンテナ一覧表の内、利用可に✓が入っている項目になります。

表 3 利用可能なコンテナ一覧表

No.	コンテナ名称	略称	機能	利用可	ドキュメント情報
1	AI 画像解析	AIP	AI 画像解析	-	-
2	C2C インターフェース(W)	C2C(W)	MC	-	(C2C インターフェース)zip ファイル¥ DOCRTEDEGESRV_C2C.pdf
3			FINS	-	
4	C2C インターフェース(R)	C2C(R)	CC-Link IE Control	✓	
5			EtherNet/IP	✓	
6			FL-net	✓	
7	SCADA/HMI	SH	DDE サーバー	✓	(SCADA/HMI)zip ファイル¥ DOCRTEDEGESRV_DDESVR.pdf
8	外部通信インターフェース	EXTIF	MQTT	✓	(外部通信インターフェース)zip ファイル¥ DOCRTEDEGESRV_EXTIF.pdf
9			FTPS	✓	
10			TCP/IP	✓	
11	WEB サーバー	WEB	WEB サーバー	✓	(WEB サーバー)zip ファイル¥ DOCRTEDEGESRV_Web.pdf
12	挙動監視	BM	挙動監視	✓	(挙動監視)zip ファイル¥ DOCRTEDEGESRV_BM.pdf
13	I-I/O インターフェース(W)	IIO(W)	Modbus RTU	-	(I-I/O インターフェース)zip ファイル¥ DOCRTEDEGESRV_IIO.pdf
14	I-I/O インターフェース(R)	IIO(R)	EtherCAT	✓	
15			RS-232C	-	
16			Modbus TCP	-	
17	ソフトウェア PLC	INPLC	INplc	✓	(ソフトウェア PLC)zip ファイル¥ DOCRTEDEGESRV_INPLC.pdf
18	仮想マイコン	VM	仮想マイコン	✓	(仮想マイコン)zip ファイル¥ DOCRTEDEGESRV_VM.pdf
19	SDAT	SDAT	SDAT	✓	(SDAT)zip ファイル¥ DOCRTEDEGESRV_SDAT.pdf
20	保守機能	MAINT	保守機能	✓	(保守機能)zip ファイル¥ DOCRTEDEGESRV_BKUP.pdf DOCRTEDEGESRV_DIAG.pdf DOCRTEDEGESRV_GATHER.pdf DOCRTEDEGESRV_RESET.pdf

2.1. AI 画像解析

2.1.1. AI 画像解析

将来対応予定

2.2. C2C インターフェース(W)

2.2.1. MC

将来対応予定

2.2.2. FINS

将来対応予定

2.3. C2C インターフェース(R)

2.3.1. CC-Link IE Control

概要

本コンテナの CC-Link IE Control は、CC-Link IE コントローラネットワーク上のデータをタグでやりとりできます。

使用条件

別途、RSI-CCIE 製品が必要です。詳しくは付属ドキュメントをご参照ください。RSI-CCIE 製品については、以下の URL をご参照ください。

<https://www.mnc.co.jp/INtime/CC-linkIE.pdf>

2.3.2. EtherNet/IP

概要

本コンテナの EtherNet/IP は、CIP 通信にてコントローラ間でタグデータをやりとりできます。

使用条件

別途、Hilscher 製 CIFS 50-RE 通信ボード製品、および、Hilscher 製 CIFS INtime Driver が必要です。詳しくは付属ドキュメントをご参照ください。

2.3.3. FL-net

概要

本コンテナの FL-net は、FL-net 通信を使用したコントローラ間通信機能を持ち、FL-net ネットワーク上のコントローラとタグデータのやりとりができます。

使用条件

別途、RSI-040 製品が必要です。詳しくは付属ドキュメントをご参照ください。RSI-040 製品については、以下の URL をご参照ください。

<https://www.mnc.co.jp/INtime/FLnet/FLnet.pdf>

2.4. SCADA/HMI

2.4.1. DDE サーバー

概要

本コンテナの DDE サーバーは、DDE 通信におけるサーバー機能を持ち、DDE クライアント機能を有するユーザーアプリケーション(DDE クライアント)とタグデータをやりとりできます。

使用条件

ありません。

2.5. 外部通信インターフェース

2.5.1. MQTT

概要

本コンテナの MQTT は、タグデータを、リモート機器から参照できるようにする MQTT プロトコル通信のクライアント機能を提供します。

使用条件

Windows で利用する Ethernet が必要です。詳しくは付属ドキュメントをご参照ください。

2.5.2. FTPS

概要

本コンテナの FTPS は、自身が FTP クライアントとなり、FTPS をサポートする FTP サーバーとの接続を可能とし、設定ファイル・データファイル等を転送、受信を可能にします。

使用条件

Windows で利用する Ethernet が必要です。詳しくは付属ドキュメントをご参照ください。

2.5.3. TCP/IP

概要

本コンテナの TCP/IP は、自身が TCP/IP クライアントとなり、外部 TCP/IP サーバアプリケーションとの接続を可能とし、タグデータのやりとりを可能にします。

使用条件

Windows で利用する Ethernet が必要です。詳しくは付属ドキュメントをご参照ください。

2.6. WEB サーバー

2.6.1. WEB サーバー

概要

本コンテナの WEB サーバーは、WEB サーバー用のライブラリを用いて、ブラウザ経由で

の GUI が提供できます。

使用条件

Windows で利用する Ethernet が必要なことや、IIS などの設定が必要です。詳しくは付属ドキュメントをご参照ください。

2.7. 挙動監視

2.7.1. 挙動監視

概要

本コンテナの挙動監視は、USB カメラやイーサネットカメラから映像を取得し、映像データや画像データを逐次、またはトリガーのタイミングでファイルへ記録します。

使用条件

別途 USB カメラやイーサネットカメラが必要です。

2.8. I-I/O インターフェース(W)

2.8.1. Modbus RTU

将来対応予定

2.9. I-I/O インターフェース(R)

2.9.1. EtherCAT

概要

本コンテナの EtherCAT は、EtherCAT マスタの機能を有しており、EtherCAT スレーブからのデータ収集、またはタグデータを EtherCAT スレーブへ出力します。

使用条件

別途、RSI-ECAT-Master 製品が必要です。RSI-ECAT-Master 製品については、以下の URL をご参照ください。

<https://www.mnc.co.jp/ethercat/RSIECAT/index.htm>

2.9.2. RS-232C

将来対応予定

2.9.3. Modbus TCP

将来対応予定

2.10. ソフトウェア PLC

2.10.1. INplc

概要

本コンテナの INplc は、INplc が持つ M/I/Q エリアをタグにして、INplc からタグに対して入出力ができるようにします。

使用条件

別途、INplc 製品が必要です。INplc 製品については、以下の URL をご参照ください。

<https://www.mnc.co.jp/INplc/index.htm>

2.11. 仮想マイコン

2.11.1. 仮想マイコン

概要

本コンテナの仮想マイコンは、タグに対しての入出力や設定をレジスタのように扱え、リアルタイムアプリケーションとして実行することができます。

使用条件

次の内いずれかの条件を満たしている場合、仮想マイコンライブラリを使ってプログラムを開発することができます。

- (1) INtime 開発キット(INtime for Windows)をお持ちの場合
- (2) トレーサブルコントローラ開発キットをお持ちの場合
- (3) 仮想マイコン製品をお持ちの場合

仮想マイコン製品については、以下の URL をご参照ください。

<https://www.mnc.co.jp/RT-EDGE/container/vmc.htm>

2.12. SDAT

2.12.1. SDAT

概要

本コンテナの SDAT は、複数のタグデータを定周期で蓄積することができます。

使用条件

ありません。

2.13. 保守機能

2.13.1. 保守機能

概要

本コンテナの保守機能は、早期復旧、日々のメンテナンスやバックアップ、運転状況・障害発生状況の記録保存等、システム運用中のさまざまな保守フェーズで必要となる機能が利用できます。

使用条件

ありません。

3. 仕様

3.1. 動作環境

Windows:

- Windows 10 以降

INtime: (INtime を組み込む場合以下インストール必須)

- INtime version 6.4.21125.1 以降
- トレーサブルコントローラ 6.4.21125.1 以降

その他:

- .NET Framework 4.6 対応



動作可能な Windows のバージョン、エディションは、INtime のバージョンにより異なります。詳細は INtime 製品ドキュメントを参照ください。

3.2. 開発環境

RSL/Windows dll をリンクしたサービスコンテナを開発する環境**(コンパイラ・リンカ環境)**

- Visual Studio 2017



INtime for Windows 上で動作するサービスコンテナを作成する場合は、上記 Visual Studio をサポートするバージョンの INtime SDK 開発キットが必要となります。

3.3. 基本仕様

RT-edge の基本仕様は以下の表のとおりです：

表 4 基本仕様

大項目	項目	説明
タグ (EgTag)	最大数	10,000 個 ※設定変更可
	名称最大長	48 バイト(半角英数)
	コメントプロパティ最大長	48 バイト(半角英数+全角)
	アドレスプロパティ最大長	48 バイト(半角英数)
	対応データ型	Boolean : true/false SByte : 符号付き 8bit 整数 Byte : 符号なし 8bit 整数 Int16 : 符号付き 16bit 整数 UInt16 : 符号なし 16bit 整数 Int32 : 符号付き 32bit 整数 UInt32 : 符号なし 32bit 整数 Int64 : 符号付き 64bit 整数 UInt64 : 符号なし 64bit 整数 Float : 単精度実数(32bit) Double : 倍精度実数(64bit) String : 文字列型※ ¹ Segment: メモリブロック※ ¹
	サービスコンテナ (EgService)	動作プラットフォーム(いずれか一方) (1) pure Windows(.exe) (2) INtime(.rta) + NTX(.exe)
	最大数	32 個
	名称最大長(プログラムファイル名)	8 バイト(半角英数拡張子含まない)
	ファイルパスプロパティ最大長	260 バイト
	データセット (EgDataset)	
データセット (EgDataset)	最大数	512 個
	名称最大長	32 バイト(半角英数)
	タグリンク最大数	2048 個
RT-edge フレー ムワーク	入力タグ参照(TagRefIN)最大数	1024 個
	出力タグ参照(TagRefOUT)最大数	1024 個
	コレクタ定義最大数	16 個
	コレクタ名称最大長	8 バイト(半角英数)
	コレクタスレッドプライオリティ指定	可能、256 段階
	メールボックス機能	対応、FIFO
	メールボックス定義最大数	16 個

大項目	項目	説明
	メールボックス名称最大長	8 バイト(半角英数)
	メールボックス メッセージ最大データサイズ (最大レコードデータサイズ)	10,352 バイト※ ²
	メールボックスメッセージ保持件数 (最大レコード数)	4,096 レコード※ ²
	コンテナ間メッセージ通信	対応、FIFO
	コンテナ間メッセージ最大データサイズ (最大レコードデータサイズ)	10,352 バイト (デフォルト 2,048 バイト)
	コンテナ間メッセージ保持件数 (最大レコード数)	4,096 レコード (デフォルト 1,024 バイト)
	タグトリガー 最大数	128 タグ(デフォルト)
	タグデータアクセス時間	約 3μsec ※ ⁴
システム	メモリ使用量	約 2MB ※ ³
	タグデータアクセス時間	約 3μsec ※ ⁴



- ※1 Segment タイプのタグは最大 256 個、セグメント合計サイズは 1MB
- ※2 詳細は「3.7.3.20 レコードデータサイズ・レコード数に関して」を参照
- ※3 1 サービスコンテナあたりのメモリ使用量の参考値
- ※4 1 タグあたりのアクセス時間

3.4. RT-edge アーキテクチャ

RT-edge は「RTCD」「edge API」「サービスコンテナ」から構成するエッジコンピューティングプラットフォームです。RTCD はデータバッファリング機能やメッセージ通信機能を搭載しております。外部アプリケーション用インターフェース「edge API」を利用する事で独自のアプリケーションが実装可能です。

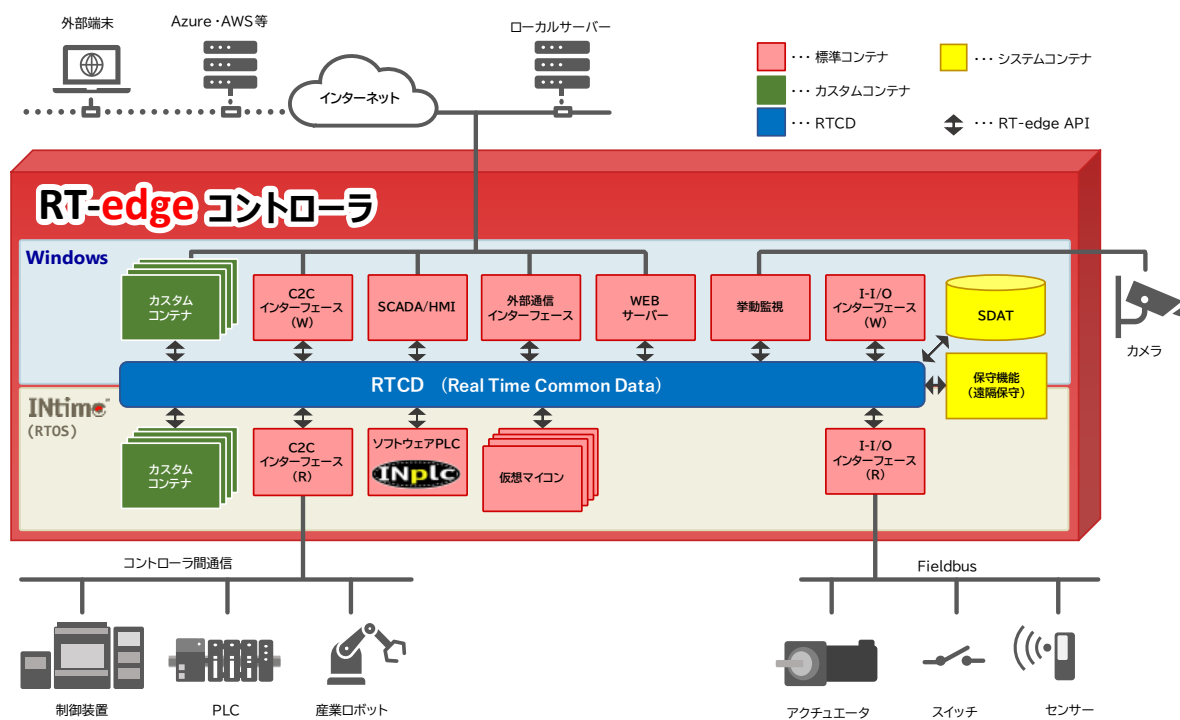


図 6 全体概要図



INtime for Windows がインストールされていない環境は、Windows 側に RTCD が生成されます。

3.5. RT-edge オブジェクト

RT-edge システムでは次のオブジェクトの概念があります：

表 5 RT-edge オブジェクト

No	通称	オブジェクト名
①	タグ	EgTag オブジェクト
②	データセット	EgDataset オブジェクト
③	コレクタ	EgCollector オブジェクト
④	メールボックス	EgMailbox オブジェクト
⑤	タグ参照	TagRef オブジェクト
⑥	コレクタ参照	CollectorRef オブジェクト

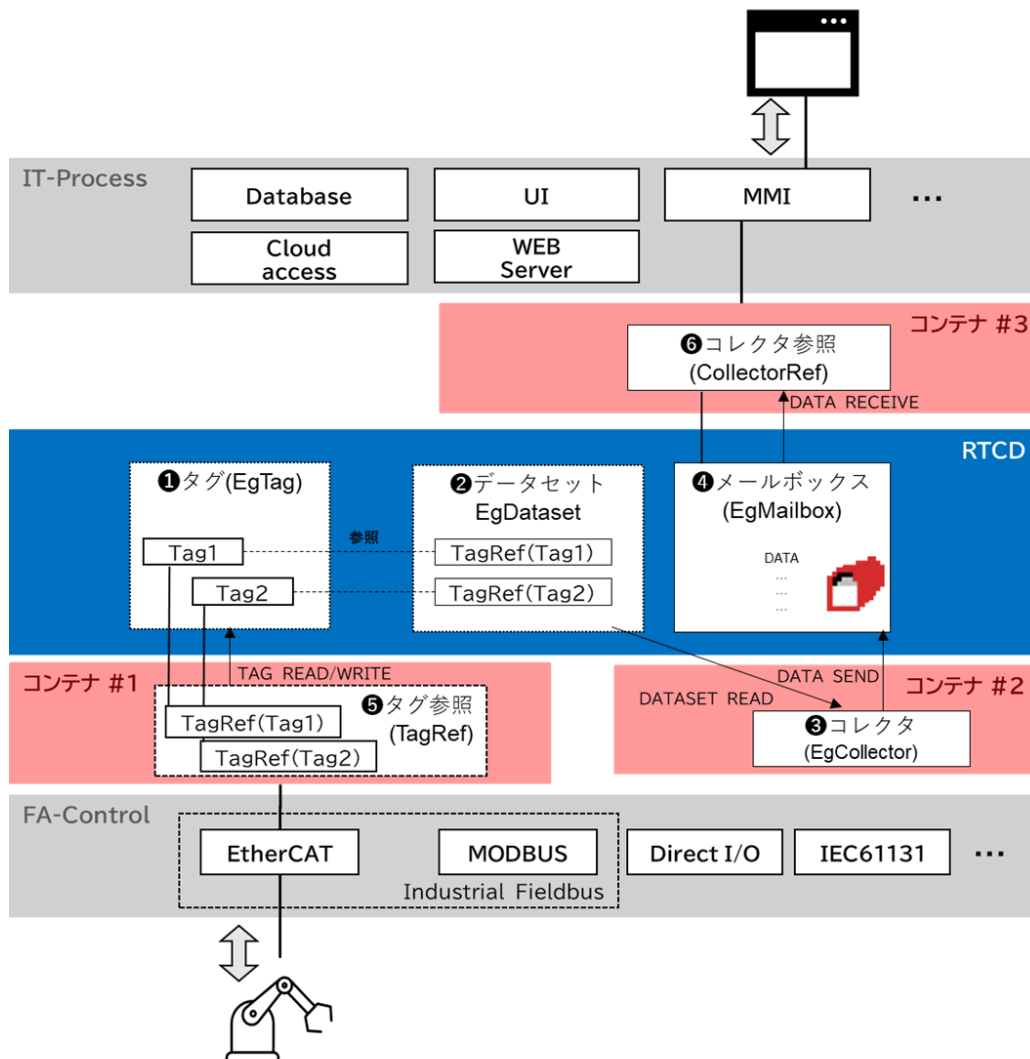


図 7 主要な RT-edge オブジェクトとその仕組み

3.5.1. ①タグ(EgTag)オブジェクト

瞬時値データ値 1 つを示すオブジェクトで、ユニーク名と、グローバルなスコープを持ち、すべてのサービスコンテナから参照が可能なオブジェクトです。主な仕様は以下のとおりです。

- 1) アドレス書式によるデータ位置とのリンケージ情報を保持
- 2) 書き込まれた最終値を保持
- 3) あらゆるデータ型のうち 1 つを選択して値を保持
- 4) リンクタグによりサービスコンテナ毎に異なるプロパティ情報の利用が可能

EgTag オブジェクトは以下のデータを定義・保持します。定義は ECI ファイルに記載するか、RT-edge API を使用します。(特に理由がない場合は、ECI ファイルへ定義してください)

ECI ファイルの記載方法に関しては「3.7. ECI ファイル」を参照してください。

表 6 EgTag 保持データ

データ項目	説明
EgTag 名	EgTag 名 48 バイト(半角英数)
型	「表 7 EgTag 型一覧」に示すデータ型
コメント	コメント 48 バイト(半角英数+全角)
アドレス	データ取得元、宛先となるアドレス情報 48 バイト(半角英数) 詳細は「3.5.1.1 アドレス(Address)」参照
値	Tag の値

表 7 EgTag 型一覧

No	型	サイズ(byte)	用途
0	UNDEFINE	-	未定義
1	Boolean	1	bool 値
2	SByte	1	符号付き 8 ビット整数
3	Byte	1	符号なし 8 ビット整数
4	Int16	2	符号付き 16bit 整数
5	UInt16	2	符号なし 16bit 整数
6	Int32	4	符号付き 32bit 整数
7	UInt32	4	符号なし 32bit 整数
8	Int64	8	符号付き 64bit 整数
9	UInt64	8	符号なし 64bit 整数
10	Float	4	単精度実数(32bit)
11	Double	8	倍精度実数(64bit)
12	String	-	文字列 サイズは ECI で指定
13	Segment	-	メモリブロック サイズは ECI で指定

3.5.1.1. アドレス(Address)

アドレス(Address)にはデータ取得元、宛先となるアドレス情報を定義します。Address 表記は IEC 表記に似せて作られています。バイナリデータ列を使った値の読み書きを簡略化できるユーティリティ API 関数で使します。

フォーマット形式は以下で構成されます。

%[A][B].[C]

A) : サイズ接頭語 ("X", "B", "W", "D", "L")

B) : バイトのメモリアドレス

C) : ビット位置 (サイズ接頭語が X:シングルビットサイズのときのみ定義)

表 8 アドレス表記

サイズ 接頭語	型	説明	例	例説明
X	Boolean	シングルビットサイズ	%X2.4	オフセット 2 の 4 ビット目の値
B	SByte	バイトサイズ(1 バイト)	%B12	オフセット 12 から 1 バイト分の値
	Byte			
W	Int16	ワードサイズ(2 バイト)	%W52	オフセット 52 から 2 バイト分の値
	UInt16			
D	Int32	ダブルワードサイズ(4 バイト)	%D3	オフセット 3 から 4 バイト分の値
	UInt32			
	Float			
L	Int64	ロングワードサイズ(8 バイト)	%L7	オフセット 7 から 8 バイト分の値
	UInt64			
	Double			



- 上記フォーマットは、以下の RT-edge API 関数で使します。
 - EgEncode() : バイナリデータ列への書き込みを行う API
 - EgDecode() : バイナリデータ列からのデータ抽出を行う API
 データはリトルエンディアンとして参照・書き込みされます。
- 標準コンテナで使用する EgTag の Address はサービスコンテナ毎に異なります。詳細はご利用の標準コンテナユーザズマニュアルをご確認ください。

3.5.1.2. リンクタグ

リンクタグは、複数のプロパティ情報(ソース/コメント)を設定するために用いるオブジェクトです。タグにはソース情報/コメント情報を1つずつ設定することができますが、サービスコンテナ毎に異なるアドレス書式を使用したい場合があります。例えば EtherCAT や Modbus といった Fieldbus を用いたサービスコンテナと、それら Fieldbus のデータをソフトウェア PLC(INplc)の入出力データとして利用する場合です。

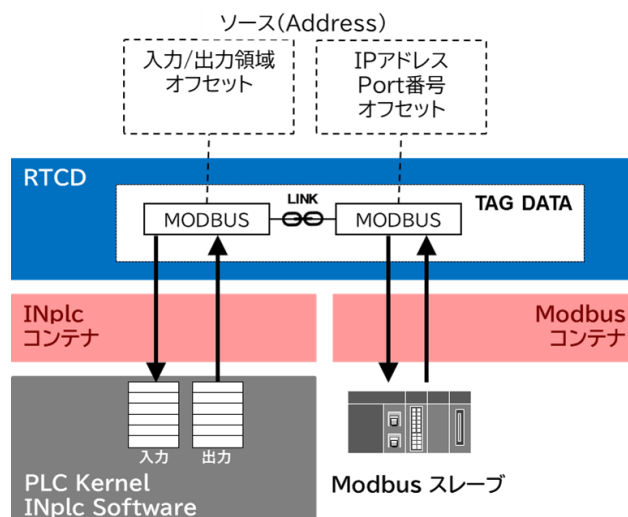


図 8 リンクタグ使用例

Fieldbus(Modbus)を用いたサービスコンテナでは、格納するデータのソース情報として、接続機器であるスレーブ IP アドレス情報やオフセット値をソース(Address)に定義します。INplc サービスコンテナでは、これらのタグの値をどの PLC カーネルの入出力領域にマッピングするかの設定をソース情報に定義します。このようなケースの場合、リンクタグを利用することで一つのタグに対し複数のプロパティ情報を定義することが可能になります。

リンクタグは、既に生成済みのタグ(以下、オリジナルタグ)と同一名称のタグが ECI ファイルに定義されていた場合、RT-edge フレームワークは自動的に別名称を付与したタグ(以下、リンクタグ)を生成します。このリンクタグは、下記の仕様に従います。

- 1) タグ名先頭に"#0"~"#9"の2文字が付与されます。

#[A][B]

A : 0~9 の数字が順番に割り当てられます

B : オリジナルタグの名称

- 2) 最大 10 個まで生成可能です。
- 3) オリジナルタグと型の異なるリンクタグは生成出来ません。
- 4) 値のみオリジナルタグのデータを参照します。
- 5) それ以外のプロパティ情報(コメントやソースなど)はリンクタグのデータを参照します。

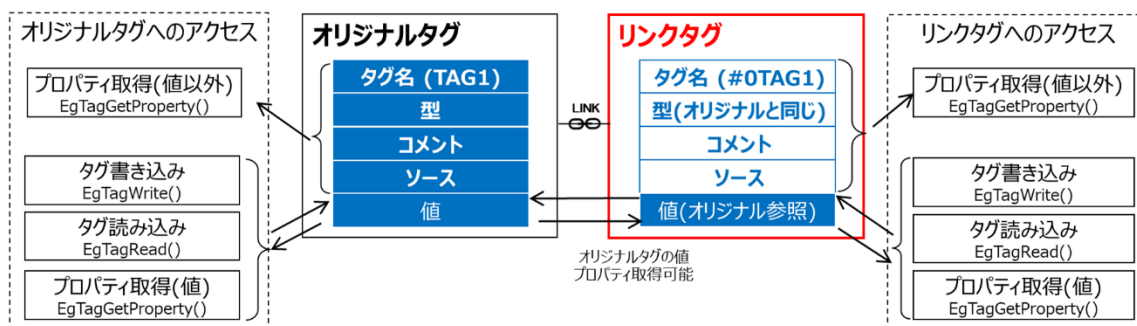


図 9 リンクタグの参照イメージ

3.5.2. ②データセット(EgDataset)オブジェクト

タグ(EgTag)を 1 つ以上組み合わせでデータ並び順（データ構造）を定義する名前付きオブジェクトです。コレクタ(EgCollector)のデータ取得・参照に使用します。

EgDataset オブジェクトは以下のデータを定義・保持します。定義は ECI ファイルに記載するか、RT-edge API を使用することも可能です。（特に理由がない場合は、ECI ファイルへ定義して下さい）

ECI ファイルの記載方法に関しては「3.7. ECI ファイル」を参照ください。

」を参照してください。

表 9 EgDataset 保持データ

データ項目	説明
EgDataset 名	EgDataset 名 48 バイト(半角英数)
タグ数	EgTag へのリンク数
タグへのリンク	EgTag へのリンク

3.5.3. ③コレクタ(EgCollector)オブジェクト

データセットに定義されたデータ構造を使って、同時刻のバイナリデータ列で生成し、データレコードとしてメールボックスに送信する機能です。主な仕様は以下のとおりです。

- 1) 各サービスコンテナ（RT-edge フレームワーク内）で、ECI ファイルに定義された EgCollector 名のメールボックスを作成
- 2) ECI ファイル内で定義された EgTag 情報を定周期(指定値)で取得し、取得した順(ECI ファイルに記載順)に 1 レコードとして、上記メールボックスに格納

EgCollector オブジェクトは以下のデータを定義・保持します。定義は ECI ファイルに記載するか、RT-edge API を使用して登録することも可能です。（特に理由がない場合は、ECI ファイルへ定義してください）

表 10 EgCollector 保持データ

データ項目	説明
EgCollector 名	EgCollector 名 8 バイト(半角英数)
収集周期	収集周期

データ項目	説明
プライオリティ	INtime サービスコンテナ用スレッドプライオリティ INtime 同様 0~254 を指定(デフォルト 150)
メールボックス 1 レコードデータサイズ	格納する 1 レコードのデータサイズ(デフォルト 2048)
メールボックスレコード数	格納数
データセット名	データセット名

3.5.4. ④メールボックス(EgMailbox)オブジェクト

時系列なデータセット、または時系列メッセージを FIFO で蓄えることができ、また受信イベントとして処理できるオブジェクトです。

メッセージを送信する側では、送信したいメッセージを FIFO に入れ、メッセージを受信する側では、FIFO に入っているメッセージを一つ取り出します。FIFO にメッセージが入っていない場合は、次のメッセージが送られてくるまで受信待ち状態になります。受信待ち時間は設定できます。

3.5.5. ⑤TagRef/⑥CollectorRef オブジェクト

タグ(EgTag)の参照、コレクタ(EgCollector)の参照として実態の名前を保持し、名前を用いて実態(EgTag, EgCollector で集めたデータを保持するメールボックス)のデータ取得等に使用するオブジェクトです。

3.6. RT-edge フレームワーク

サービスコンテナは RT-edge API を用いてスクラッチで開発できますが、RT-edge API の組み合わせで実現する機能、使用頻度の高い処理構造を「RT-edge フレームワーク」として提供しています。RT-edge フレームワークの使用は必須ではありません。しかし使用することで次のような利点があります：

- 1) 自コンテナ専用 ECI ファイル(XML)を用意して、カスタマイズ可能なコンテナを提供できます。
- 2) RT-edge API を使ったプログラミングを行わずとも、ECI ファイルで定義した複数のタグやメールボックスが自動生成されます。
- 3) ECI ファイル内でのタグ名称変更が可能となりプログラム改造の頻度を少なくします。

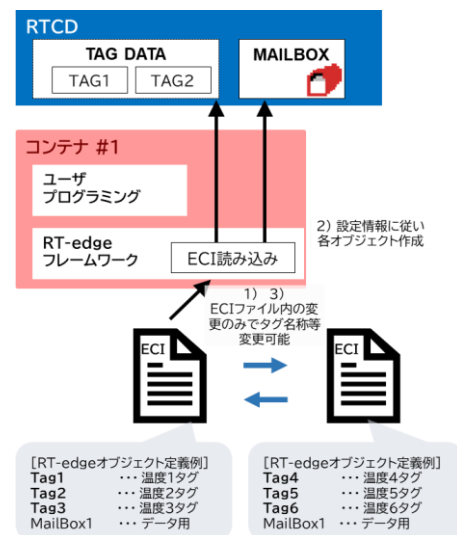


図 10 ECI 読み込みイメージ

- 4) サービスコンテナプログラムは RT-edge フレームワーク変数(グローバル変数を介して ECI ファイルで定義された RT-edge オブジェクトリスト (TagRefs 等)を取り出すことができます。

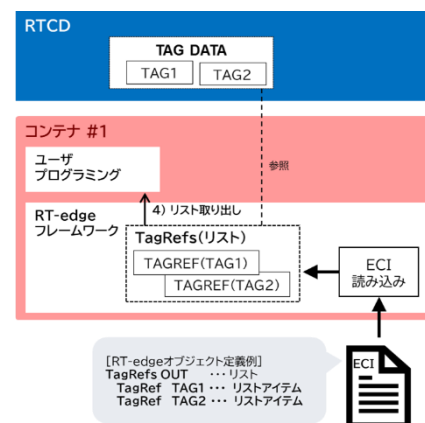


図 11 RT-edge オブジェクトリストとの取り出し

- 5) ECI ファイルで定義されたコンテナプロパティ値を取り出しコンテナプログラム動作を変えるなど応用ができます。

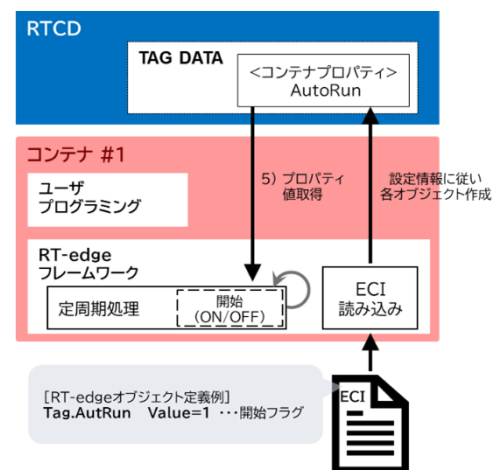


図 12 コンテナプロパティ値の取り出し

- 6) 標準コンテナステータスインジケータが自動生成され、コンテナ状態を公開可能することができます。

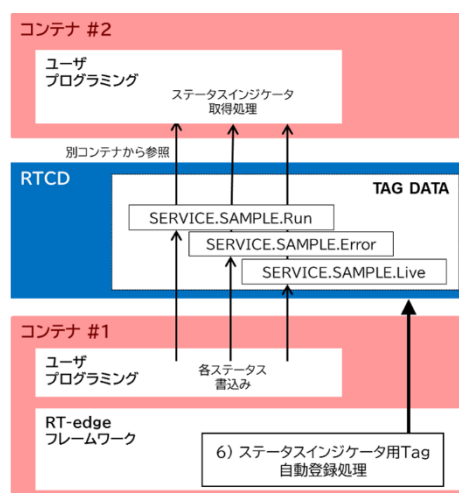


図 13 ステータスインジケータの自動生成

- 7) RT-edge システムのコンテナ間メッセージ通信機能が備わり、標準定義メッセージは自動処理が行われます。
- 8) ユーザー独自メッセージ通信ハンドラが登録可能となり、受信メッセージに応じた反応処理に書き換えることができます。

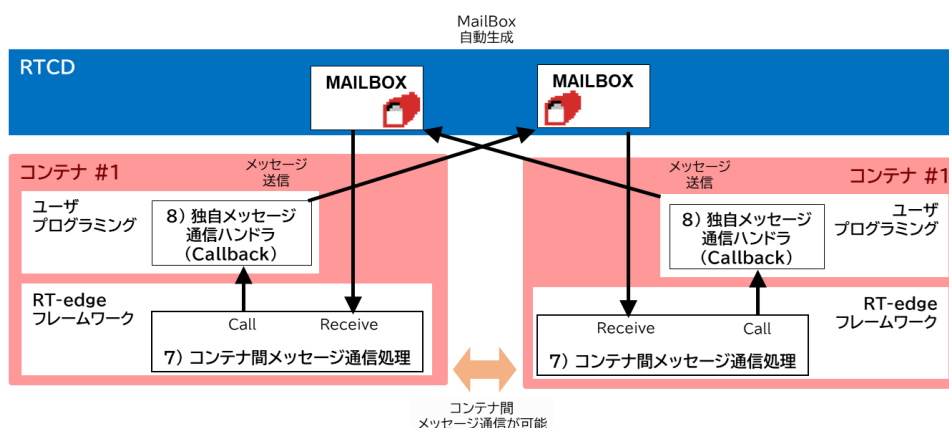


図 14 コンテナ間メッセージ通信機能

3.6.1. RT-edge フレームワーク構造図

RT-edge フレームワークは RT-edge API ライブラリに組み込まれており、EgInit()API をコールすれば容易に RT-edge フレームワークを備えたサービスコンテナになります。

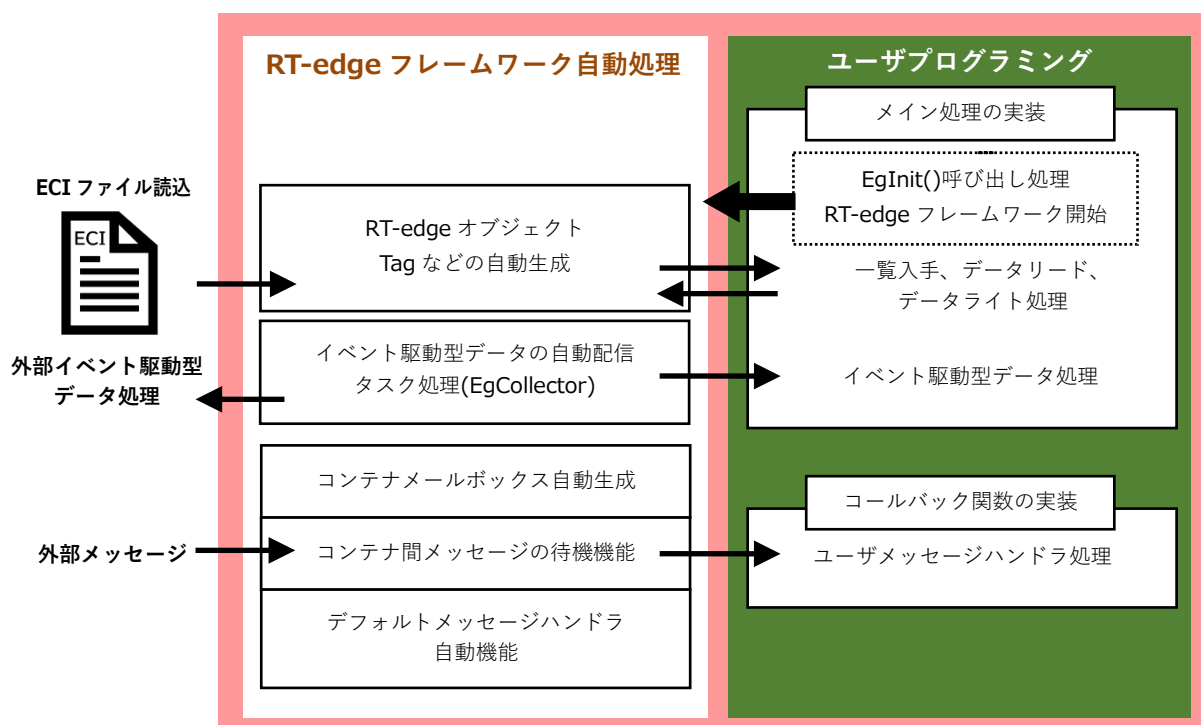


図 15 RT-edge フレームワークを利用したサービスコンテナ

3.6.2. EgService 間メッセージ通信

RT-edge フレームワークを使用することで、EgService 間のメッセージ送受信を容易に実装することができます。メッセージ番号は以下の仕様で決定します。

- 1) 00001～19999：システムメッセージ(予約)
- 2) 20000～29999：ユーザーメッセージ

メッセージの送信方法、ならびにシステムメッセージについては、「RT-edge API リファレンスマニュアル(DOCTREDGEAPI.pdf)」の付録内「メッセージ通信用システムメッセージ一覧」を参照してください。

3.6.3. RT-edge フレームワークで登録する EgTag 情報(サービスコンテナインジケータ)

RT-edge フレームワークでは以下の EgTag を自動で登録します。

表 11 RT-edge フレームワークで登録する EgTag 情報

EgTag 名	型	用途
SERVICE.サービスコンテナ名	-	Source にパスを格納
SERVICE.サービスコンテナ名.State	Byte	0:終了済 1:正常起動中 2:initialize 中エラー発生 3:finalize 中エラー発生 4:System エラー発生(上記以外)
SERVICE.サービスコンテナ名.Run	Boolean	0:停止、1:正常
SERVICE.サービスコンテナ名.Error	Boolean	0:正常、1:データ更新異常
SERVICE.サービスコンテナ名.Live	UInt32	動作中はインクリメント処理実行



- 1) RT-edge フレームワークで登録するサービスコンテナ固有の EgTag 名称の先頭には「SERVICE.」が付きます。また、上記 EgTag はあくまでも自動で登録するのみです。値の更新は作成したサービスコンテナごとに行う必要があります。
- 2) 通常のタグはシステム内でユニークな情報ですが、リンクタグを利用することで一つのタグに対し複数のプロパティ情報を定義することが可能となります。これは、サービスコンテナ毎に異なるアドレス書式を使用したい場合などに有効です。
- 3) 表で記載のサービスインジケータタグは瞬時値です。「SERVICE.サービスコンテナ名.Error」などは保持しません。

3.6.4. タグトリガー機能

RT-edge フレームワークにはタグトリガーという機能が用意されています。これは、ECI ファイルにタグトリガー(イベント受信対象タグ)を登録しておくことで、EgTag の値が書き換わったときにイベントを受信できます。下図に動作のイメージを示します。

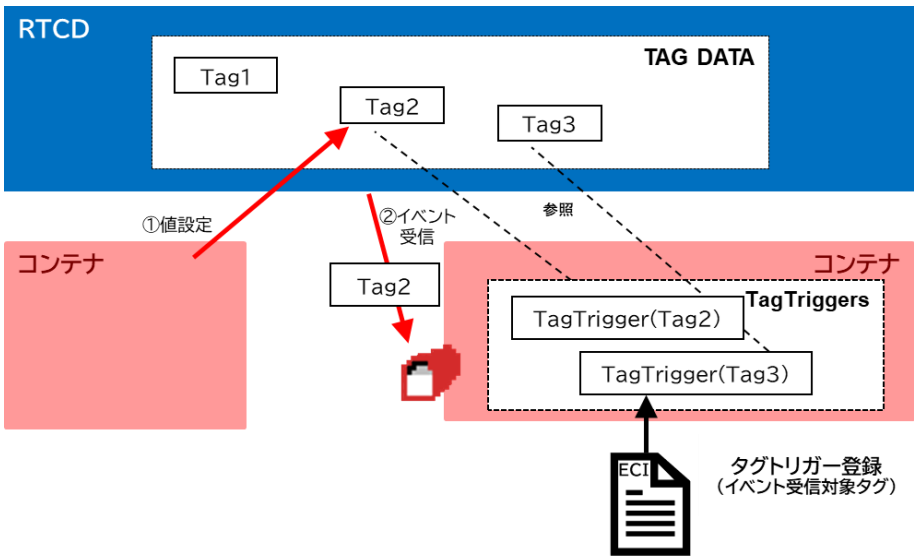


図 16 TagTrigger イメージ図

タグトリガー機能の仕様は以下の通りです。

項目	最大値	備考
サービスコンテナで登録可能なタグトリガー数	128	



複数のタグトリガーを登録した場合や、コンテナへの通知処理のタイミングによって、性能へ影響がある場合がございます。

3.7. ECI ファイル

RT-edge システムでは、タグ、データセット、コレクタ、メールボックスの作成と参照、サービスコンテナの起動を ECI ファイルに定義することで、RT-edge フレームワークが自動処理(作成・参照・サービスコンテナの起動)を行います。これにより、プログラミングの省略が可能になっています。以下に ECI ファイルについて説明します。

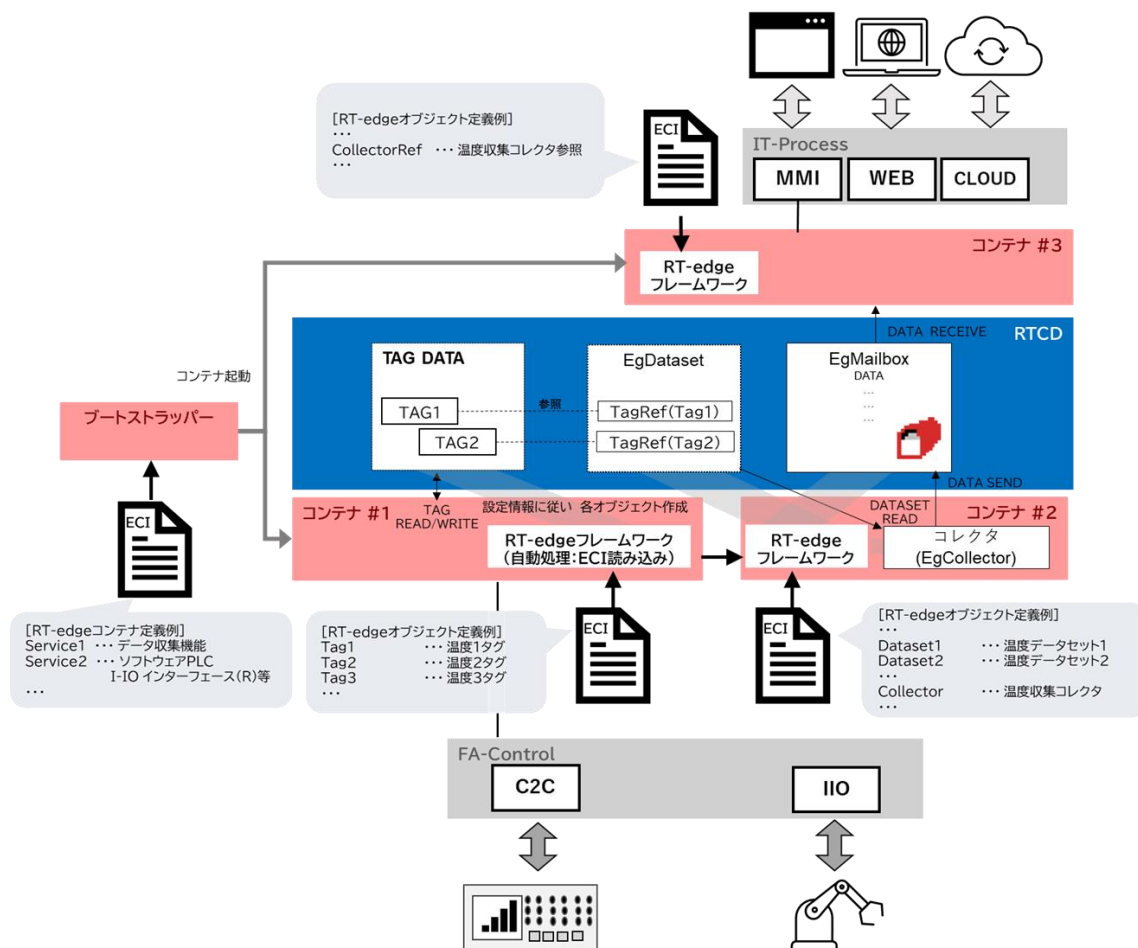


図 17 ECI ファイルとオブジェクトの関連図

3.7.1. ファイル定義

- 1) RT-edge API の EgInit() をコールすることで、RT-edge フレームワークを初期化し、ECI ファイルが解釈処理されます。
- 2) ECI ファイルは、サービスコンテナ名.xml として、サービスコンテナの実行ファイルと同一フォルダに配置する必要があります。
- 3) サービスコンテナごとに定義可能です。

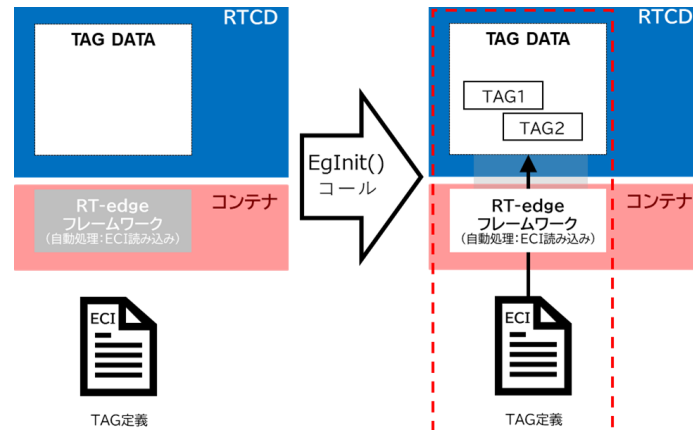


図 18 RT-edge フレームワーク初期化イメージ

3.7.2. 基本構造

ECI ファイルは XML フォーマットで定義されています。ECI ファイルの構造は以下のとおりです。

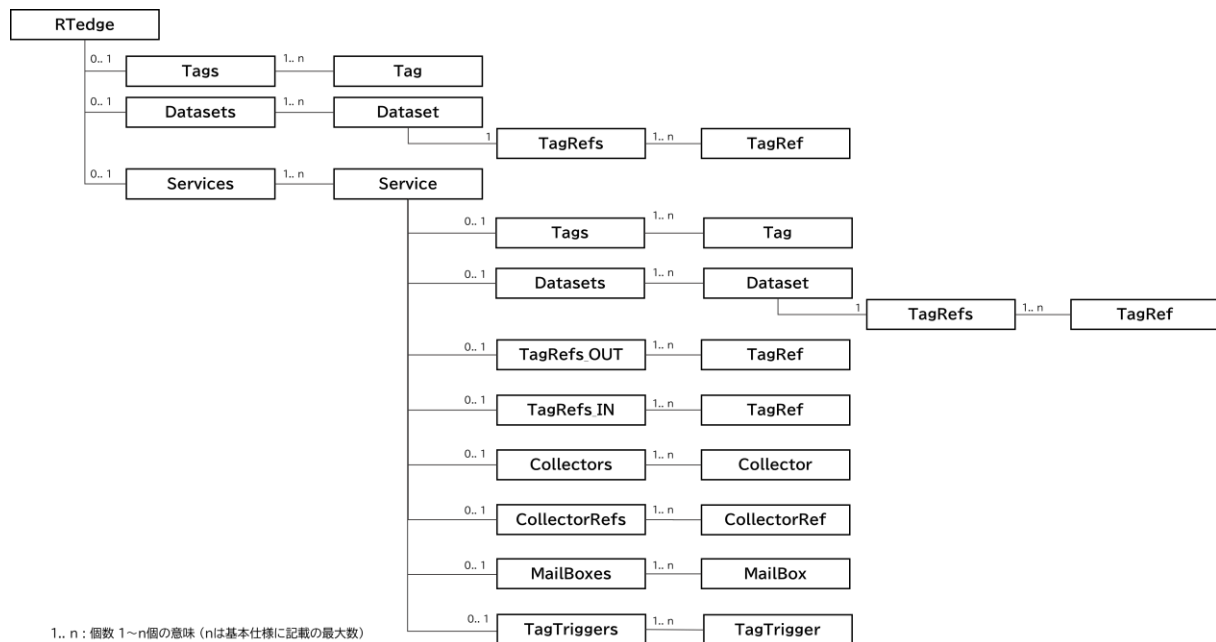


図 19 ECI ファイル構造 ツリー表示

表 12 ECI ファイル基本構造

構文	説明
<?xml version="1.0 .../>	Xml 固定、ユニコード UTF-8 で保存します、必須
<RTedge xmlns:xsi ...>	RTedge エLEMENTの始まり、必須
<Tags>	Tags ELEMENTの始まり、任意[0…1]
<Tag Name="" Type="" Address="" Comment="" />	Tag ELEMENT、RT-edge システム内で重複禁止[1…n]
<Tag Name="" Type="" Address="" Comment="" />	任意の個数[1…n]
<Tag Name="" Type="" Address="" Comment="" />	任意の個数[1…n]
</Tags>	Tags ELEMENTの終了
<Datasets>	Datasets ELEMENTの始まり、任意[0…1]
<Dataset Name="">	Dataset ELEMENTの始まり、任意、複数可能[1…n]
<TagRefs>	TagRefs ELEMENTの始まり[1…1]
<TagRef Name="" />	TagRef ELEMENT[1…n]
<TagRef Name="" />	任意の個数[1…n]
<TagRef Name="" />	任意の個数[1…n]
</TagRefs>	TagRefs ELEMENTの終了
</Dataset>	Dataset ELEMENTの終了
<Dataset Name="">	Dataset ELEMENTの始まり、任意、複数可能[1…n]
<TagRefs>	
<TagRef Name="" />	
<TagRef Name="" />	
<TagRef Name="" />	
</TagRefs>	
</Dataset>	
</Datasets>	Datasets の終了
<Services>	Services ELEMENTの始まり、任意[0…1]
<Service Name="" />	Service ELEMENT、複数可能[1…n]
<Service Name="" />	任意の個数[1…n]
<Service Name="" />	任意の個数[1…n]、サービスコンテナ内部に生成指定
<Tags>	Tags ELEMENTの始まり、任意[0…1]

<Tag Name="" Type="" Address="" Comment="" />	Tag エlement、RT-edge システム内で重複禁止[1…n]
</Tags>	Tags エlementの終了
<Datasets>	Datasets エlementの始まり、任意[0…1]
<Dataset Name="">	Dataset エlementの始まり、任意、複数可能[1…n]
<TagRefs>	TagRefs エlementの始まり[1…1]
<TagRef Name="" />	TagRef エlement[1…n]
</TagRefs>	TagRefs エlementの終了
</Dataset>	Dataset エlementの終了
</Datasets>	Datasets の終了
<TagRefs_OUT>	TagRefs_OUT エlementの始まり、任意[0…1]
<TagRef Name="" />	TagRef エlement[1…n]
<TagRef Name="" />	任意の個数[1…n]
<TagRef Name="" />	任意の個数[1…n]
</TagRefs_OUT>	TagRefs_OUT エlementの終了
<TagRefs_IN>	TagRefs_IN エlementの始まり、任意[0…1]
<TagRef Name="" />	TagRef エlement[1…n]
<TagRef Name="" />	任意の個数[1…n]
<TagRef Name="" />	任意の個数[1…n]
</TagRefs_IN>	TagRefs_IN エlementの終了
<TagRefs>	TagRefs エlementの始まり、任意[0…1]
<TagRef Name="" />	TagRef エlement[1…n]
<TagRef Name="" />	任意の個数[1…n]
<TagRef Name="" />	任意の個数[1…n]
</TagRefs>	TagRefs エlementの終了
<Collectors>	Collectors エlementの始まり、任意[0…1]
<Collector Name="" Interval="" DatasetName="" />	Collector エlement[1…n]
<Collector Name="" Interval="" DatasetName="" />	任意の個数[1…n]
<Collector Name="" Interval="" DatasetName="" />	任意の個数[1…n]
</Collectors>	Collector エlementの終了
<CollectorRefs>	CollectorsRef エlementの始まり、任意[0…1]
<CollectorRef Name="">	CollectorRef エlement[1…n]
<CollectorRef Name="">	任意の個数[1…n]
<CollectorRef Name="">	任意の個数[1…n]
</CollectorRefs>	CollectorsRef エlementの終了
<MailBoxes>	Mailboxes エlementの始まり、任意[0…1]
<MailBox Name="" ColSize="" RowSize="" />	MailBox エlement[1…n]
<MailBox Name="" ColSize="" RowSize="" />	任意の個数[1…n]
<MailBox Name="" ColSize="" RowSize="" />	任意の個数[1…n]
</MailBoxes>	Mailboxes エlementの終了
<TagTriggers>	TagTrigger エlementの始まり、任意[0…1]
<TagTrigger Name="" />	TagTrigger エlement[1…n]
<TagTrigger Name="" />	任意の個数[1…n]
<TagTrigger Name="" />	任意の個数[1…n]
</TagTriggers>	TagTrigger エlementの終了
</Service>	Service エlementの終了
</Services>	Services エlementの終了
</RT-edge>	RT-edge 定義の終了

サンプル用 ECI ファイルを次頁に記載します。

```

<?xml version="1.0" encoding="utf-8"?>
<RTedge xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Tags>
    <Tag Name="Temp001" Type="6" Address="%D0" Comment="温度 1 オフセット 0,4 バイト"/>
    <Tag Name="Temp002" Type="11" Address="%L4" Comment="温度 2 オフセット 4,8 バイト"/>
    <Tag Name="Temp003" Type="10" Address="%D12" Comment="温度 3 オフセット 12,4 バイト"/>
  </Tags>
  <Datasets>
    <Dataset Name="Dset1">
      <TagRefs>
        <TagRef Name="Temp001"/>
        <TagRef Name="Temp002"/>
        <TagRef Name="Temp003"/>
      </TagRefs>
    </Dataset>
  </Datasets>
  <Services>
    <Service Name="EgBoot" TagMaxNum="10000" Argument="RTCD=NodeA"/>
    <Service Name="EgLog" Path="EgLog.exe" Argument="DispNumMax=500"/>
    <Service Name="EgTime" Path="EgTime.exe"/>
    <Service Name="IoSmpl" Path="IoSmpl.rta"/>
    <Service Name="ClctSmpl" Path="ClctSmpl.rta">
      <TagRefs_OUT>
        <TagRef Name="Temp001"/>
        <TagRef Name="Temp002"/>
        <TagRef Name="Temp003"/>
      </TagRefs_OUT>
      <Collectors>
        <Collector Name="ClctIO" Interval="100" Priority="140" DatasetName="Dset1"/>
      </Collectors>
    </Service>
    <Service Name="DispSmpl" Path="DispSmpl.exe">
      <TagRefs_IN>
        <TagRef Name="Temp001"/>
        <TagRef Name="Temp002"/>
        <TagRef Name="Temp003"/>
      </TagRefs_IN>
      <CollectorRefs>
        <CollectorRef Name="ClctIO" />
      </CollectorRefs>
    </Service>
  </Services>
</RTedge>

```

RT-edge
オブジェクト
定義例

RT-edge
オブジェクト
定義例

RT-edge
コンテナ
定義例

RT-edge
オブジェクト
定義例

3.7.3. 構成要素

ECI ファイルの構成要素について説明します。

3.7.3.1. RTedge エlement

項目名	エレメント名	説明
親	-	-
子	Tags	[0..1] (個数 0～1 個の意味 以降説明省略)
	Datasets	[0..1]
	Services	[0..1]
属性名	サイズ(byte)	説明
なし	-	-

3.7.3.2. Tags エlement

項目名	エレメント名	説明
親	RTedge	
	Service	
子	Tag	[1..n] (個数 1～n 個の意味 以降説明省略)
属性名	サイズ(byte)	説明
なし	-	-

3.7.3.3. Datasets エlement

項目名	エレメント名	説明
親	RTedge	
	Service	
子	Dataset	[1..n]
属性名	サイズ(byte)	説明
なし	-	-

3.7.3.4. Services エlement

項目名	エレメント名	説明
親	RTedge	
子	Service	[1..n]
属性名	サイズ(byte)	説明
なし	-	-

3.7.3.5. Collectors エlement

項目名	エレメント名	説明
親	Service	
子	Collector	[1..n]
属性名	サイズ(byte)	説明
なし	-	-

3.7.3.6. MailBoxes エlement

項目名	エレメント名	説明
親	Service	
子	MailBox [1..n]	
属性名	サイズ(byte)	説明
なし	-	-

3.7.3.7. TagTriggers エlement

項目名	エレメント名	説明
親	Service	
子	TagTrigger [1..n]	
属性名	サイズ(byte)	説明
なし	-	-

3.7.3.8. Tag エlement

RT-edge フレームワークに EgTag の生成を要求できます。

項目名	エレメント名	説明
親	Tags	
子	なし -	
属性名	サイズ(byte)	説明
Name	48	EgTag 名(半角英数字)
Type	2	データタイプ(型) (表 7 EgTag 型一覧)
Comment	48	任意のコメント文字列(2 バイトデータに関して、24 文字まで)
Address	48	データのソースを表現する文字列を格納
Value	8	初期値を指定可。指定のないとき値ゼロ
Size	-	Segment 型、String 型指定時に使用可能なサイズを指定、左記以外の型は指定なし

3.7.3.9. Dataset エlement

RT-edge フレームワークに EgDataset の生成を要求できます。

項目名	エレメント名	説明
親	Datasets	
子	TagRefs [1...1]	
属性名	サイズ(byte)	説明
Name	48	EgDataset 名(半角英数字)、RT-edge システム内で重複禁止

3.7.3.10. Collector エlement

RT-edge フレームワークに EgCollector の生成を要求できます。

項目名	エレメント名	説明
親	Collectors -	
子	なし -	
属性名	サイズ(byte)	説明
Name	8	EgCollector 名(半角英数字)、RT-edge システム内で重複禁止

項目名	エレメント名	説明
Interval	4	EgTag の値を読み取り、RTCD に格納する周期(ms) (1 ~3600000) 0 を指定した場合は、デフォルト 100ms として動作します。
Priority	2	EgCollector を実行するスレッドの Priority (0~254 デフォルト 150)
DatasetName	48	EgDataset 名
ColSize	4	1 レコードデータサイズ (最大長はシステムに依存 詳細は「3.7.3.20 レコードデータサイズ・レコード数に関して」を参照)
RowSize	4	レコード数 (最大数はシステムに依存 詳細は「3.7.3.20 レコードデータサイズ・レコード数に関して」を参照)
Comment	48	EgCollector 名(半角英数字)、RT-edge システム内で重複禁止

3.7.3.11. MailBox エレメント

RT-edge フレームワークに EgMailBox の生成を要求できます。

項目名	エレメント名	説明
親	MailBoxes	-
子	なし	-
属性名	サイズ(byte)	説明
Name	8	EgMailBox 名(半角英数字)、RT-edge システム内で重複禁止
ColSize	4	1 レコードデータサイズ (最大長はシステムに依存 詳細は「3.7.3.20 レコードデータサイズ・レコード数に関して」を参照)
RowSize	4	レコード数 (最大数はシステムに依存 詳細は「3.7.3.20 レコードデータサイズ・レコード数に関して」を参照)

3.7.3.12. TagTrigger エレメント

RT-edge フレームワークに EgTag へのタグトリガー登録を要求できます。

項目名	エレメント名	説明
親	TagTriggers	-
子	なし	-
属性名	サイズ(byte)	説明
Name	48	タグトリガーを登録したい EgTag 名(半角英数字)

3.7.3.13. Service エlement

RT-edge フレームワークに EgService の生成(プロセス起動)を要求できます。

起動済みのサービスコンテナの設定は定義しないでください。正しく設定が行われません。

項目名	エレメント名	説明
親	Services	-
子	Tags	[0..1]
	Datasets	[0..1]
	TagRefs_OUT	[0..1]
	TagRefs_IN	[0..1]
	Collectors	[0..1]
	CollectorsRef	[0..1]
	Mailboxes	[0..1]
	TagTriggers	[0..1]
属性名	サイズ(byte)	説明
Name	24	EgService 名(半角英数字)、RT-edge システム内で重複禁止
Path	256	EgServiceCreate で起動する実行モジュールファイルパス
Priority	1	サービスコンテナメッセージポンプスレッドのプライオリティ (0~254 デフォルト 150)
Argument	-	サービスコンテナ起動引数
Comment	48	任意のコメント文字列
Timeout	4	起動待機時間(sec)を指定 (0~600 デフォルト 30) RT-edge フレームワークを備えていないサービスコンテナには値ゼロを指定
Dependent	-	依存するサービスコンテナ名を指定 (複数指定の場合は、セミコロンで区切って入力)
Node	64	サービスコンテナを起動させる INtime ノード名を指定(デフォルトローカルノード)。RTA(INtime アプリケーション)の場合のみ使用
ColSize	4	サービスコンテナ間通信用メールボックスの 1 レコードデータサイズ (最大長はシステムに依存 詳細は「3.7.3.20 レコードデータサイズ・レコード数に関して」を参照)
RowSize	4	サービスコンテナ間通信用メールボックスのレコード数 (最大数はシステムに依存 詳細は「3.7.3.20 レコードデータサイズ・レコード数に関して」を参照)
OnlyOwner	1	起動元と同じ Name(Service Name)の時のみ読み込む設定 TRUE:起動元と同じ Name の時のみ読み込む FALSE:常に読み込む (同一の ECI を複数読み込む場合に設定します)
ECI	256	起動時に読み込む ECI ファイルパス



表で定義している値範囲に関して、範囲外の値を指定した場合は、エラー終了、または動作が不安定になる可能性があります。定義された範囲外の値は設定しないでください。

3.7.3.14. TagRefs_OUT エレメント

RT-edge フレームワークに出力用タグのリスト定義を要求できます。主にタグの値を読み込む処理に使用する TagRef のリストです。以下図は、ECI ファイルに定義した TagRef の名前属性を用いて EgTag の値を読み込む流れを記載しています。

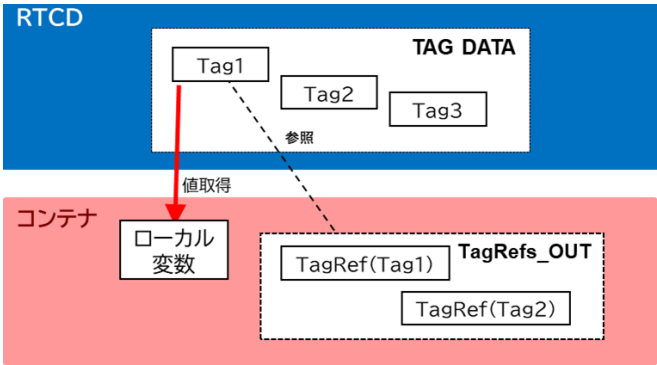


図 20 TagRefs_OUT イメージ図

項目名	エレメント名	説明
親	Service	-
子	TagRef	[1..n]
属性名	サイズ(byte)	説明
なし	-	-

3.7.3.15. TagRefs_IN エレメント

RT-edge フレームワークに入力用タグのリスト定義を要求できます。主にタグへ値を書き込む処理に使用する TagRef のリストです。以下図は、ECI ファイルに定義した TagRef の名前属性を用いて EgTag へ値を書き込む流れを記載しています。

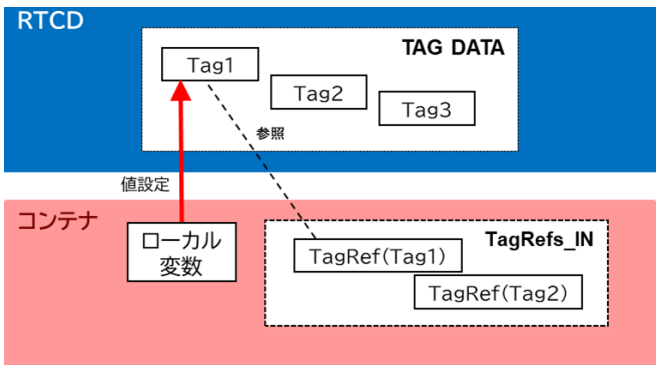


図 21 TagRefs_IN イメージ図

項目名	エレメント名	説明
親	Service	-
子	TagRef	[1..n]
属性名	サイズ(byte)	説明
なし	-	-

3.7.3.16. TagRefs エレメント

RT-edge フレームワークにデータセット(EgDataset)のタグリスト定義を要求できます。データセットのイメージは「[図 7 主な RT-edge オブジェクトとその仕組み](#)」を参照してください。

項目名	エレメント名	説明
親	Dataset	-
子	TagRef	[1..n]
属性名	サイズ(byte)	説明
なし	-	-

3.7.3.17. TagRef エレメント

項目名	エレメント名	説明
親	TagRefs	データセット(EgDataset)内で定義
	TagRefs_IN	-
	TagRefs_OUT	-
子	なし	-
属性名	サイズ(byte)	説明
Name	48	参照 EgTag 名(半角英数字)

3.7.3.18. CollectorRefs エlement

RT-edge フレームワークにコレクタ情報のリスト定義を要求できます。以下図では、ECI ファイルに定義した **CollectorRef** の名前属性を用いて **EgCollector** で取得したデータが格納されているメールボックスへアクセスし、データを取得する流れを記載しています。

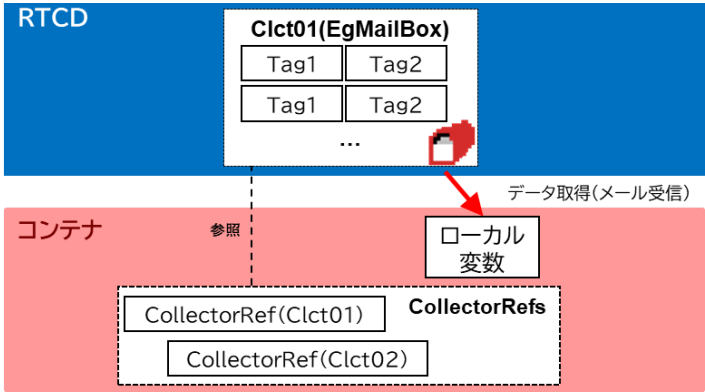


図 22 CollectorsRefs イメージ図

項目名	エレメント名	説明
親	Service	-
子	CollectorRef	[1..n] (同名の CollectorRef は登録できません)
属性名	サイズ(byte)	説明
なし	-	-

3.7.3.19. CollectorRef Element

項目名	エレメント名	説明
親	CollectorRefs	-
子	なし	-
属性名	サイズ(byte)	説明
Name	8	参照 EgCollector 名(半角英数字)



- 1) エレメント名、属性名は大文字小文字を区別して読み込みます。正しい名前ではない場合、値を読み込みませんので注意してください。
- 2) システム/サービスコンテナ起動中に ECI ファイルを変更しても実行中サービスコンテナには反映されません。ECI ファイルを変更後は、RT-edge システムを再度立ち上げなおしてください。

3.7.3.20. レコードデータサイズ・レコード数に関して

MailBox のレコードデータサイズとレコード数のイメージ図を以下に記載します。

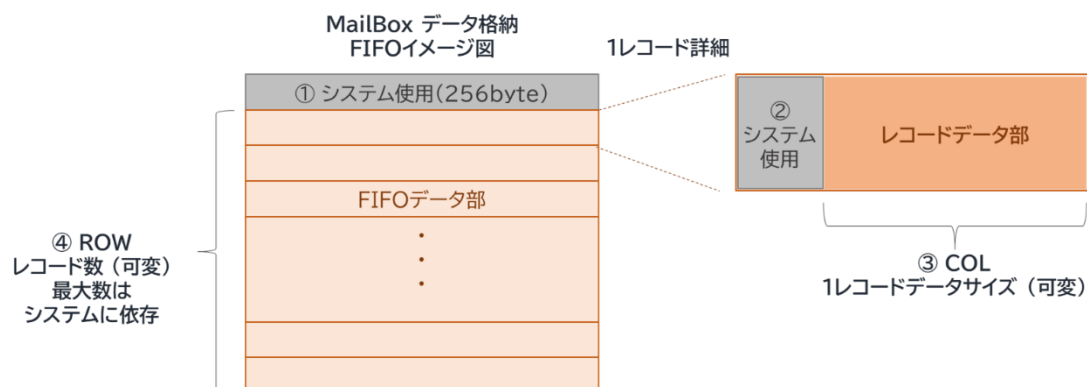


図 23 MailBox レコードデータサイズ/レコード数イメージ図

MailBox 1 個当たりのメールボックスサイズ(=メモリサイズ)を算出する式は以下の通りです。

$$\text{メールボックスサイズ[byte]} = \text{①} + ((\text{②} + \text{③}) \times \text{④})$$

- ① MailBox ヘッダ部 システム使用 : 256 (byte) (固定)
- ② レコードヘッダ部 システム使用 : 32 (byte) (固定)
- ③ 1 レコードデータサイズ : n (byte) (可変)
- ④ レコード数 : n (レコード) (可変)

以下はメールボックスサイズの計算例になります。下記の例の場合は最大約 3MB の空きメモリが必要です。

表 13 メールボックスサイズ計算例

例 No	①MailBox ヘッダ部 (byte)	②レコードヘッダ部 (byte)	③1 レコードデータサイズ(byte)	④レコード数	メールボックスサイズ合計(byte)
A	256	32	512	1000	544256
B	256	32	1024	2000	2112256
C	256	32	256	3000	864256

3.7.3.21. タグトリガー機能の性能設定に関して

タグトリガー機能では、タグの値変化の通知タイミングを設定によって変更することができます。設定はタグトリガーを登録する対象サービスコンテナの Service エレメント引数(Argument)に以下を指定します。

引数	値	説明
-trigcycle	1~100000	指定した値(ms)内でタグの値変化があった場合にイベントを送信します。

3.7.4. EgBoot 固有設定

EgBoot の Service エlementにおいてサービスコンテナ起動引数を利用して下記の設定が可能です。

※通常は未指定で問題ありません。必要に応じて定義してください。

表 14 EgBoot 起動引数

属性名	設定項目	説明
Argument	RTCD	RTCD を生成する INtime ノード名を指定します。未指定の場合は、ローカルノード(通常は NodeA)に生成されます。
	TagMaxNum	タグ最大登録数を指定します。未指定の場合は、1 万件まで生成可能です。 使用するタグ数が 1 万を超える場合は、設定を変更してください。

定義例) EgBoot.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RTedge xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Tags>
    <Tag Name="EDGESYSTEM.CurrentTime" Type="9" Comment="RT-edge Time"/>
    <Tag Name="EDGESYSTEM.CurrentTime_ms" Type="5" Comment="RT-edge Time ms"/>
    <Tag Name="EDGESYSTEM.CurrentTime_us" Type="5" Comment="RT-edge Time us"/>
    <Tag Name="EDGESYSTEM.OperatingTime" Type="9" Comment="RT-edge Operating Time"/>
  </Tags>
  <Datasets>
  </Datasets>
  <Services>
    <Service Name="EgBoot" Argument="RTCD=NodeB;TagMaxNum=20000"/>
    <Service Name="EgLog" Path="EgLog.exe" Argument="DispNumMax=500"/>
    <Service Name="EgTime" Path="EgTime.exe"/>
  </Services>
</RTedge>
```

4. ファイル情報

4.1. 実行環境

以下に実行環境用のファイルを記載します。

表 15 実行環境ファイル

フォルダ階層	ファイル名	説明
RT-edge¥	EgBrow.exe	RT-edge オブジェクトブラウザ
RT-edge¥	egapiWrap.dll	RT-edge 実行用ライブラリ
RT-edge¥	EgBoot.exe	RT-edge 起動用サービスコンテナモジュール
RT-edge¥	EgBoot.exe.config	RT-edge 起動用サービスコンテナモジュール
RT-edge¥	EgBoot.xml	EgBoot 用 ECI ファイル
RT-edge¥	EgDebug.exe	診断サービスコンテナ
RT-edge¥	eghgapiWin.dll	RT-edge 実行用ライブラリ
RT-edge¥	eghgapi.rsl	RT-edge 実行用ライブラリ
RT-edge¥	EgLog.exe	ログサービスコンテナモジュール
RT-edge¥	EgLog.exe.config	ログサービスコンテナモジュール
RT-edge¥	egosdepntx.dll	RT-edge 実行用ライブラリ
RT-edge¥	egosdepwin.dll	RT-edge 実行用ライブラリ
RT-edge¥	egosdep.rsl	RT-edge 実行用ライブラリ
RT-edge¥	egRTCDntx.dll	RT-edge 実行用ライブラリ
RT-edge¥	egRTCD.rsl	RT-edge 実行用ライブラリ
RT-edge¥	egRTCDwin.dll	RT-edge 実行用ライブラリ
RT-edge¥	EgShDown.exe	サービスコンテナのシャットダウンツール
RT-edge¥	EgShDown.xml	シャットダウンツール用 ECI ファイル
RT-edge¥	EgTime.exe	タイムサービスコンテナモジュール(Windows 環境用)
RT-edge¥	EgTime.exe.config	タイムサービスコンテナモジュール(Windows 環境用)
RT-edge¥	EgTime.rta	タイムサービスコンテナモジュール(INTIME 環境用)
RT-edge¥	EgTime.xml	タイムサービスコンテナ用 ECI ファイル
RT-edge¥	EgSysDef.xml	RT-edge システム用ファイル

4.2. 開発環境

サービスコンテナの開発を行うためのファイルは、ZIP ファイル展開時に、実行環境ファイルと同時に配置されます。必要に応じてパスの設定、ファイルの移動等を行ってください。

表 16 開発環境ファイル

フォルダ階層	ファイル名	説明
RT-edge¥Library¥	edgeAPI_RTCD.h	RT-edge 開発環境 API ヘッダ
RT-edge¥Library¥	egAPI.h	RT-edge 開発環境 API ヘッダ
RT-edge¥Library¥RT¥	eghgapi.lib	RT-edge 開発環境 API ライブラリ(INTime)
RT-edge¥Library¥RT¥	egosdep.lib	RT-edge 開発環境 API ライブラリ(INTime)
RT-edge¥Library¥RT¥	egRTCD.lib	RT-edge 開発環境 API ライブラリ(INTime)
RT-edge¥Library¥WIN¥	eghgapiWin.lib	RT-edge 開発環境 API ライブラリ(Windows)
RT-edge¥Library¥WIN¥	egosdepntx.lib	RT-edge 開発環境 API ライブラリ(Windows)
RT-edge¥Library¥WIN¥	egosdepwin.lib	RT-edge 開発環境 API ライブラリ(Windows)
RT-edge¥Library¥WIN¥	egRTCDntx.lib	RT-edge 開発環境 API ライブラリ(Windows)
RT-edge¥Library¥WIN¥	egRTCDwin.lib	RT-edge 開発環境 API ライブラリ(Windows)

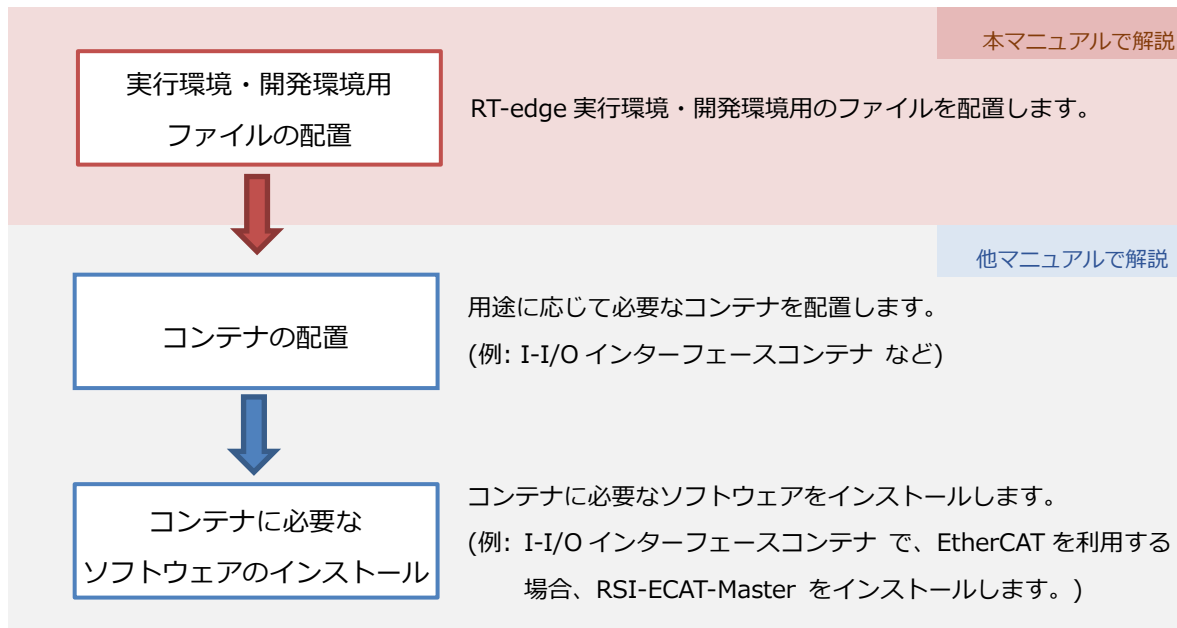
5. 開発環境・実行環境の構築

ここでは RT-edge を用いて開発環境や実行環境を構築する方法について説明します。

RT-edge 一式には RT-edge 開発環境ライブラリと RT-edge 実行環境モジュールがあります。それぞれについて説明します：

5.1. 環境構築の流れ

RT-edge の環境を構築するには、次の流れを実施します。



5.2. 開発環境の構築

開発環境には、開発用のライブラリやフレームワークを含んだ開発環境用ファイルの配置と、動作を確認する目的で実行環境用のファイルを配置します。

5.2.1. 必要ソフトウェアのインストール

RT-edge システムを開発、実行するには以下のソフトウェアをインストールする必要があります。

表 17 インストールソフトウェア一覧

ソフトウェア名	内容
INtime for Windows	Ver 6.4.21125.1 以降※
トレサブルコントローラ	Ver 6.4.21125.1.1 以降※
.NET Framework	Ver 4.6
その他	利用するコンテナのドキュメントを参照してください。



※トレサブルコントローラは、INtime for Windows より後にインストールする必要があります。また INtime のバージョンに対応したトレサブルコントローラをインストールする必要があります。

5.2.2. 実行環境/開発環境用ファイルの配置

RT-edge 実行環境モジュールを含んだフォルダー一式を任意のディレクトリへ展開します。

フォルダ	フォルダ名	説明
(CD-ROM)¥01_RTedgeFiles	RTedge	実行環境/開発環境用ファイル

以下は展開後のファイルを、C ドライブ直下の RTedge フォルダへ展開した場合の bin フォルダのイメージ図です。

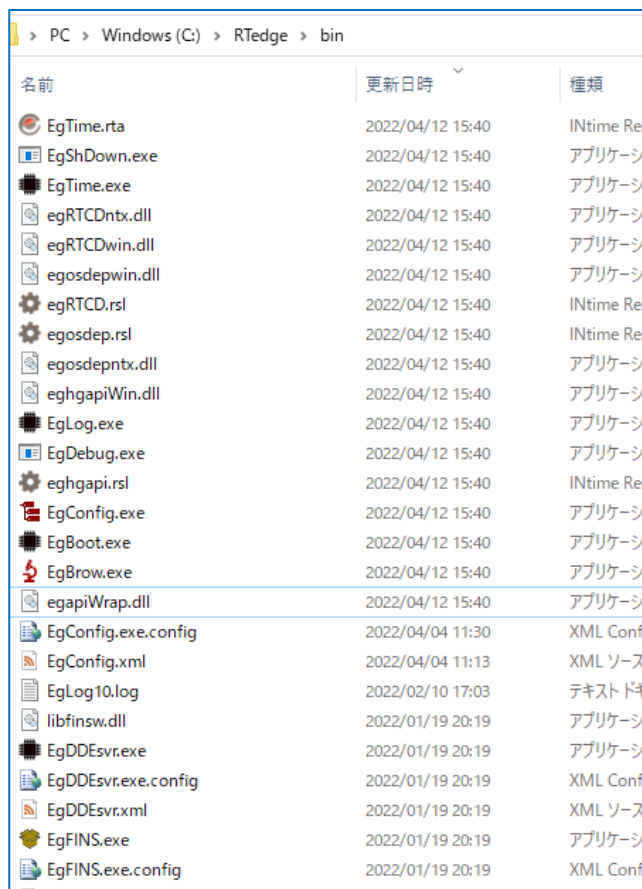


図 24 配置例

5.2.3. Visual Studio ランタイムライブラリのインストール

本フレームワークを実行するには、Visual Studio ランタイムライブラリをインストールする必要があります。下記のインストーラを実行し、インストールを行ってください。

フォルダ	ファイル名	説明
(CD-ROM)¥03_Others	VC2012redist_x64.exe	Microsoft Visual C++ 2012 Redistributable(x64)
(CD-ROM)¥03_Others	VC2017redist_x64.exe	Microsoft Visual C++ 2017 Redistributable(x64)

5.2.4. サンプル/サービス作成用テンプレートのインストール

サンプルプログラム、およびサービス作成用テンプレートを含んだ ZIP ファイル (RTedgeSample[バージョン番号].zip)を任意のディレクトリへ解凍します。サンプル/サービスコンテナ作成用テンプレートの使用方法は、サービスコンテナ作成マニュアル (DOCTREDGESRV.pdf)を参照してください。

フォルダ	ファイル名	説明
(CD-ROM)¥02_RTedgeSample	RTedgeSample***.zip	サンプル/サービスコンテナ作成用テンプレート

5.3. 実行環境の構築

実行環境/開発環境用ファイルを配置後、実行環境に不要なファイルを削除します。

5.3.1. 必要ソフトウェアのインストール

RT-edge システムを実行するには以下のソフトウェアをインストールする必要があります。

表 18 インストールソフトウェア一覧

ソフトウェア名	内容
INtime for Windows ランタイム	Ver 6.4.21125.1 以降※
トレサブルコントローラ ランタイム	Ver 6.4.21125.1.1 以降※
.NET Framework	Ver 4.6
その他	利用するコンテナのドキュメントを参照してください。



※ INtime のバージョンに対応したトレサブルコントローラをインストールする必要があります。

5.3.2. 実行環境/開発環境用ファイルの配置

RT-edge 実行環境モジュールを含んだフォルダー一式を任意のディレクトリへ展開します。

フォルダ	フォルダ名	説明
(CD-ROM)¥01_RTedgeFiles	RTedge	実行環境/開発環境用ファイル

以下は展開後のファイルを、C ドライブ直下の RTedge フォルダへ展開した場合の bin フォルダのイメージ図です。

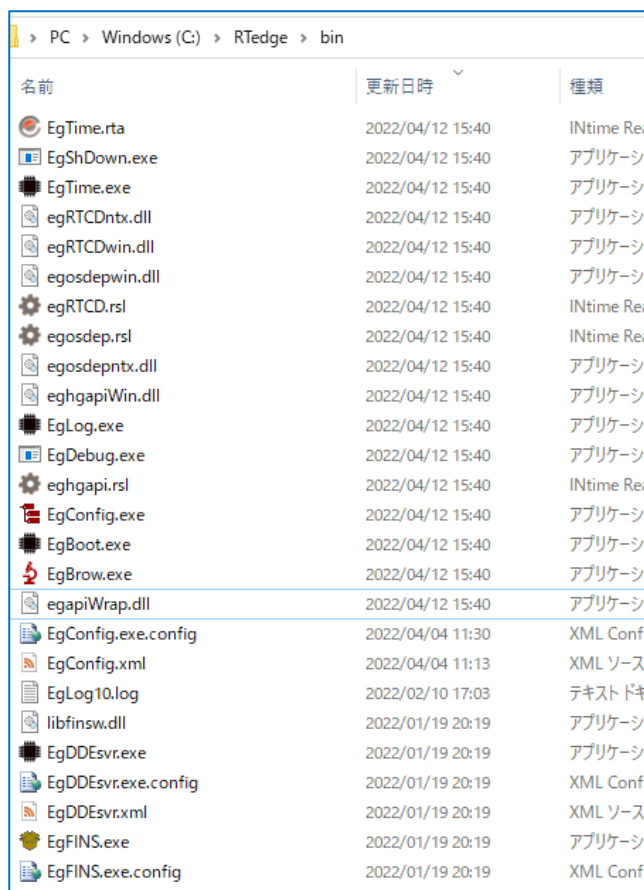


図 25 配置例

5.3.3. Visual Studio ランタイムライブラリのインストール

本フレームワークを実行するには、Visual Studio ランタイムライブラリをインストールする必要があります。下記のインストーラを実行し、インストールを行ってください。

フォルダ	ファイル名	説明
(CD-ROM)¥03_Others	VC2012redist_x64.exe	Microsoft Visual C++ 2012 Redistributable(x64)
(CD-ROM)¥03_Others	VC2017redist_x64.exe	Microsoft Visual C++ 2017 Redistributable(x64)

5.3.4. 不要ファイルの削除

展開先にある、表 19 開発環境ファイルのファイル、フォルダを削除します。

表 19 開発環境ファイル

フォルダ階層	ファイル名	説明
RT-edge¥Library¥	edgeAPI_RTCD.h	RT-edge 開発環境 API ヘッダ
RT-edge¥Library¥	egAPI.h	RT-edge 開発環境 API ヘッダ
RT-edge¥Library¥RT¥	eghgapi.lib	RT-edge 開発環境 API ライブラリ(INtime)
RT-edge¥Library¥RT¥	egosdep.lib	RT-edge 開発環境 API ライブラリ(INtime)
RT-edge¥Library¥RT¥	egRTCD.lib	RT-edge 開発環境 API ライブラリ(INtime)
RT-edge¥Library¥WIN¥	eghgapiWin.lib	RT-edge 開発環境 API ライブラリ(Windows)
RT-edge¥Library¥WIN¥	egosdepntx.lib	RT-edge 開発環境 API ライブラリ(Windows)
RT-edge¥Library¥WIN¥	egosdepwin.lib	RT-edge 開発環境 API ライブラリ(Windows)

フォルダ階層	ファイル名	説明
RT-edge¥Library¥WIN¥	egRTCDntx.lib	RT-edge 開発環境 API ライブラリ(Windows)
RT-edge¥Library¥WIN¥	egRTCDwin.lib	RT-edge 開発環境 API ライブラリ(Windows)

5.4. 開発環境・実行環境の削除

配置時に展開した各ファイル、および、フォルダを削除してください。

その後、Microsoft Visual C++ 2012 Redistributable(x64)、および、Microsoft Visual C++ 2017 Redistributable(x64)をアンインストールしてください。

5.5. 製品動作設定(実行環境設定)

5.5.1. 必要ソフトウェアのインストール

RT-edge システムを実行するには以下のソフトウェアをインストールする必要があります。

表 20 インストールソフトウェア一覧

ソフトウェア名	内容
INtime for Windows	INtime 実行環境、Ver 6.4.21125.1 以降※
トレーサブルコントローラ	トレーサブルコントローラ、Ver 6.4.21125 以降※
.NET Framework	Ver 4.6
Microsoft Visual C++ 2017 ランタイムライブラリ	x64
その他	利用するコンテナのドキュメント等を参照してください。



※トレーサブルコントローラは、INtime for Windows より後にインストールする必要があります。また INtime のバージョンに対応したトレーサブルコントローラをインストールする必要があります。

5.5.2. 実行ファイルの配置

お客様が作成したアプリケーション/コンテナを、RT-edge システムを起動するための EgBoot コンテナ(EgBoot.exe)と同じディレクトリ(本製品をインストールしたディレクトリ)に配置してください。

5.5.3. ECI ファイルの編集

5.5.3.1. EgBoot.xml の修正

本製品をインストールしたディレクトリ内の EgBoot.xml を編集します。以下の情報を編集してください。

- 1) EgBoot 固有設定 (必要に応じて)
- 2) EgTag 情報
- 3) Dataset 情報 (必要に応じて)
- 4) Service 情報(起動するサービスコンテナを追加)

① Service の起動パスを記載してください。

② Service の起動 INtime ノード名を記載してください。

(必要に応じて)INtime アプリケーションを指定の INtime ノードで実行させることが可能です。

定義例) <Service Name="IoSmpl" Path="IoSmpl.rta" Node="NodeB"/>

③ TagRef 情報、Collector 情報、CollectorRef 情報等必要に応じて追加してください。

④ Service ごとの詳細な情報を EgBoot.xml に記載しても良いですし、サービスコンテナ固有の ECI ファイルへ記載しても構いません。

以下、サンプルプロジェクト内の EgBoot.xml 構成とサービスコンテナごとの ECI ファイルの構成例です

例 1. EgBoot.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RTedge xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Tags>
    <Tag Name="EDGESYSTEM.CurrentTime" Type="9" Comment="RT-edge Time"/>
    <Tag Name="EDGESYSTEM.CurrentTime_ms" Type="5" Comment="RT-edge Time ms"/>
    <Tag Name="EDGESYSTEM.CurrentTime_us" Type="5" Comment="RT-edge Time us"/>
    <Tag Name="EDGESYSTEM.OperatingTime" Type="9" Comment="RT-edge Operating Time"/>
    <Tag Name="Temp001" Type="6" Address="%D0" Comment="温度 1 オフセット 0,4 バイト"/>
    <Tag Name="Temp002" Type="11" Address="%L4" Comment="温度 2 オフセット 4,8 バイト"/>
    <Tag Name="Temp003" Type="10" Address="%D12" Comment="温度 3 オフセット 12,4 バイト"/>
  </Tags>
  <Datasets>
    <Dataset Name="Dset1">
      <TagRefs>
        <TagRef Name="Temp001"/>
        <TagRef Name="Temp002"/>
        <TagRef Name="Temp003"/>
      </TagRefs>
    </Dataset>
  </Datasets>
  <Services>
    <Service Name="EgBoot"/>
    <Service Name="EgLog" Path="EgLog.exe" Argument="DispNumMax=500"/>
    <Service Name="EgTime" Path="EgTime.exe"/>
    <Service Name="IoSmpl" Path="IoSmpl.rta"/>
    <Service Name="ClctSmpl" Path="ClctSmpl.rta"/>
    <Service Name="DispSmpl" Path="DispSmpl.exe"/>
  </Services>
</RTedge>
```

例 2. ClctSmpl.xml

```

<?xml version="1.0" encoding="utf-8"?>
<RTedge xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Tags>
    <Tag Name="SERVICE.ClctSmpl.Cycle" Type="7" Comment="ClctSmpl In/Out Thread Interval" Value="100"/>
    <Tag Name="SERVICE.ClctSmpl.InPriority" Type="3" Comment="ClctSmpl In Thread Priority" Value="172"/>
    <Tag Name="SERVICE.ClctSmpl.OutPriority" Type="3" Comment="ClctSmpl Out Thread Priority" Value="171"/>
    <Tag Name="SERVICE.ClctSmpl.Mode" Type="3" Comment="ClctSmpl Mode" Value="1"/>
    <Tag Name="SERVICE.ClctSmpl.AutoRun" Type="1" Comment="ClctSmpl Thread Auto Run Mode" Value="0"/>
  </Tags>
  <Services>
    <Service Name="ClctSmpl">
      <TagRefs_OUT>
        <TagRef Name="Temp001"/>
        <TagRef Name="Temp002"/>
        <TagRef Name="Temp003"/>
      </TagRefs_OUT>
      <Collectors>
        <Collector Name="ClctIO" Interval="100" Priority="140" DatasetName="Dset1"/>
      </Collectors>
      <MailBoxes>
        <MailBox Name="CLCTMBOX" ColSize="2048" RowSize="1024"/>
      </MailBoxes>
    </Service>
  </Services>
</RTedge>

```

5.5.3.2. その他 ECI ファイルの編集

EgBoot.xml へ登録したサービスコンテナに関して、サービスコンテナ固有の ECI ファイルを編集します。編集方法は EgBoot.xml と同様です。



管理上等の理由で EgBoot.xml とは別に個別で編集を行いたい場合は、サービスコンテナ固有の ECI ファイルを用意してください。個別に用意する必要が無い場合は、サービスコンテナ固有の ECI ファイルを用意せずに EgBoot.xml に全ての定義を記載してください。

5.5.4. RT-edge 起動スクリプトについて

RT-edge システムの実行は、(RT-edge インストールパス)¥bin¥StartEdge.bat を実行することで開始します。本項では StartEdge.bat について説明します。

5.5.4.1. デフォルト状態の解説

以下にデフォルト状態の StartEdge.bat について説明します。StartEdge.bat の処理は関連するドライバやプロセス類の起動パラメータ設定部とアプリ起動部のふたつに分けられます。まずは起動パラメータ設定部について説明します。

設定するパラメータは INtime、EtherCAT ドライバ、INplc エンジンの 3 つがあり、デフォルトではすべて起動する設定となっています。設定内容につきましては、ユーザーによって変更可能な項目に絞って 5.5.4.2 にて詳細な解説を行います。

@echo off @SET INTIME_USE=1 @SET NODENAME=NodeA @SET LOADER="%INTIME%¥bin¥ldrta.exe" @SET WAITCTG="%INTIME%¥bin¥waitfor.exe"	} INtime 起動設定
REM EtherCAT Driver Load Setting @SET ECAT_USE=1 @SET ECAT_DRV="%RSIECAT%¥bin¥RtEcHdr.rta" @SET ECAT_ARG="-cycle=eni" @SET ECAT_INIT="PRCS_RTECHDR" @SET ECAT_WAIT=32	} EtherCAT ドライバ 起動設定
REM INplc engine Load Setting @SET INPLC_USE=1 @SET INPLC_DRV="%INplcRT%INpMgr.rta" @SET INPLC_ARG="" @SET INPLC_INIT="ProConOS" @SET INPLC_WAIT=32	} INplc エンジン 起動設定
REM Waiting for RT-edge initialization to finish setting @SET EDGE_ENDINIT="EG?END_INIT" @SET EDGE_WAIT=60	} RT-edge 初期化 終了待

図 26 RT-edge 起動スクリプト-起動パラメータ設定部

続いて、アプリ起動部について示します。

デフォルトでは INtime→EtherCAT ドライバ→RT-edge→INplc エンジンの順に起動する設定となっています。(これらの起動順は変更しないことを推奨します。) ユーザーアプリの起動を追加したい場合は以下のユーザーアプリ起動追加箇所①～③に記述を追加します。それぞれの箇所で起動可能なアプリの種類と記述方法につきましては 5.5.4.2 以降で説明します。

REM Pre-start INtime Kernel processing... REM TODO... REM	}	ユーザー アプリ起動 追加箇所①
REM Start INtime Kernel if %INTIME_USE% neq 0 (@CALL "StartRTK.bat" %NODENAME% -v)	}	INtime 起動処理
REM INtime Kernel started processing... REM TODO... REM	}	ユーザー アプリ起動 追加箇所②
REM Start EtherCAT Driver if %ECAT_USE% neq 0 (@%LOADER% %ECAT_DRV% -a %ECAT_ARG% @%WAITCTG% -node %NODENAME% -wait %ECAT_WAIT% %ECAT_INIT%)	}	EtherCAT ドライバ 起動処理
REM Start RT-edge boot sequence @CD "%RTEDGE%¥bin" @START EgBoot.exe @%WAITCTG% -node %NODENAME% -wait %EDGE_WAIT% %EDGE_ENDINIT%	}	RT-edge 起動処理
REM Start INplc engine if %INPLC_USE% neq 0 (@%LOADER% %INPLC_DRV% -a %INPLC_ARG% @%WAITCTG% -node %NODENAME% -wait %INPLC_WAIT% %INPLC_INIT%)	}	INplc エンジン 起動処理
REM REM TODO... REM	}	ユーザー アプリ起動 追加箇所③

図 27 RT-edge 起動スクリプト-アプリ起動部

5.5.4.2. RT-edge 起動スクリプトの内容変更

RT-edge 起動スクリプトの内容を変更する際は、以下ファイルをメモ帳などのテキストエディタで開いてください。

(RT-edge インストールパス)¥bin¥StartEdge.bat

まずは起動パラメータ設定部の変更可能な箇所と内容について表 21～表 23 で説明します。

表 21 設定可能な INtime 起動時パラメータ

パラメータ	説明	設定値	デフォルト値
INTIME_USE	INtime を起動するかを設定します。 RT-edge の構成が Windows と INtime のハイブリット構成の場合は起動を有効にしてください	0 (起動しない) 1 (起動する)	1 (起動する)
NODENAME	RT-edge の INtime 用プロセスを動作させるノードを設定します。	例)NodeA NodeB	NodeA

表 22 設定可能な EtherCAT ドライバ起動時パラメータ

パラメータ	説明	設定値	デフォルト値
ECAT_USE	EtherCAT ドライバを起動するかを設定します。 サービスコンテナで EtherCAT を利用する場合は起動を有効にしてください。	0 (起動しない) 1 (起動する)	1 (起動する)
ECAT_ARG	EtherCAT ドライバの起動引数を設定します。 設定可能な引数の詳細は EtherCAT ドライバのマニュアルを参照してください。	例) -cycle=eni -mode=std	-cycle=eni

表 23 設定可能な INplc エンジン起動時パラメータ

パラメータ	説明	設定値	デフォルト値
INPLC_USE	INplc エンジンを起動するかを設定します。 サービスコンテナでソフトウェア PLC プログラムを利用する場合は起動を有効にしてください。	0 (起動しない) 1 (起動する)	1 (起動する)

続いて、ユーザーアプリの起動設定の追加方法です。図 27 のユーザーアプリ起動追加箇所①～③に示しました通り、ユーザーアプリ起動のタイミングは 3 か所あります。それぞれのタイミングで起動可能なアプリの種類について下表で説明します。

タイミング	説明	起動可能なアプリ
INtime 起動前	INtime 起動より先に実行したいユーザーアプリを記載します。 INtime が起動していないため、Windows アプリのみ記載可能です。	exe (Windows アプリ)
INtime 起動後	INtime 起動後に実行したいユーザーアプリを記載します。 INtime アプリ、Windows アプリのいずれも記載可能ですが、EtherCAT ドライバを利用して動作する INtime アプリの記載はできません。	exe (Windows アプリ) rta (INtime アプリ)
RT-edge 起動後	RT-edge 起動後に実行したいユーザーアプリを記載します。 INtime アプリ、Windows アプリのいずれも記載可能です。	exe (Windows アプリ) rta (INtime アプリ)

アプリ起動を追加するための書式については、5.5.4.2.1、5.5.4.2.2 にて説明します。

5.5.4.2.1.exe ファイル起動設定

exe ファイル(Windows アプリ)を起動する際は以下のような書式で RT-edge 起動スクリプトへの追記を行います。以下の中括弧部分を編集してください。

```
@CD "{起動したい exe ファイルの存在するフォルダパス}"
@START {exe ファイル名}
```

5.5.4.2.2.rta ファイル起動設定

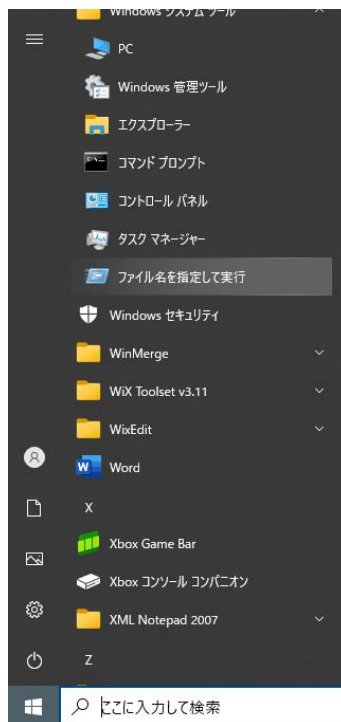
rta ファイル(INtime アプリ)を起動する際は以下のような書式で RT-edge 起動スクリプトへの追記を行います。以下の中括弧部分を編集してください。

```
@%LOADER% {起動したい rta ファイル名} -a {起動引数}
@%WAITCTG% -node %NODENAME% -wait {起動待機時間[ms]} {ハンドル名}
```

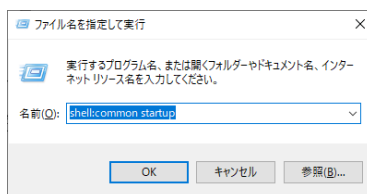
5.5.4.3. スタートアップへの登録と有効/無効化

StartEdge.bat をスタートアップに登録することで、コントローラの起動と同時に RT-edge システムを起動することができます。スタートアップへの登録方法は以下の通りです。

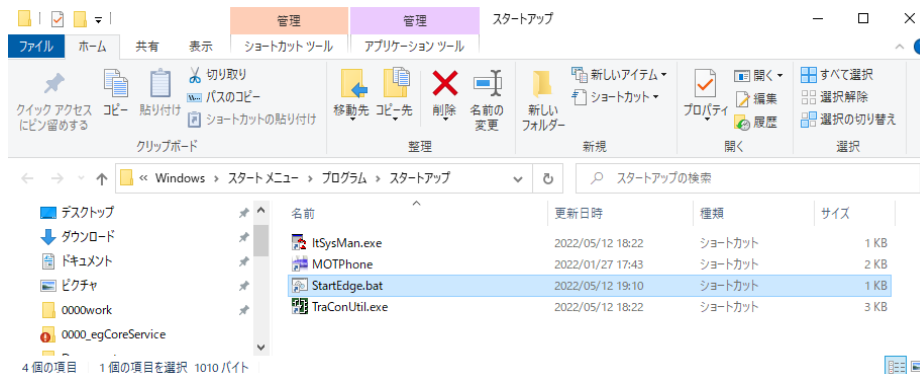
- 1) 「メニュー」→「Windows システムツール」→「ファイル名を指定して実行」を選択します。



- 2) 「ファイル名を指定して実行」ダイアログの名前に[shell:common startup]と入力し、OK ボタンを押します。



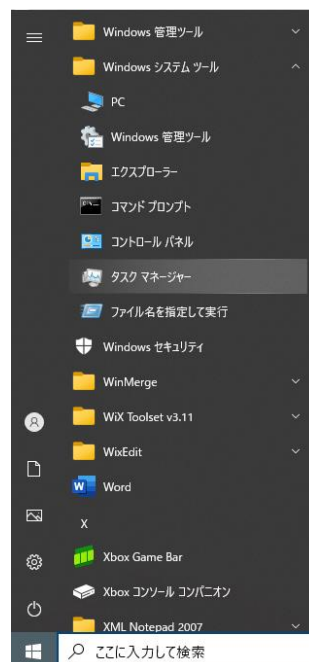
- 3) エクスプローラが開き、「スタートアップ」フォルダの内容が表示されますので、ここに StartEdge.bat のショートカットを配置します。



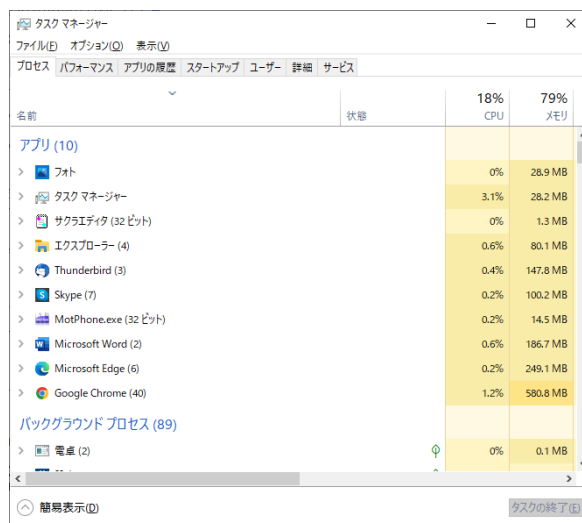
以上で、StartEdge.bat のスタートアップ登録は完了です。本設定で次回以降のコントローラ起動時に RT-Edge が自動起動するようになります。

続いて、スタートアップの有効/無効化方法について説明します。(スタートアップ登録後はデフォルトで有効となっています。)

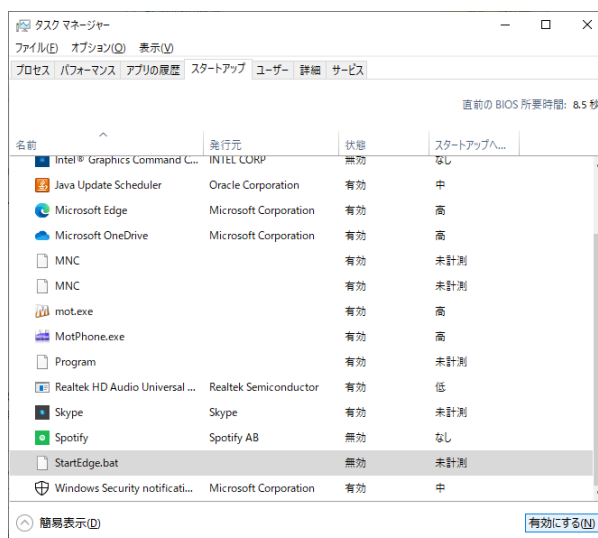
- 1) 「メニュー」→「Windows システムツール」→「タスクマネージャー」を選択します。



- 2) タスクマネージャーが起動しますので、「スタートアップ」タブを選択します。

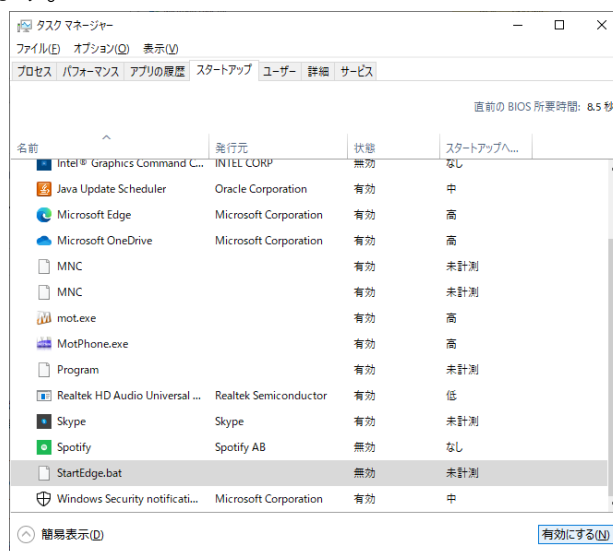


- 3) 登録されているスタートアップの中から StartEdge.bat を選択し、右下の「有効にする (A)」ボタンをクリックします。



4) StartEdge.bat の状態が「有効」になったことを確認します。

無効化したい場合は、有効状態の StartEdge.bat を選択し、右下の「無効にする(A)」ボタンをクリックします。



以上で、スタートアップ有効/無効化の作業は完了です。

5.5.5. RT-edge システム動作確認

RT-edge システムが正常に起動することを確認します。

- 1) EgBoot サービスコンテナが正常に起動しているか確認します。EgBoot サービスコンテナは起動後タスクトレイに格納されます。タスクトレイに格納されていることを確認してください。



図 28 EgBoot タスクトレイ

- 2) (RT-edge インストールパス)¥EgBrow.exe を起動します。
- 3) サービスコンテナインジケータタグから、各サービスコンテナが正常状態であることを確認します。以下では EgBoot サービスコンテナが正常であることを確認している例です。他サービスコンテナに関しても正常に動作している状態かどうか、同様の方法(各サービスコンテナインジケータタグ)で確認してください。

正常状態

- SERVICE.EgBoot.Status が **1** であること
- SERVICE.EgBoot.Run が **1 / True** であること
- SERVICE.EgBoot.Error が **0 / False** であること
- SERVICE.EgBoot.Live が **増加**していくこと

RT-edge Object Browser				
Tags Containers Collectors Datasets Mailboxes				
Name	CurrentValue	Type	Source	Comment
SERVICE.EgCore	00000000 (0)	Int32		
SERVICE.EgCore.Status	01 (1)	byte		
SERVICE.EgCore.Error	false (0)	bool		
SERVICE.EgCore.Run	true (1)	bool		
SERVICE.EgCore.Live	00001125 (4389)	UInt32		


図 29 EgBrow 起動時の様子

5.5.6. RT-edge システム終了手順

RT-edge システムの終了方法はお客様の運用方法に依存します。例として以下に終了方法を記載します。

- 1) PC(Windows)を終了/再起動する
- 2) Windows 起動時に RT-edge システムを起動し、Windows 終了時に RT-edge システムを終了する場合等アプリケーション/サービスコンテナを個別に終了する

アプリケーション毎に終了方法がある場合、個別に終了したい場合等に使用します。

- ① EgBoot サービスコンテナの終了は、タスクトレイに常駐しているアイコン  のコンテキストメニューから「終了」を選択することで終了します。
- ② サービスコンテナが終了処理(サービスコンテナ間通信処理)を実装している場合、サービスコンテナごとに、「EM_SERVICE_STOP」メッセージを送信し終了させます。詳細は「RT-

edge コンテナ作成マニュアル」を参照してください。

- 3) サービスコンテナ一括終了用ツールを使用し、アプリケーション/サービスコンテナを終了する

指定したサービスコンテナを一括で終了します。使用方法は「7.4. EgShDown サービスコンテナ」を参照してください。



PC を再起動せずに RT-edge システムを再起動(終了後、再度 EgBoot.exe を使用して起動)する場合は、EgBoot.xml に記載している Service を必ず全て終了させてから再度 EgBoot.exe から起動してください。

5.6. サンプルプログラムを使っでの動作確認

製品付属サンプル IOSample を使うことで、タグへの読み込み/書き込み処理を確認できます。サンプル処理となっていますので、動作確認及びプログラミングの仕方についての確認ができます。

詳細は、製品メディア内サンプルフォルダに配置しております IOSample プログラム取扱説明書 (DOCRTEGEDGESMP_IOSAMPLE.pdf) を参照ください。

5.7. 製品動作設定(開発環境設定)

RT-edge 開発環境ライブラリを用いたアプリケーション/サービスコンテナ作成方法・環境設定に関しては、別マニュアル「RT-edge コンテナ作成マニュアル (DOCRTEGEDGESRV.pdf)」を参照してください。

6. トラブルシューティング

6.1. エラーが発生して起動できない(INtime サービスコンテナ)

INtime 環境にて、RT-edge システム・EgService を起動時に、仮想セグメントの空き容量が足りない場合はエラーとなります。RT-edge システム内部の RTCD を生成する際、プロセスの仮想セグメントが消費しています。空き容量が足りない場合は、仮想セグメントを増やしてください。

1) 利用可能な仮想セグメントサイズの確認方法

edge API を利用するアプリケーションのプロセス

4230 (data)

(data)と表示されているメモリオブジェクトを選択します。

プロセスの仮想セグメント総サイズは、**Total address space** にて確認できます。この例では仮想セグメント総サイズが 8MB※であることを示しています。
※非 XM モードの場合の仮想セグメント総サイズのデフォルトは 8MB です。

Size	0x00800000	Address	04800000
Container proc	4208	va R/W	A P=1 Dpl=3
Total address space		8.0 M	
Currently used		1840.0 K (22.5%)	
Largest fragment		2048.0 K	

Largest fragment の値が、現在利用可能な仮想セグメントサイズです。この例では、生成する FIFO のサイズが 2048KB(2097152 バイト)以下であれば確保可能です。2048KB を超える場合(2097153 バイト以上)は、FIFO 生成時にエラーとなります。その場合は下記「仮想セグメントを増やす方法」を参照し、仮想セグメントの設定を行ってください。

図 30 利用可能な加速セグメントサイズの確認方法

2) 仮想セグメントを増やす方法

INtime アプリケーションの仮想セグメントサイズを増やす方法を以下に説明します。

- ① Visual Studio にて INtime アプリケーションのプロジェクトのプロパティ画面→[構成プロパティ]→[INtime Properties]を選択します。
- ② [Virtual segment size]に設定したい仮想セグメントサイズ(単位はバイト)を入力して下さい。例として 16MB の RTCD を 1 個生成する場合は、仮想セグメントサイズには目安として 26MB 以上を設定して下さい。また、RTCD の他にも動的にメモリを確保する場合は、そのメモリサイズも加算して仮想セグメントサイズを設定して下さい。

RTCDSample プロパティページ

構成(C): アクティブ(Debug) プラットフォーム(P): アクティブ(INtime)

General

INtime Node Local

Pool Minimum (bytes) 0

Pool Maximum (bytes) 0

Virtual segment size (bytes) 27262976

Initial thread stack size (bytes) 0

Object directory size 0

Allow execution in data segment 0

Use XM mode Do not use XM mode

仮想セグメントサイズを設定します。(単位はバイト)
デフォルトは 0 に設定されています。(デフォルトの場合、仮想セグメントサイズは 8MB となります)
この例では、仮想セグメントサイズを 26MB(27262976 バイト)に設定しています。

※左記イメージは Visual Studio 2012 の画面です

図 31 仮想セグメントの設定

6.2. エラーが発生して起動できない(.NET サービスコンテナ)

.NET で作成したサービスコンテナにおいて、サービスコンテナが参照するライブラリ「egapiWrap.dll」がビルド時のライブラリバージョンと違いがある場合起動しません。

実行環境に配置された「egapiWrap.dll」でビルドしなおすか、または、App.config 内を修正し、アプリケーションのバインド動作の変更を行ってください。

詳細はマイクロソフトの「アセンブリ バージョンのリダイレクト」ページを参照してください。

<<https://docs.microsoft.com/ja-jp/dotnet/framework/configure-apps/redirect-assembly-versions>>



上記ページが参照できない場合は、Web 上においてキーワード「アセンブリ バージョンのリダイレクト」で検索してください。

6.3. タグの生成に失敗する

RT-edge システム起動時、下記のエラー(0x8000040e)が表示され、タグの生成に失敗する場合は、[EgBoot 固有設定](#)を参照し、タグの最大生成数を変更してください。

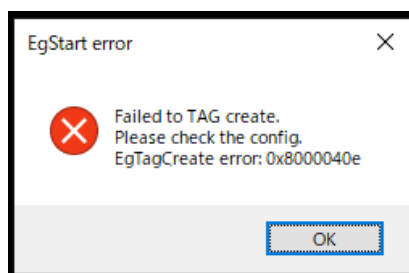


図 32 タグの生成エラーダイアログ



EgTag オブジェクトは、タグ名称から生成したハッシュコードを利用しているため、タグの最大生成数未満でも生成に失敗する場合があります。

上記以外のエラーでタグ生成に失敗する場合は、ECI ファイルの Tag エLEMENT の定義に誤りがない確認してください。

7. 付録

7.1. RT-edge オブジェクトブラウザ

RT-edge オブジェクトブラウザ(EgBrow.exe)を起動することにより、RT-edge のシステム情報、ECI ファイルで定義したタグなどの現在値を参照することができます。

また、定義済みのタグに対して、値変更が可能です。

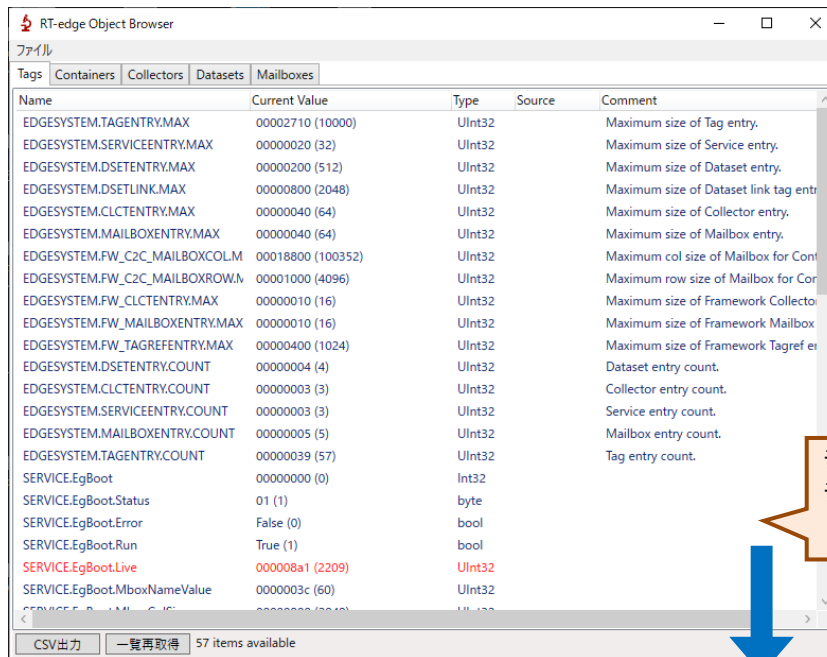
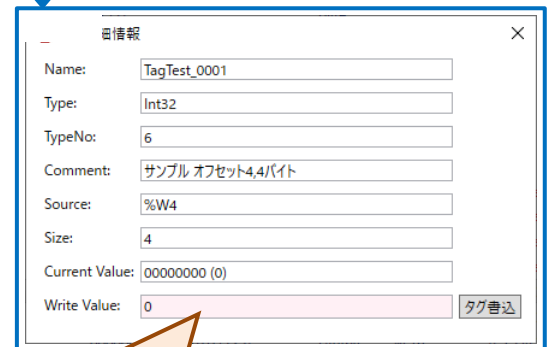


図 33 RT-edge オブジェクトブラウザ

データをダブルクリックすると、選択したデータの詳細情報が表示されます。
※Tags/Datasets タブのみ



タグ値のみ変更可能です。
Write Value に値をセットし、タグ書込ボタンをクリックしてください。

7.1.1. Tags 表示タブ

タグ情報を一覧表示します。

表 24 Tag 表示タブ表示項目一覧

項目名	表示内容	説明
Name	タグ名称	タグの名称を表示します。
Current Value	現在値	現在値を表示します。
Type	変数の型	変数の型を表示します。 byte、Int16、Int64、float、double など
TypeNo	データタイプ	データタイプ (EgType 型の定義 No) を表示します。
Source	タグのソース	タグのソースを表示します。
Comment	コメント文字列	コメント文字列を表示します。
Size	データサイズ	データサイズ (バイト単位) を表示します。

7.1.2. Containers 表示タブ

サービスコンテナ情報を一覧表示します。

表 25 Containers 表示タブ表示項目一覧

項目名	表示内容	説明
Name	サービスコンテナ名称	サービスコンテナの名称を表示します。
Status	サービスコンテナ起動状態	現在のサービスコンテナ起動状態を表示します
Error	サービスコンテナエラー状態	現在のサービスコンテナエラー状態を表示します。
Run	サービスコンテナ実行状態	現在のデーターリフレッシュ動作の状態を表示します。
Live	サービスコンテナ実行カウンタ	サービスコンテナが健全であることを示すカウンタを表示します
Path	実行モジュールファイルパス	実行モジュールファイルパスを表示します
Arg	起動引数	サービスコンテナの起動引数を表示します。
NodeName	起動ノード名	サービスコンテナが起動している INtime ノード名を表示します。INtime アプリケーションの場合のみ使用します。
MboxName	メールボックス名	サービスコンテナ起動時に自動生成されるメールボックスの名称を表示します。

7.1.3. Collectors 表示タブ

コレクタ情報を一覧表示します。

表 26 Collectors 表示タブ表示項目一覧

項目名	表示内容	説明
Name	コレクタ名称	コレクタの名称を表示します。
Type	コレクタタイプ	コレクタのスレッドタイプを表示します
OwnerName	コレクタの収集を開始するサービスコンテナ名	コレクタの収集を開始するサービスコンテナ名を表示します。
Interval	収集周期	コレクタの収集周期を表示します。単位:ms
Priority	コレクタスレッドのプライオリティ	コレクタスレッドのプライオリティを表示します。 非リアルタイムスレッドの場合は、ハイフン表示。
ColSize	レコードデータサイズ	収集データのレコードデータサイズを表示します。

RowSize	レコード数	収集データのレコード数を表示します。
DatasetName	データセット名	収集データのデータセット名を表示します。
WritePos	データ書き込み位置	収集データの書き込み位置を表示します。
Queue	データキューイング数	収集データの現在のキューイング数を表示します。

7.1.4. Datasets 表示タブ

データセット情報を一覧表示します。

表 27 Datasets 表示タブ表示項目一覧

項目名	表示内容	説明
Dataset	データセット名称	データセットの名称を表示します。
Name	タグ名称	タグの名称を表示します。
Current Value	現在値	現在値を表示します。

7.1.5. Mailboxes 表示タブ

メールボックス情報を一覧表示します。

表 28 Mailboxes 表示タブ表示項目一覧

項目名	表示内容	説明
Name	メールボックス名称	メールボックスの名称を表示します。
ColSize	レコードデータサイズ	メールボックスデータの 1 レコードデータサイズを表示します。
RowSize	レコード数	メールボックスデータのレコード数を表示します。
WritePos	データ書き込み位置	メールボックスデータの書き込み位置を表示します。
Queue	データキューイング数	メールボックスデータの現在のキューイング数を表示します。

7.1.6. 更新周期設定

各タブに表示されるデータの更新周期を設定可能です。

ファイルメニューから「更新周期設定」を選択し、更新周期設定ダイアログにてタブ毎の値を設定してください。(初期値:1000ms)

更新周期設定 X

Tags [ms]

Containers [ms]

Collectors [ms]

Datasets [ms]

Mailboxes [ms]

閉じる

図 34 更新周期設定ダイアログ

7.1.7. CSV 出力

表示中タブのデータを CSV 形式でファイル出力することが出来ます。

ウィンドウ左下の「CSV 出力」ボタンをクリックしてください。

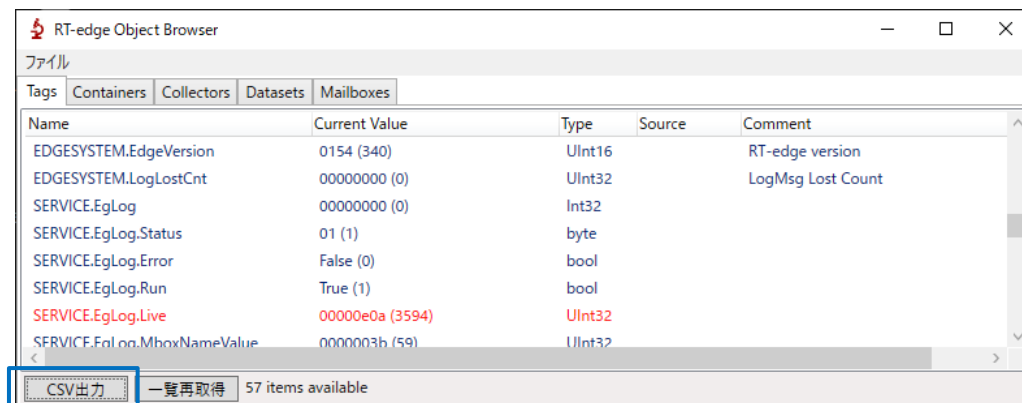


図 35 CSV 出力

ファイルは本ツール(EgBrow.exe)の起動ディレクトリに下記のファイル名で出力されます。

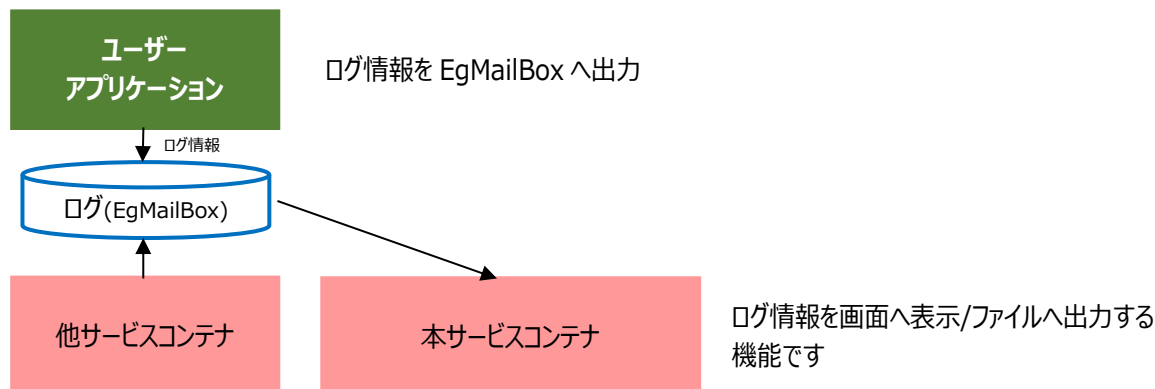
表 29 CSV 出力ファイル名一覧

No	選択中のタブ	出力ファイル名
1	Tags	Dump_Tags_yyyyMMddHHmmss.csv
2	Containers	Dump_Containers_yyyyMMddHHmmss.csv
3	Collectors	Dump_Collectors_yyyyMMddHHmmss.csv
4	Datasets	Dump_Datasets_yyyyMMddHHmmss.csv
5	Mailboxes	Dump_Mailboxes_yyyyMMddHHmmss.csv

※yyyyMMddHHmmss には出力時点の日時分秒が入ります。

7.2. EgLog ログサービスコンテナ

RT-edge 内で出力されたログは EgMailbox(EGSYSLOG)に格納されます。格納されたログ情報を取得し、受信したログを画面上の時系列リストに表示(ファイル出力)するログサービスコンテナ(EgLog)を提供します。



Log File Path: C:\EdgeSystem\EgLog28.log

Date	Writer	Service	Kind	Message
2021/01/28 11:36:14:292	EGSYS	DispSmpl	OPE	Add EgTagRef OUT to Service SUCCESS. Name: Temp003, State: I
2021/01/28 11:36:14:292	EGSYS	DispSmpl	MSG	[MessageNo: 121] [Sender: EgCore]
2021/01/28 11:36:14:292	EGSYS	DispSmpl	OPE	Add EgCollectorRef to Service SUCCESS. Name: CldtID, DatasetNa
2021/01/28 11:36:14:292	EGSYS	DispSmpl	MSG	[MessageNo: 1003] [Sender: EgCore]
2021/01/28 11:36:14:292	EGSYS	DispSmpl	MSG	[MessageNo: 1002] [Sender: EgCore]
2021/01/28 11:36:14:293	EGSYS	(null)	DBG	InitMember Start
2021/01/28 11:36:14:295	EGSYS	DispSmpl	DBG	InitMember End
2021/01/28 11:36:14:295	EGUSR	DispSmpl	DBG	End plintEndFunc
2021/01/28 11:36:14:308	EGUSR	DispSmpl	DBG	An error occurred in EgTagRead[SERVICE.DispSmpl AutoRun]. Statu
2021/01/28 11:36:14:308	EGUSR	DispSmpl	DBG	Auto Run False
2021/01/28 11:36:14:308	EGSYS	DispSmpl	DBG	End Initialize
2021/01/28 11:36:14:308	EGUSR	DispSmpl	DBG	End Win Initialize
2021/01/28 11:36:14:308	EGSYS	N/A	DBG	End Win Initialize
2021/01/28 11:37:37:101	EGSYS	N/A	DBG	Add Initialize End Function
2021/01/28 11:37:37:166	EGUSR	(null)	DBG	Start Win Initialize
2021/01/28 11:37:37:167	EGSYS	DispSmpl	DBG	Start Initialize
2021/01/28 11:37:37:167	EGSYS	DispSmpl	DBG	Process Argument = "C:\EdgeSystem\DispSmpl.exe"
2021/01/28 11:37:37:167	EGSYS	DispSmpl	DBG	Add User Message Handler Function
2021/01/28 11:37:37:167	EGSYS	DispSmpl	ERR	Cannot Create Own EgMailBox[DispSmpl]
2021/01/28 11:37:38:275	EGSYS	DispSmpl	ERR	Error MBoxHandle. [FUNC = _EgFWInitialize]

表 30 EgLog サービス概要

7.2.1. 機能

1) ログの表示機能

- ① ログサービスコンテナ画面へログ情報を表示します。画面を表示させるか否かは、タスクトレイに常駐している EgLog サービスコンテナアイコンで指定が可能です。
- ② 画面へのログ表示上限数は、EgLog.exe の起動引数指定により変更可能です。表示上限数を超過する場合、最古のログが表示リストから削除されます。(デフォルト:100 件)

```
<Service Name="EgLog" Path="EgLog.exe" Argument="DispNumMax=100">
...
</Service>
```

表示上限数は 1～10000 件まで指定できます。

(10001 以上も指定できますが、表示件数が大きければ大きいほど多くのメモリをし、OS や他のアプリケーションに影響を及ぼすため推奨しません)

- ③ エラー("ERR")のメッセージは赤色で表示します。
- ④ 画面への表示項目は以下のとおりです。

表 31 表示項目

項目名	表示内容	説明
Date	出力に日時	タグの名称を表示します。
Writer	ログ書き込み者	RT-edge フレームワーク内やシステムで出力しているログは"EGSYS"として表示します。
Service	ログ書き込みサービスコンテナ名	ログを書き込んでいるサービスコンテナ名を表示します。
Kind	メッセージタイプ	メッセージタイプを表示します。システムで定義しているメッセージタイプ文字列は以下のとおりです <ul style="list-style-type: none"> ・ ERR(エラー), ・ OPE(操作) ・ MSG(メッセージ), ・ WRN(ワーニング) ・ DBG(デバッグ情報)
Message	メッセージ	メッセージを表示します。

2) ログのファイル出力

- ① ログは画面へ出力すると共にファイルへ出力します。
- ② ログ出力ファイルは、EgLog.exe と同一フォルダに "EgLog" + 日付(1～31) + ".log" の名称で作成します。1 カ月経過後はファイルをクリアしたうえで、上書き利用されます。

3) その他

出力されたログが正しく格納されなかった場合(欠損)、または一度に 100 件以上のログを要求され、システム動作に影響を避けるためにシステムが意図的に欠損させた場合は、システムタグ"EDGESYSTEM.LogLostCnt"がインクリメントされます。

7.2.2. サービスコンテナパラメータ

EgLog サービスのサービスパラメータ仕様を以下に記載します。

1) サービスコンテナインジケータタグ

本サービスコンテナは以下のデータタグでサービスコンテナ状態を現示します。EgTagRead でアクセスできます。

表 32 EgLog サービスコンテナパラメータ

サービスコンテナインジケータ名	型	対応状況	説明
SERVICE.EgLog.Status	byte	対応	起動状態を示す(1=起動中/0 停止中)
SERVICE.EgLog.Run	bool	対応	エラー状態を示す(1=障害発生中/0=エラーなし)
SERVICE.EgLog.Error	bool	対応	実行状態を示す(1=実行中/0=一時停止中)
SERVICE.EgLog.Live	UInt32	対応	稼働カウンタ (起動中インクリメントされます)

2) サービスコンテナプロパティタグ

本サービスコンテナは以下のデータタグでサービスコンテナ動作設定を持ちます。動作設定は ECI ファイルで設定を定義できるほか、EgTagRead でアクセスできます。本サービスコンテナでは全て非対応です。

表 33 EgLog サービスコンテナプロパティタグ

サービスコンテナプロパティ名	型	対応状況	説明
SERVICE.EgLog.AutoRun	bool	非対応	自動起動指定機能 本サービスコンテナは常に自動開始
SERVICE.EgLog.Mode	byte	非対応	動作モード指定 本サービスコンテナは動作モード無
SERVICE.EgLog.Cycle	UInt32	非対応	処理周期指定 本サービスコンテナは周期処理無
SERVICE.EgLog.InPriority	byte	非対応	入力サービスコンテナの処理優先度 本サービスコンテナは無
SERVICE.EgLog.OutPriority	byte	非対応	出力サービスコンテナの処理優先度 本サービスコンテナは無

3) サービスコンテナデータタグ

本サービスコンテナは以下のデータタグの情報を Read/Write します。

表 34 EgLog サービスコンテナデータタグ

タグ名 (edge への入力[IN]方向)	型	範囲	説明
EDGESYSTEM.LogLostCnt	UInt32	0～	欠損カウント機能
タグ名 (edge への出力[OUT]方向)	型	範囲	説明
※このサービスコンテナにアウトプットはありません			

4) サービスコンテナメッセージ

本サービスコンテナはメッセージを使って外部サービスコンテナからコントロールできます。

表 35 EgLog サービスコンテナメッセージ

サービスメッセージ	番号	対応状況	説明
EM_SERVICE_STOP	101	対応	サービスコンテナを終了します。
EM_SERVICE_RUN	102	非対応	データ更新を開始します。
EM_SERVICE_PAUSE	103	非対応	データ更新を一時停止します。
EM_SERVICE_UPDATE	104	非対応	データ更新を指令します。

7.2.3. サービスコンテナ動作設定

本サービスコンテナを RT-edge システムに導入する作業は次のようになります。

1) サービスコンテナの起動登録

EgBoot.xml の Service タグへサービスコンテナ起動を定義します。

```

<...>
<RTedge>
  <...>
  <Services>
    <...>
    <Service Name="EgLog" Path="EgLog.exe"/>
    <...>
  </Services>
</RTedge>

```

2) サービスコンテナの動作設定

EgLog サービスコンテナの動作設定は表示上限数の設定のみで他の設定はありません。上記 EgBoot.xml ファイルの Service 定義時に以下のとおり Argument 属性を追加します。

```

<Service Name="EgLog" Path="EgLog.exe" Argument="DispNumMax=100">
  ...
</Service>

```

3) サービスコンテナの実行

EgBoot.exe を実行することで起動します。

7.3. EgTime 時刻サービスコンテナ

RT-edge 内で時刻情報を管理する EgTime サービスコンテナ (EgTime.rta) を提供します。

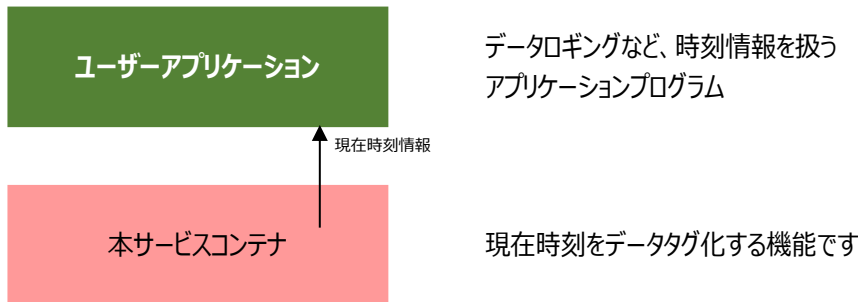


図 36 EgTime サービスコンテナ概要

7.3.1. 機能

1) 現在時刻入力機能

1970.01.01 からの経過秒/ミリ秒/μ 秒を処理周期で Tag に書き込みます。

2) システム稼働時間入力機能

RT-edge を起動してからの時間(経過秒)を処理周期で Tag に書き込みます。

3) 動作クロック指定機能

EgTime.xml のプロパティ "SERVICE.EgTime.TimeSource" で時計のクロック源を選択できます。

0 = リアルタイムクロック (INtime/Windows 時計) デフォルト

1 = INtime カーネルティックカウント



1(INtime カーネルティックカウント)は、INtime による処理回数(ティック)を時間基準にした動作で、処理する回数を基準に考える場合使用します。(例えば 1000 回処理=10 分経過とした場合、現実時間と異なることもあります。)

0(リアルタイムクロック)は 1 の逆で時間内の処理回数が 1 回少なかったり、多かったり誤差が生じます。(例えば 10 分間=999 回処理)

1(INtime カーネルティックカウント)を使用する場合は以下の「時刻リフレッシュ指令メッセージ機能」もあわせて実装してください。

4) 時刻リフレッシュ指令メッセージ機能

EgTime サービスへメッセージ「EM_SERVICE_TIMEREFRESH(19999)」を送信することで、時刻を RTC から再取得します。Windows 時刻との差が大きくなった場合に使用します。

TimeSource=1 のとき有効に作用します。

5) その他

上記は EgTime.rta の説明です。EgTime.rta とは別に、EgTime.exe があります。EgTime.exe は Windows の時刻、システム稼働時間を Tag に書き込み、画面へ表示しますが、動作クロック指定機能、サービスコンテナパラメータには対応していません。また EgTime.rta と EgTime.exe は同時に起動しないでください。

7.3.2. サービスコンテナパラメータ

EgTime(.rta)サービスのサービスパラメータ仕様を以下に記載します。

1) サービスコンテナインジケータタグ

本サービスコンテナは以下のデータタグでサービスコンテナ状態を現示します。EgTagReadでアクセスできます。

表 36 EgTime サービスコンテナパラメータ

サービスコンテナインジケータ名	型	対応状況	説明
SERVICE.EgTime.Status	byte	対応	起動状態を示す(1=起動中/0 停止中)
SERVICE.EgTime.Run	bool	対応	実行状態を示す(true=実行中/false=一時停止中)
SERVICE.EgTime.Error	bool	対応	エラー状態を示す(true=障害発生中/false=エラーなし)
SERVICE.EgTime.Live	UInt32	対応	稼働カウンタ (起動中インクリメントされます)

2) サービスコンテナプロパティタグ

本サービスコンテナは以下のデータタグでサービスコンテナ動作設定を持ちます。動作設定はECI ファイルで設定を定義できるほか、EgTagReadでアクセスできます。

表 37 EgTime サービスコンテナプロパティタグ

サービスコンテナプロパティ名	型	対応状況	説明
SERVICE.EgTime.AutoRun	bool	対応	自動起動指定機能 初期 false
SERVICE.EgTime.Mode	byte	対応	動作モード指定 SEMIAUTO(2)固定
SERVICE.EgTime.Cycle	UInt32	対応	処理周期指定機能、ミリ秒単位(1~1000) 初期 1
SERVICE.EgTime.InPriority	byte	対応	入力サービスの処理優先度 (1~254) 初期 148
SERVICE.EgTime.OutPriority	byte	非対応	出力サービスの処理優先度 本サービスは無
SERVICE.EgTime.TimeSource	Byte	対応	動作クロック選択機能(0=RTC, 1=KernelTick) 初期 0

3) サービスコンテナデータタグ

本サービスコンテナは以下のデータタグの情報を Read/Write します。

表 38 EgTime サービスコンテナデータタグ

タグ名 (edge への入力[IN]方向)	型	範囲	説明
EDGESYSTEM.CurrentTime	UInt64	0~	現在時刻 1970.01.01 からの経過秒を格納します。
EDGESYSTEM.CurrentTime_ms	UInt16	0~999	現在時刻ミリ秒を格納します。
EDGESYSTEM.CurrentTime_us	UInt16	0~999	現在時刻マイクロ秒を格納します。
EDGESYSTEM.OperatingTime	UInt64	0~	システム稼働時間秒数を格納します。
タグ名 (edge への出力[OUT]方向)	型	範囲	説明
※このサービスにアウトプットはありません。			

4) サービスコンテナメッセージ

本サービスコンテナはメッセージを使って外部サービスコンテナからコントロールできます。

表 39 EgTime サービスコンテナメッセージ

サービスメッセージ	番号	対応状況	説明
EM_SERVICE_STOP	101	対応	サービスコンテナを終了します。
EM_SERVICE_RUN	102	対応	データ更新を開始します。
EM_SERVICE_PAUSE	103	非対応	データ更新を一時停止します。
EM_SERVICE_UPDATE	104	非対応	データ更新を指令します。
EM_SERVICE_TIMEREFRESH	19999	-	ベース時刻を再取得するための指令です。

7.3.3. サービスコンテナ動作設定

本サービスコンテナを RT-edge システムに導入する作業は次のようになります。

1) サービスコンテナの起動登録

EgBoot.xml の Service タグへサービスコンテナ起動を定義します。

```

...:
<RTedge...>
  ...:
  <Services>
    ...:
    <Service Name="EgTime" Path="EgTime.rta"/>
    ...:
  </Services>
</RTedge>

```

2) サービスコンテナの動作設定

EgTime.xml の Tag を編集します。

```

...:
<RTedge...>
  <Tags>
    <Tag Name="SERVICE.EgTime.AutoRun" Type="1" Value="1" Comment=" Startup auto"/>
    <Tag Name="SERVICE.EgTime.TimeSource" Type="3" Value="1" Comment=" Time source(0=RTC,1=KTick)"/>
  </Tags>
  <Services>
    <Service Name="EgTime">
      <TagRefs_IN>
        <TagRef Name="EDGESYSTEM.CurrentTime"/>
        <TagRef Name="EDGESYSTEM.CurrentTime_ms"/>
        <TagRef Name="EDGESYSTEM.CurrentTime_us"/>
        <TagRef Name="EDGESYSTEM.OperatingTime"/>
      </TagRefs_IN>
    </Service>
  </Services>
</RTedge>

```

3) サービスコンテナの実行

EgBoot.exe を実行することで起動します。

7.4. EgShDown サービスコンテナ

RT-edge システム上で動作しているサービスコンテナ/アプリケーションを一括終了する EgShDown サービスコンテナ (EgShDown.exe) を提供します。

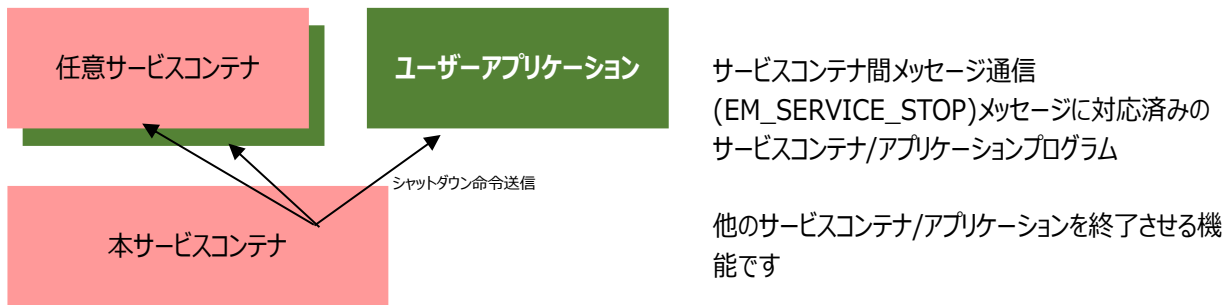


図 37 EgShDown サービスコンテナ概要

7.4.1. 機能

- 1) 他サービスコンテナ/アプリケーション一括終了

サービスコンテナ間メッセージ通信処理にて「EM_SERVICE_STOP」メッセージに対応しているサービスコンテナ/アプリケーションを終了させることができます。

- 2) 終了サービスコンテナ/アプリケーションの定義

EgShDown.exe と同フォルダに存在する EgShDown.xml に終了したいサービスコンテナ名を記載します。以下に例を記載します。黄色で塗りつぶしている箇所がサービス名になります。以下の例では、IIO(ECAT)サービスコンテナ、EgLog サービスコンテナ、EgTime サービスコンテナ、EgBoot サービスコンテナを終了させる定義になります。下記以外のサービスコンテナを終了させたい場合は、行をコピー＆ペーストし、黄色で塗りつぶしてあるサービスコンテナ名を変更して下さい。

```
...:
<ArrayOfAnyType ...>
  <anyType xsi:type="xsd:string">ECAT</anyType>
  <anyType xsi:type="xsd:string">EgLog</anyType>
  <anyType xsi:type="xsd:string">EgTime</anyType>
  <anyType xsi:type="xsd:string">EgBoot</anyType>
</ArrayOfAnyType>
```

7.4.2. サービスコンテナの実行

EgShDown.exe を実行することで EgShDown.xml に記載したサービスコンテナが終了します。

更新履歴

版	日付	更新説明
1	2021.6	● 初版作成
2	2021.8	● RT-edge Ver.3.3.0 向け変更内容反映
3	2021.10	● RT-edge Ver.3.4.0 向け変更内容反映
4	2022.5	● 5.5.6. RT-edge システム終了手順 にて、EgShDown への参照先が誤っていた点を修正 ● 仕様に記載の最大値を変更 ● 体裁・誤字脱字を修正
5	2023.5	● RT-edge Ver.3.5.2 向け変更内容反映 ・ タグトリガー機能の説明を追記 ・ タグのタイプに Segment 型、String 型を追加
6	2023.12	● 開発から運用までの流れの説明を追加 ● 環境構築の流れの説明を追加 ● 開発環境・実行環境の構築の修正
7	2023.12	● 利用可能なコンテナ情報の追加

REALTIME SERVICE for Windows

User's Manual

発行元：株式会社マイクロネット

TEL: +81(0)299-90-1733

FAX: +81(0)299-90-8557

- ・ 本書の著作権は、マイクロネットに帰属します。
- ・ 本書の内容、及び付属のソフトウェアの全部または一部を無断で転載することは禁止しております。
- ・ 本製品の内容については、将来予告なしに変更することがあります。
- ・ 本製品の内容について万一ご不審な点や記載もれなどお気づきの点がございましたら、お手数ですが、当社までご連絡ください。
- ・ Windows XP、Windows 7、Windows 8、Windows 10 等、Windows は、米国 Microsoft Corporation における登録商標です。
- ・ Visual Studio、Visual C++等は、米国、およびその他の国における Microsoft Corporation の登録商標です。
- ・ INtime は米国 TenAsys における登録商標です。
- ・ その他、記載されている会社名、製品名は、各社の商標又は登録商標です