C-021

# 組込み制御システム向け分散リアルタイム OS の コンフィギュレーション環境

A Configuration Environment of a Distributed Real-Time Operating System for Embedded Control Systems

齊藤　政典†　知場　貴洋†　伊丹　悠一†‡　横山　孝典†　兪　明連†

Masanori Saito　Takahiro Chiba　Yuichi Itami　Tkanori Yokoyama　Myungryum Yoo

## 1. はじめに

近年，自動車制御システムなどの組込み制御システムは複雑化，巨大化してきている．また，複数の組み込みコンピュータをネットワークでつなぎ，制御処理を分散して行う分散システムが増えている．そのため分散型の組み込みシステムに対応した実装環境が求められている．

自動車制御システム向けの基盤ソフトウェアの仕様として，OSEK/VDX 仕様がある．OSEK/VDX 仕様では，リアルタイムオペレーティングシステム (RTOS) の仕様である OSEK OS[1]，通信仕様である OSEK COM[2] をそれぞれ定義している．アプリケーションはシステムコールを発行することで OSEK OS にタスク起動などの処理を要求する．しかし他ノードのタスクに対して処理を行うためには，OSEK COM を用いて対象ノードに対して通信を行い，対象ノード上で OSEK OS のシステムコールを呼び出す必要がある．このため対象のタスクがどのノード上にあるかを意識したアプリケーション開発が求められる．

そこで我々は　，対象タスクがどのノード上にあるかを意識せずにアプリケーションを記述できる分散処理環境の実現を目的に位置透過性のあるシステムコールを有する分散 RTOS を開発している [3]．本論文では本分散 RTOS におけるコンフィギュレーション環境について述べる．

## 2. 分散 RTOS の構成と動作

我々が開発している分散 RTOS で，は OSEK OS 仕様に基づく TOPPERS/OSEK カーネル [4] に通信機能を追加し，他ノード上のタスクを直接システムコールの対象として引数で指定できるようにした．

他ノード上のタスクを対象にしたシステムコールを遠隔システムコールと呼ぶ．本分散 RTOS の構成を図 1 に示す．本分散 RTOS では OSEK OS の構成データとは別に遠隔システムコールを実現するためのデータを追加している．これを分散構成データと呼ぶ．分散構成データには，システム上でタスクがどのノードにあるのかを判別するための，全ノードで共通のグローバルタスク ID 等が含まれる．グローブルタスク ID は，OSEK OS では 1 バイトであったタスク ID を拡張して 2 バイトとし，上位 1 バイトでノードを，下位 1 バイトでノード内のタスクを識別できるようにした．

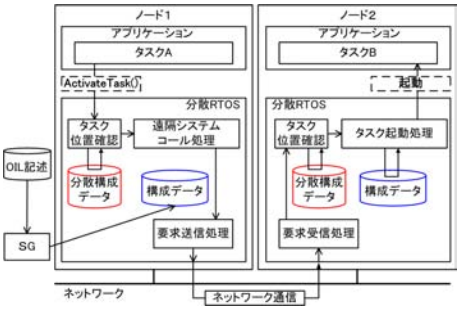


**図 1**　遠隔システムコールの処理の流れ

図 1 にはノード 1 上のタスク A が遠隔システムコールを発行してノード 2 上のタスク B を起動する場合の処理の流れも記載している．タスク A がタスク B の起動要求を出すと，分散 RTOS は対象タスクがどのノード上にあるのか分散構成データを参照して確認する．この例では，タスク B はノード 2 上にあるので，ノード 1 側の分散 RTOS は要求送信処理によってノード 2 側の分散 RTOS にタスク B を起動する処理を要求する．起動要求を受けたノード 2 側の分散 RTOS は，タスク B を起動する処理を行う．

## 3. OIL の拡張

OSEK OS 仕様では OIL と呼ばれる言語でタスクやリソース等のアプリケーションの設定情報を記述する．そして，システムジェネレータ (SG) と呼ばれるツールを用いて，OIL から OS が使用する構成データのソースコードを生成する．従来の OIL では 1 ノードの情報しか記載できなかった．本研究では複数のノードの設定情報を 1 つの OIL 記述ファイル内に記述できるように OIL の構文を拡張する．

図 2 は拡張 OIL の構文を BNF 記法で表記したものである．`<application definition>`が 1 ノード分のアプリケーションの設定情報を表す．従来の OIL では 1 つのファイルにひとつの`<application definition>`しか記述できなかったが，拡張 OIL では複数の`<application definition>`からなる`<application definition list>`を記述できるようにした．拡張 OIL 記述の例を図 3 に示す．図 3 はノードが 2 つそれぞれのノード内のタスク数が 2 つの場合の記述例である．ノードが 2 つあるためにノードを CPU1 と CPU2 で表し，その中にそれぞれのノードの設定 (タスクの宣言等) を記述している．

† 東京都市大学
‡ 現在，日立情報通信エンジニアリング (株)

**従来のOILの構文**

```
<file>::=
        <OIL_version>
        <implementation_definition>
        <application_definition>
                        …
<application_definition>::=
        “CPU”<name>”{“<object_definition_list>”}”<definition>”;”
```

**拡張OILの構文**

```
<file>::=
        <OIL_version>
        <implementation_definition>
        <application_definition_list>
                        …
<application_definition_list>::=
        <application_definition>
        |<application_definition_list><application_definition>
<application_definition>::=
        “CPU”<name>”{“<object_definition_list>”}”<definition>”;”
```

**図 2**　従来の OIL と拡張 OIL の構文

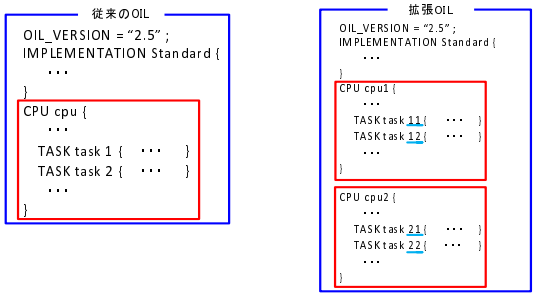


**図 3**　拡張 OIL による記述例

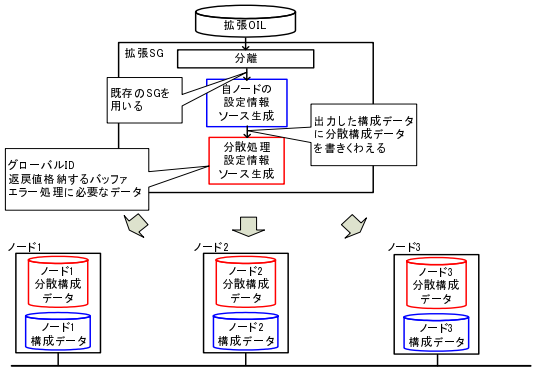


**図 4**　分散構成データを出力する流れ

## 4. 分散 RTOS 向け SG

我々はまた，拡張 OIL による記述ファイルから分散構成データを出力する SG を開発した．SG の処理の流れを図 4 に示す．まず拡張 OIL を従来の SG に対応した記述形式にするために分割する．それを従来の SG 記述に通しノードごとの構成データを出力する．出力した構成データを元に分散構成データのソースコードを生成する．分散構成データはノード ID，グローバルタスク ID，バッファ，エラー処理に必要なデータの 4 つのデータで構成される．以下それらの具体的生成方法について説明する．

**(1) ノード ID の生成**

分散 RTOS では位置透過性のある通信処理を行うために，ノードを識別するためのノード ID を生成する．拡張 OIL 記述においてノードを意味する CPU の宣言の順番と対応させてノード ID の値を割り振る．

**(2) グローバルタスクの ID 生成**

OSEK OS ではノードごとにタスク ID が SG により自動的に割り振られる．本分散 RTOS では，タスクがどのノードにあるかを識別するため，分散システム全体で一意に識別できるグローバルタスク ID をタスクに割り振る必要がある．もともと 1 バイトのタスク ID を 2 バイトに拡張し，従来の SG で割り振られたタスク ID の上位 1 バイトにノード ID を加えることで，グローバルタスク ID を生成する．

**(3) 返戻値を格納するバッファの生成**

OSEK OS 仕様では，システムコールを発行したタスクは正常に処理が完了したかを表す返戻値を受け取る．そのため分散 RTOS では，タスク毎に遠隔システムコールを発行した際に返戻値を保持するバッファが必要である．返戻値のサイズは 1 バイトなので，タスク数と同じ要素数の 1 バイトの配列を生成する．

**(4) エラー処理に必要なデータの生成**

OSEK OS 仕様では，存在しないタスクを起動しようとした場合にタスクを起動する前にエラーを返す．分散 RTOS では自ノードだけでなく他ノードのタスクが存在するかを確認する必要がある．そのために全ノードのタスク数を記憶した配列を用意する．タスク数より大きなタスク ID を指定した場合はエラーとする．

## 5. おわりに

本論文では，位置透過性のあるシステムコールを有する分散 RTOS におけるコンフィギュレーション環境について述べた．まず，OIL の構文を拡張することによって分散システム全体のアプリケーションの設定情報を記述できるようにした．次に拡張 OIL から分散構成データを出力する拡張 SG を開発した．本コンフィギュレーション環境を用いることで，本分散 RTOS を用いた組込み制御システムを効率よく開発できる．

## 参考文献

[1] OSEK/VDX，OSEK/VDX Operating System Version 2.2.3，February 17th 2005.
[2] OSEK/VDX，OSEK/VDX Communication Version 3.0.3，July 20 2004.
[3] Chiba，T.，Itami，Y.，Yoo M.and Yokoyama，T.，A Distributed Real-Time Operating System with Location-Transparent System Calls for Task Management and Inter-Task Synchronization，Proccedings of the 8th IEE International Conference On Embedded Software aad Systems PP.1133-1138，2011
[4] http://www.toppers.jp/osek-os.html