1J-05

論理時間を用いた分散型組込みシステムのための コンフィギュレーションデータ生成ツール

田口 峻平† 横山 孝典† 兪 明連†

東京都市大学‡

1. はじめに

近年，組込み制御システムは分散化が進んでいる．例えば，自動車分野において，車載ネットワークを用いた自動車内の通信のみではなく，無線ネットワークを用いた分散型組込み制御システムが増えている．分散型組込み制御システムの基本的な構成はセンサからの入力値を用いて，アクチュエータに処理結果を出力する一連のサイクルをネットワークに接続された複数の組込みコンピュータで行うというものである．そして，分散型組込み制御システムではセンサ入力からアクチュエータ出力までのデッドラインを守るリアルタイム性が要求される．従って，リアルタイム性を考慮した分散処理環境及び開発環境が必要となっている．

通信時間が変動するネットワークにおいてもリアルタイム性を考慮した分散処理システム開発環境として PTIDES[1][2]がある．全てのノードにおいて時刻同期を行うことでデッドラインを守る処理を実現しているが，専用の開発環境を使用する必要があるため既存のアプリケーションを利用できない．

我々は，無線ネットワークを用いたシステムのための分散処理環境として，論理時間を用いた分散処理ミドルウェア[3]を提案している．詳細な時間制約を与えることなく時刻同期をできるだけ少数のノードのみとし，物理時間とは別に仮想的な論理時間を用いて通信時間が変動する無線ネットワーク等に対応している．ミドルウェアでリアルタイム性を考慮した分散処理環境を実現することで，アプリケーションとは独立にリアルタイム性を保証できる．

本論文では，本分散処理ミドルウェア向けのコンフィギュレーションデータを自動生成するツールを提案する．

A Configuration Data Generator for Distributed Embedded Control Systems Based on Logical Time

†Shunpei Taguchi, Myungryun Yoo, Takanori Yokoyama

‡Tokyo City University

2. 論理時間を用いた分散処理環境

2.1 分散処理環境の動作

本分散処理環境を用いたシステムの動作例を図 1 に示す．この例では 4 つのノード上にそれぞれ入力タスク，算出タスク A，算出タスク B，出力タスクが分散配置されている．タスクの周期は全て 10 である．

本環境では，入出力タスクは物理時間に同期した時間駆動アーキテクチャに基づいて起動するため，ジッタの少ない処理が行われる．一方，算出タスクはメッセージ受信イベントで起動し，仮想的な時間である論理時間に基づいて管理を行い，通信時間の変動を許容している．

本分散処理環境では，タスク間の起動時刻の差を遅延時間，論理時間上の遅延時間を論理遅延時間と呼ぶ．また，算出タスクにおける起動時刻を論理起動時刻と呼ぶ．そして，各タスクのジョブがメッセージを送信するときに，そのジョブの論理起動時刻をタイムスタンプとして付加することで，ノード間で論理時間に基づいたタスク管理を可能としている．

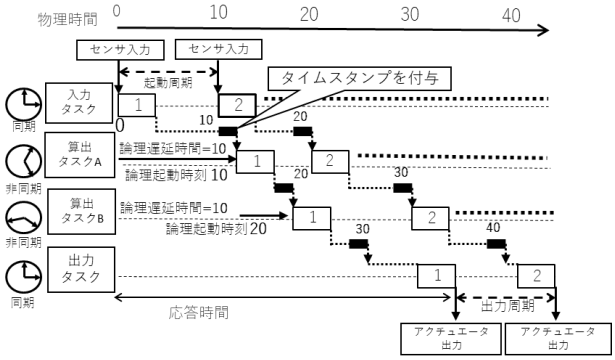

図 1：論理時間を用いた分散処理の動作例

2.2 分散処理ミドルウェアの構成

論理時間を用いた分散処理環境のソフトウェア構成を図 2 に示す．本ミドルウェアはノード毎に配置されたアプリケーションの処理に用いる情報を含むコンフィギュレーションデータに記述されたタスクやメッセージ通信に関するコンフィギュレーション情報を参照して動作する．コンフィギュレーション情報にはタスクやメッセージ通信に関する情報が含まれる．タスクに

関する情報は，タスク種別，タスク起動周期，タスクの論理遅延時間が含まれ，メッセージに関する情報は，メッセージの遅延時間が含まれる．

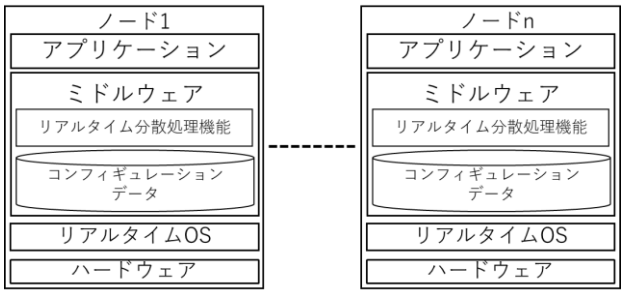

図 2: ソフトウェア構成

2.3 コンフィギュレーションデータの生成

本研究では，論理時間を用いた分散処理ミドルウェアが参照するコンフィギュレーション情報を，C言語によるプログラム形式による記述ではなく，宣言的に記述する方法を提案し宣言的な記述方式で記述されたコンフィギュレーション情報からプログラム形式のコンフィギュレーションデータを自動生成するツールを開発する．

本ツールを使用した場合のコンフィギュレーションデータ生成の流れを図 3 に示す．開発者がアプリケーションに応じたコンフィギュレーション情報をコンフィギュレーションデータ生成ツールに入力することでコンフィギュレーションデータを自動生成する．

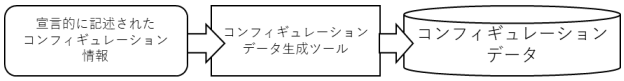

図 3: コンフィギュレーションデータ生成の流れ

3. コンフィギュレーションの記述と作成

3.1 コンフィギュレーション情報記述法

提案する記述法は，RTOS のコンフィギュレーション情報の記述法である OIL[4]を拡張し，コンフィギュレーション情報を属性と値の組み合わせによる宣言のみで記述可能する．また，開発者が入力する情報をできるだけ少数とすることで，プログラム形式の記述に比べ記述量を減らし開発工数の削減を狙う．提案する記述法によるタスクのコンフィギュレーション情報の記述例を図 4 に示す．この例では，タスク種別(入力，算出，出力)，タスク起動周期，タスクの論理遅延時間の三つを記述している．

3.2 コンフィギュレーションデータ生成ツール

本ツールは，構文解析部とコード生成部によって構成される．構文解析部は，パーサジェネレータの ANTKR4[5]を用いて作成し，入力されるコンフィギュレーション情報の構文解析が行われて，タスクやメッセージ通信に関する情報の抽出が行われる．コード生成部は，Java 及び AspectJ[6]により開発した．コンフィギュレーション情報の構文解析が行われて抽出された，タスクやメッセージ通信に関する情報やそれらを元に新たに算出される情報からコンフィギュレーションデータを生成する．

```
task = CAL_TASK;
task_cyc = 500;
DESTIME = 500;
```

図 4: 提案する記述法による記述例

図 5 に実際に出力されるコンフィギュレーションデータの例を示す．配列の 1 番目にノード内 1 つ目のタスクやメッセージの情報が記述され，それらがタスクやメッセージ数分存在する．

```
const TaskAtrType task_atr[TASK_NUM] = {INPUT_TASK,CAL_TASK,.....};
const TspTimeType task_cyc[TASK_NUM] = {500,500,500,.....};
TspTimeType logic_start[TASK_NUM] = {0,0,0,.....};
TspTimeType next_logic_start[TASK_NUM] = {0,0,0,.....};
TspTimeType timestamp[TASK_NUM] = {0,0,0,.....};
const TspTimeType  des_time[TASK_NUM] = {500,500,500,.....};
```

図 5: コンフィギュレーションデータ

4. おわりに

本論文では，論理時間を用いた分散処理ミドルウェアにおける，コンフィギュレーションデータを自動生成するツールについて述べた．今後は，さらなる開発工数削減のため，アプリケーションのモデルからコンフィギュレーション情報を抽出することでコンフィギュレーションデータを自動生成できるツールを開発する予定である．

謝辞

本研究は JSPS 科研費 18K11225 の助成を受けたものである．

参考文献

[1] J.C. Eidson, E.A.Lee S.Matic c, S.A.Seshia, J.Zou,Distributed Real-Time Software for Cyber-Physical Systems, Proceedings of the IEEE, Vol. 100, No.1, pp.45-59, (2012).
[2] J.Zou, S.Matic, E.A.Lee, PtidyOS: A Lightweight Microkernel for Ptides Real-Time Systems, IEEE 18th Real Time and Embedded Technology and Applications Symposium, pp.209-218,(2012).
[3] Ayumu ichimura, Takanori Yokoyama and Myungryun Yoo :A Time-triggered Distributed Computing Envirnment for Cyber-Phycical Systems Based on Physical Time and Logical Time, Proceedings of TENCON 2018-2018 IEEE Region 10 Conference, pp.1510-1515(2018).
[4] OSEK/VDX, System Generation, OIL: OSEK Implementation Language, Version 2.5, (2004).
[5] ANTLR ; http://www.antlr.org/
[6] The AspectJ Project - Eclipse ; https://www.eclipse.org/aspectj/