研究成果報告

実行可能 UML に基づく組込みシステムネットワーク設計に関する研究

Research on Embedded Sysem Network Design Based on Executable UML

星野 達也*、松本 倫子*、吉田 紀彦*
Tatsuya Hoshino*, Noriko Matsumoto*, Norihiko Yoshida*

Embedded systems have recently been working on networks. Such embedded system networks involve communication in various layers, thus their design is more difficult than of single embedded systems. This paper presents how communication exploration can be done in a design process of embedded system networks using an example of event-triggered and time-triggered communication. A design process begins from abstract specification without assuming any communication category, then explores the categories in a stepwise manner, and is followed by physical implementation synthesis. All the models involved in this process are represented in Executable UML. This encourages stepwise decision making, component and framework reuse, and early stage verification.

Keywords: Embedded system network, Executable UML, System design

1. はじめに

組込みシステムは，携帯電話やディジタル家電から自動車などまで,様々な機器に制御用に内蔵され,産業界で大きな重要性を持つ．ハードウェアとソフトウェアが密接に連携して動作することから，両者の設計を統合的に並行して進める必要があり，システムの複雑化と大規模化に伴なって設計の困難さが増大している [1]．一方で，近年の商品サイクルの短縮化に応えなければならないため，設計プロセスの効率化が急務となっている．そこで，ハードウェア･ソフトウェアの効率的な統合設計を目指して「システムレベル設計」と呼ばれる方法論が提唱されて,実用化され始めている [2]．しかし，第 1 に，SpecC [3, 4]，SystemC [5]，SystemVerilog [6] など記述言語(一種のプログラミング言語)に依存しており,真の汎用性と利便性に欠ける．第 2 に，すでに自動車などは，数十の組込みシステムをネットワーク接続した組込みシステムネットワーク（分散組込みシステムともいう)を搭載するようになってきており,そのような形態にも対応する必要がある [7].

第 1 の課題に向けては,「実行可能 UML」という抽象度の高い記述形式を用いる方法論が提唱されて,有効性が実証されつつある [8, 9, 10]．しかし，第 2 の課題を包含するまでには至っていない．本研究では,「車載ネットワーク」と総称される自動車内の組込みシステムネットワークを特に取り上げ，記述言語依存レベルでの我々自身の研究成果 [11，12] を実行可能 UML で抽象化して再構成し，システムレベル設計を高度化することを目指した. これにより,記述言語に依存しない真の汎用性を，今後一層の重要性を増すことが確実な組込みシステムネットワークの設計においても実現することを図った [13].

* 埼玉大学 大学院理工学研究科
Graduate School of Science and engineering, Saitama University, 255 Shimo-Okubo, Sakura-ku, Saitama, Saitama 338-8570, Japan
（原稿受付日：平成 21 年 5 月 29 日）

2. アスペクト指向実行可能 UML

本研究では、アスペクト指向技術 [14] を適用した実行可能 UML を用いて、組込みシステムネットワーク設計における抽象通信モデルの記述、モデルの段階的具体化の定式化を行った [15]。

実行可能 UML（Executable UML）とは UML 2.0 のプロファイル（拡張）の一つであり、UML 2.0 にアクション言語とモデルコンパイラを追加したモデリング言語である。モデルは静的側面を表すクラス図、動的側面を表すステートチャート図を中心に構成され、その振舞いはアクション言語で記述される。アクション言語は厳密なセマンティクスに則っており、モデルコンパイラによって実行可能、したがって検証可能になっている。

アスペクト指向とは、ソフトウェアの中に散在している関心事（横断的関心事）を他の関心事から分離してモジュール化する技術である。関心事とはソフトウェアの設計やプログラミングの際に自然に抽出される処理やデータを指す。そして、横断的関心事とは、複数の関心事に横断的に関係し、単独では取り扱えない関心事を指す。横断的関心事の例としては、ロギング、キャッシング、アクセス制御、並行処理における同期などが挙げられる。横断的関心事は、オブジェクト指向ではモジュール化することが困難であった。アスペクト指向では、このような横断的関心事をアスペクトとして記述することによりモジュール化する。そして、アスペクトを「織り込む」ことにより、横断的関心事を統合したソフトウェアを実装する。これによって、ソフトウェアの保守性や再利用性を向上させる。アスペクトはジョインポイントモデルによって表現され、アドバイスとポイントカットから構成される。アドバイスとは織り込むべき機能や処理、ポイントカットとはアドバイスを織り込むべきプログラム中の位置の定義である。具体的な位置はジョインポイントと呼ばれ、例えばメソッド呼出し、フィールド参照などである。

3. 通信モデル

ネットワークの通信モデルは、大きくイベントトリガ型とタイムトリガ型の 2 つに分類される。イベントトリガ型モデルでは、データの送信タイミングは送信モジュールに委ねられており、送信モジュールは任意に送信できる。しかし、複数のモジュールが同時に送信する可能性があるため、データの衝突を回避するための機構（アービタ）が必要となる。一方、タイムトリガ型モデルでは、データの送信タイミングはあらかじめ事前に定められており、モジュールは通信の管理を行うスケジューラに従って通信を行う。データの衝突が起きないため、リアルタイム性を重視するシステムに適している。

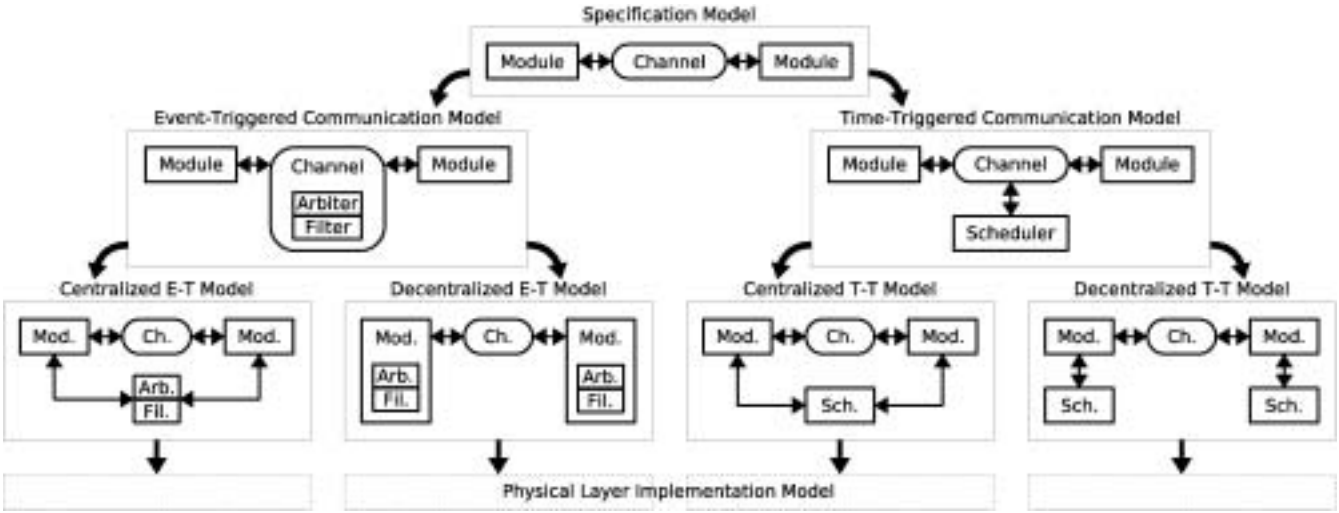

Fig. 1 Stepwise exploration of communication models.

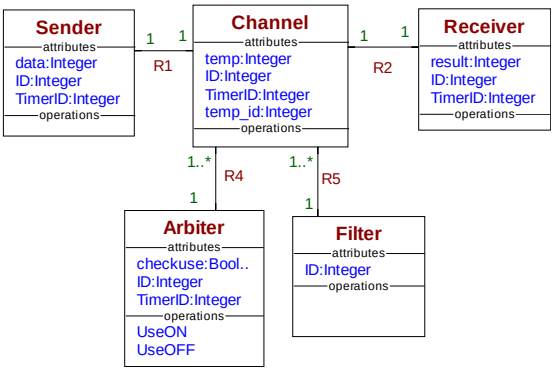

Fig. 2 Event-triggered model in executable UML.

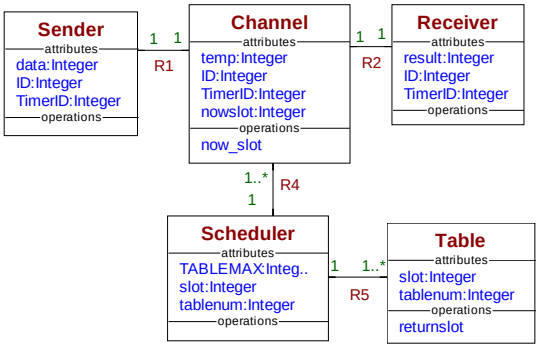

Fig. 3 Time-triggered model in executable UML.

イベントトリガ型モデルとタイムトリガ型モデルはそれぞれ、さらに集中型と分散型に分類することができる。集中型モデルでは、モジュール間の通信をネットワーク中の唯一のアービタもしくはスケジューラで管理する。分散型モデルでは、各モジュールがそれぞれアービタもしくはスケジューラを持ち、通信の管理を行う。集中型モデルは構造が単純で実装が容易である。一方、分散型モデルは設計が複雑になるが、高速な通信が可能となる。例えば、組込みシステムネットワークの代表例である車載ネットワークでは、CAN（Controller Area Network）はイベントトリガ分散型モデル、LIN（Local Interconnect Network）はタイムトリガ集中型モデル、FlexRay はタイムトリガ分散型モデルになる。

4. 通信モデルの段階的具体化

通信モデルの設計実装は、最も抽象度の高い通信モデルである仕様モデルから始める。仕様モデルではデータの送信モジュールと受信モジュールは抽象的なチャネル介して通信する。この段階では通信プロトコルなどは考慮しない。次に、仕様モデルをイベントトリガ型モデルかタイムトリガ型モデルのどちらか一方に具体化する。この具体化により、チャネルは、イベントトリガ型モデルではアービタとフィルタの機能を持ち、タイムトリガ型モデルではスケジューラの機能を持つようになる。引き続き、この 2 つの通信モデルを、集中型モデルか分散型モデルのどちらかに具体化する。集中型モデルではアービタとフィルタ、もしくはスケジューラはチャネルから独立させ、1 つのモジュールとして全モジュールと接続し通信を管理する。分散型モデルでは、通信を行う全モジュールがアービタとフィルタ，もしくはスケジューラの機能を持つ。これを Fig. 1 に示す。

5. 段階的具体化の定式化

本研究では、具体化の各段階をアスペクトの系列で記述することで、具体化手順の定式化を行う。そして、アスペクト系列を実行可能 UML モデルに自動適用することで、モデルの具体化を自動化する。例えば、仕様モデルからイベントトリガ型モデルへの変換は 48 個のアスペクト、仕様モデルからタイムトリガ型モデルへの変換は 32 個のアスペクトで記述できる。結果として得られるイベントトリガ型の実行可能 UML モデルを Fig. 2 に、タイムトリガ型のモデルを Fig. 3 に示す。これらを実際にシミュレーション実行して、動作確認、検証を行うことができる。

6. おわりに

このように設計を段階的に具体化することにより、

設計者はアプリケーションに適したモデルの設計・選択が容易となる。また、段階的にモデルの検証を行うことで、ミスが下流設計に入り込むことを防止できる、さらに, 段階ごとの部品化が可能になり、設計資産の再利用性の向上にもつながる。

参考文献

[1] F. Balarin, et al., *Hardware-Software Co-Design of Embedded Systems*, Springer (1997)

[2] K. Keutzer, et al., “System Level Design: Orthogonalization of Concerns and Platform-Based Design”, IEEE Trans. CAD19:12 (2000) 1523-1543

[3] D. D. Gajski, et al., *SpecC: Specification Language and Methodology*, Kluwer (2000)

[4] SpecC Web Site:
http://www.cecs.uci.edu/~specc/

[5] SystemC Web Site:
http://www.systemc.org/

[6] SystemVerilog Web Site:
http://www.systemverilog.org/

[7] B. Kleinjohann, et al., *Design and Analysis of Distributed Embedded Systems*, Springer (2002)

[8] S. J. Mellor, et al., "MDA Distilled: Principles of Model-Driven Architecture", Addison-Wesley (2004)

[9] S. J. Mellor, et al., *Executable UML: A Foundation for Model-Driven Architecture*, Addison-Wesley (2002)

[10] C. Raistrick, et al., *Model Driven Architecture with Executable UML*, Cambridge Univ. Press (2004)

[11] K. Kobayashi, et al., “Exploration of Communication Models in Design of Distributed Embedded Systems”, IEEJ Trans. EEE, 2:3 (2007) 402-404

[12] T. Kinoshima, et al., “Communication Model Exploration for Distributed Embedded Systems and System Level Interpretations”, LNCS 4809, Springer (2007) 355-364

[13] T. Hoshino, et al., “Communication Model Exploration in Aspect-Oriented Executable UML”, Proc. Int. Conf. on App. Comp. (2009) to appear

[14] 千葉, アスペクト指向入門, 技術評論社 (2005)

[15] A. Teruya, et al, “Embedded System Design Based on Aspect-Oriented Executable UML”, Proc. 8th Int. Conf. on App. Comp. Sys. (2008) 247-252