1L-5

モデル変換とコード生成機能を有する組み込み制御ソフトウェア開発支援ツール

神山 達哉　兪 明連　横山 孝典

東京都市大学

１．はじめに

一般に組み込み制御ソフトウェア開発は制御設計，ソフトウェア設計，実装の３工程から成る．自動車制御等多くの分野の制御設計工程では，MATLAB/Simulink[1]を用いて制御モデル(Simulinkモデル)を作成している．ソフトウェア設計工程では，制御モデルをもとに，実装を考慮したソフトウェアモデル（UMLモデル）を作成する．実装工程では，ソフトウェアモデルに基づいて制御プログラムを作成する．

しかしSimulinkモデルとUMLモデルでは，その狙いや対象，モジュール化のしかた，表記法などが異なり，それらの間で明確な対応付けはなされていない．Simulinkモデルを用いた制御設計からUMLを用いたソフトウェア設計にスムーズに移行するため，SimulinkモデルからUMLモデルへの変換方法の確立が求められている．

その１手法として，Simulinkモデルの要素をそれぞれクラスとしてUMLモデルに変換する手法が提案されている[2]．この手法では，Simulinkモデルの構造がそのままUMLモデルに反映され，部品化再利用に適したクラス構成にはならない．

そこで本研究では，Simulink モデルから部品化再利用に適した構造を持つ UML モデルに変換する方法を提案するとともに，その変換方法に対応した変換ツールを開発する．本手法では，制御ロジック上重要なデータをクラスに対応させることで，再利用性を向上する．また，変換後の UML モデルに対応したソースコードの生成手法を提案し，自動化するツールを開発する．

２．Simulinkモデルの階層化

2.1 階層化の狙いと方法

ソフトウェア設計工程ではまず，制御設計工程で作成した Simulink モデルを，制御ロジック上重要なデータのみが上位階層に現れるように階層化する．制御ロジック上重要なデータを算出するサブシステムブロックで上位階層を構成することで，変換後の UML モデルを再利用しやすいクラス構成とすることができる．

A Simulink-UML Translator and a Code Generator for Embedded Control Software Development
Tatsuya Kamiyama, Myungryun Yoo and Takanori Yokoyama, Tokyo City University

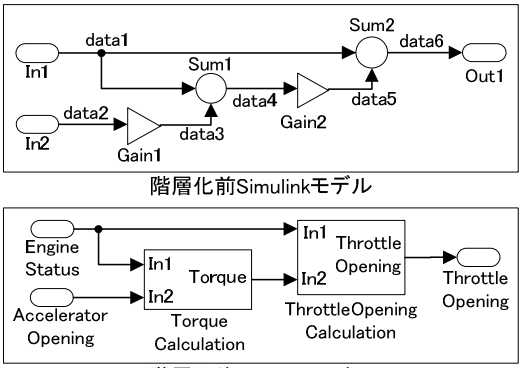

図１　Simulink モデルの階層化例

SrcBlock	DstBlock	LineName	DataName	Select
In1	Sum1,Sum2	data1	EngineStatus	✓
In2	Gain1	data2	AcceleratorOpening	✓
Gain1	Sum1	data3		
Sum1	Gain2	data4	Torque	✓
Gain2	Sum2	data5		
Sum2	Out1	data6	ThrottleOpening	✓

図２　階層化支援ツールの表示例

階層化の例を図１に示す．data1(エンジン状態)，data2(アクセル開度)，data4(トルク)，data6(スロットル開度)を重要なデータとし，上位階層に現れるように階層化している．data4, data6 を算出する具体的な処理はサブシステムブロックの下位階層で行われる．

2.2 階層化支援ツール

効率よい階層化を支援するツールを開発した．本ツールは Simulink モデルを記述した mdl ファイルを読み込み，階層化後の Simulink モデルを記述した mdl ファイルを生成し，書き出す．

図１の階層化を行うときの本ツールの画面を図２に示す．本ツールは Simulink モデルを読み込み，そのモデルに含まれるラインをリスト化し表示する．ユーザーは，リストの中から重要なデータに対応するラインの select ボックスにチェックを入れることで，階層化後の Simulink モデルを得ることができる．

３．SimulinkモデルからUMLモデルへの変換

3.1 変換方法

階層化したSimulinkモデルの上位階層に現れるデータに対応させて，クラスを定義する．クラスには属性とメソッドを記述する．メソッド

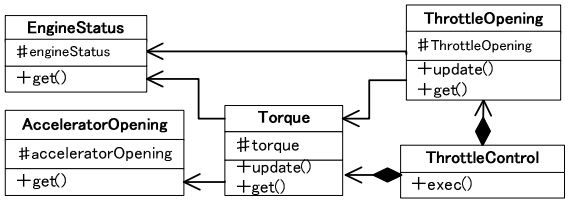

図３　変換後のクラス図例

は，値を渡すためのgetメソッド，Simulinkモデル内に値を算出する処理が記述されたデータのクラスには値を更新するupdateメソッドを設ける．また，Simulinkモデルのデータの流れに対応させて，クラス間の関連を定義する．そしてSimulinkモデル全体に対応したControlクラスを設ける．Controlクラスには，各クラスのupdateメソッドを呼び出すexecメソッドを記述する．updateメソッドを持つクラスとControlクラス間の関連も定義する．

図１の階層化後Simulinkモデルを変換したクラス図を図３に示す．トルクはエンジン状態とアクセル開度，スロットル開度はエンジン状態とトルク，Throttle Control(スロットル制御)はトルクとスロットル開度を参照する．トルクとスロットル開度のupdateメソッドでは，参照先の値を用いてそれらの値を算出し属性値に記憶する．スロットル制御のexecメソッドはトルクとスロットル開度のupdateメソッドを呼び出す．これによりSimulinkモデルに対応したクラス構成を実現できる．

3.2 モデル変換ツール

提案した変換処理を自動化するツールを開発した．開発した変換ツールは，Simulinkモデルのmdlファイルを入力し，変換後のUMLモデルのXMI(XML Metadata Interchange)ファイルを出力する．ツールに図１の階層化後Simulinkモデルを入力することで，図３のクラス図を得る．

なお，Simulinkモデルを変換して得たUMLモデルは制御機能のみを表現したものであり，ソフトウェア設計工程では，実装に必要なクラスの追加や，他の機能を実現するUMLモデルとの統合化を行い，実装を考慮したUMLモデルを作成する

４．コード生成

4.1 生成方法

実装工程では，Embedded Coderを用いてSimulinkモデルから生成した制御ソースコードと，UMLモデリングツールを用いて作成したUMLモデルから生成したクラスソースコードを合成し，最終成果物であるソースコードを生成する．Simulinkモデルから生成されたデータを算出するためのコードを，それぞれ対応するクラスの

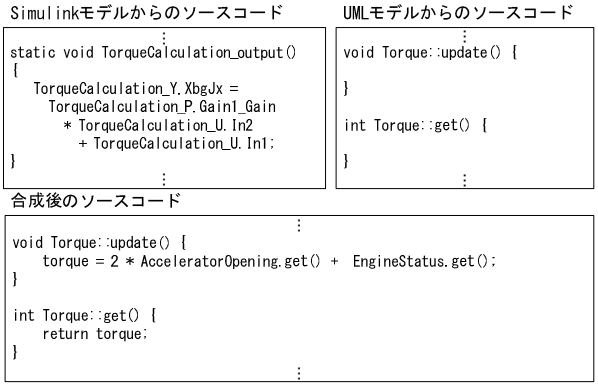

図４　ソースコードの合成例

updateメソッド内に組み込む．組み込む際には，クラス図に対応するように書き換える．

図１の階層化後Simulinkモデルのトルク算出部分と，図３のクラス図のトルククラスのコード合成例を図４に示す．図４では，トルクを算出するコードのエンジン状態とアクセル開度の値を呼び出している部分はgetメソッドを用いるように書き換えている．また，算出結果を自クラスの属性値に代入するように書き換える．

以上の合成処理を全てのクラスで行うことで，変換後のクラス図に対応したソースコードを作成できる

4.2 合成ツール

コード合成処理を自動化するための合成ツールを開発した．合成ツールは，階層化後のSimulinkモデル，Simulinkモデルから生成されたソースコード，UMLから生成されたソースコードを読み込むことで，各クラスの合成後のソースコードを生成し書き出す．

５．おわりに

本研究では組み込み制御ソフトウェアを対象に，Simulinkモデルから再利用性の高いUMLモデルへの変換方法と，モデル変換に対応したコード生成方法を提案した．またそれらの工程を自動化または支援するツールを開発することでソフトウェアの開発効率を向上させた．

今後の課題は，Simulinkのみでなく Stateflow[1]等を用いた制御モデルにも対応できる開発環境を実現することである

参考文献

[1] The Math Works, Inc., http://www.mathworks.com/products/simulink/.
[2] C. Reichmann et al., Model Level Coupling of Heterogeneous Embedded Systems, Proc. 2nd RTAS Workshop on Model-Driven Embedded Systems, 2004.