A-022

# 制御ソフトウェアの仕様整合性検証手法の検討と評価
# Verification Method of Consistency between Specifications of Control Systems

大貫 智洋†
Tomohiro Onuki

## 1. はじめに

### 1.1 背景

近年，制御ソフトウェアの大規模化が進んでいる．例えば自動車に搭載される制御ソフトウェアの規模は 2000 年当時に 100 万行程度であったものが，2009 年時点で 500 万行～1000 万行にまで増加している[1]．このような大規模化の背景には，制御系システムの電子制御化が進み，より広範な機能をソフトウェアによって実現するようになったことがある．例えば自動車では，エンジン制御やトランスミッション制御，パワーステアリング制御，ドアロック制御など，様々な制御を ECU（Electronic Control Unit：電子制御ユニット）で行っており，1 台あたりに搭載される ECU の数は年々増加する傾向にある[2]．

このように大規模化した制御ソフトウェアを効率的に開発する手法として，モデルベース開発の導入が進んでいる．モデルベース開発はシステムの制御仕様をモデルで記述することを特徴とする手法であり，設計レベルのモデルによるシミュレーション検証や，実装レベルのモデルによる実装コードの自動生成が可能となる．ソフトウェア品質の観点では，実装コードを機械的に生成することが可能であるため，従来の人手による作業に比べて実装に誤りが混入する危険を抑えることが可能となる．その一方で，要求仕様や制御仕様の記述はモデルベース開発においても人手で行っており，自然言語の曖昧さや記述漏れを原因とする誤りが混入しやすい．このため，要求仕様の定める性質を制御仕様が正しく実現しているか検証することが重要な課題となっている．

### 1.2 本論文の目的

モデルベース開発では，制御モデルの記法に状態遷移モデルを用いる．従って，その検証にはモデル検査の適用が効果的であると考える．モデル検査とは，システムの振舞いを記述した状態遷移モデルに対して遷移パスを網羅的に探索し，システムの性質を表す検査項目に違反する遷移パスを検出する手法である．

本論文では，複数の構成要素が連動して動作するシステムを設計する上で，要求仕様に記述された通信仕様が制御仕様に正しく記述されているか検証する手法を検討し，適用例を用いて提案手法の評価を行う．なお，検証対象の通信仕様はアプリケーションレベルのものとする．

†三菱電機(株)　情報技術総合研究所
Mitsubishi Electric Corp. Information Technology R&D Center

## 2. 従来技術の課題と解決方針

モデル検査により仕様の整合性を検証する方法について関連研究を調査し，課題を分析した[3]．その結果，**図 1**に示すモデル検査適用方法のうち，検査項目の作成と検査モデルの作成に課題があることが分かった．課題分析の結果を以下に要約する．

(1) 検査項目の作成に関する課題

要求仕様から検査項目を作成するためには，①要求仕様から検証すべき性質を抽出し，②抽出した性質を時相論理式で表す必要がある．調査の結果，②に関しては時相論理式のパターン化により効率化できることが分かったが，①に関しては検討課題として残った．

(2) 検査モデルの作成に関する課題

制御仕様から検査モデルを作成するためには，③設計レベルの制御モデルを通信仕様の側面に絞って抽象化し，④抽象化したモデルをモデル検査ツールで検証可能な検査モデルに変換する必要がある．調査の結果，④に関しては UML 状態マシン図から SPIN や UPPAAL 等の検査モデルを自動的に生成する手法が適用できることが分かったが，③に関しては検討課題として残った．

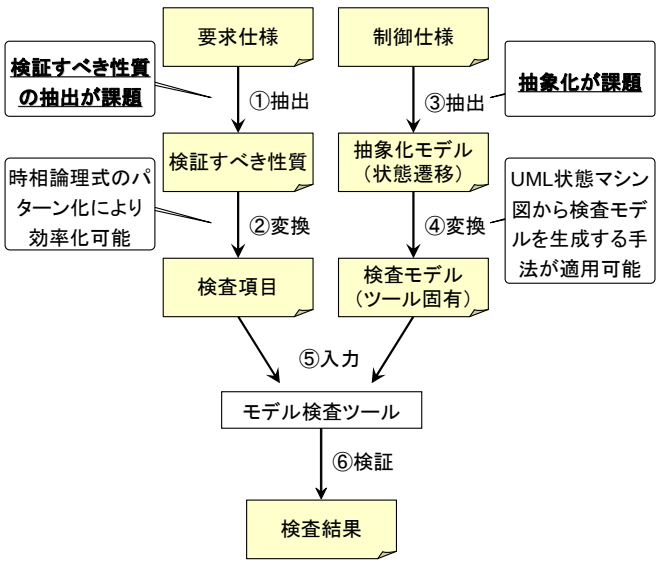


**図1　モデル検査の適用方法と課題**

## 3. 制御ソフトウェアの仕様整合性検証手法

### 3.1 概要

本手法は，上記の課題のうち(1)検査項目の作成に関する課題を解決し，**図 1**の手順に従って要求仕様と制御仕様の整合性を検証するものである．要素間の通信仕様に関して検証すべき性質を予め定め，UML シーケンス図で記述した要求仕様から検証すべき性質を機械的に抽出し，検査項目に変換することで，(1)の課題を解決する．制御仕様の抽象化（**図 1**の③）は今回の検討の範囲外であり，以下ではすでに抽象化された 制御仕様を入力とする．

### 3.2 本手法のアルゴリズム

本手法のアルゴリズムを**図 1**の手順に従って説明する．以下，3.2.1では①，3.2.2では②，3.2.3では④，3.2.4では⑤⑥の手順について説明する．

#### 3.2.1 要求仕様から検証すべき性質を抽出する方法

本手法では，要素間の通信に関して検証可能な性質を**表 1**のとおりに定める．

**表1　検証可能な性質**

| 項目 | 性質 |
|---|---|
| (1) 要求・応答の成立 | すべての要求に対して，必ず応答があること． |
| (2) 要求・応答の順序性 | 要求・応答が正しい順序で行われること． |
| (3) エラー応答の成立 | 一定回数の要求に対して応答がない場合，エラー応答が送信されること． |

UML シーケンス図によって記述した要求仕様を入力とし，各項目の検証に必要な情報をシーケンス図から機械的に抽出する．以下では，各項目の検証に必要な情報の抽出方法を説明する．

(1) 要求・応答の成立

シーケンス図から要求と応答の対を抽出し，記憶する．要求と応答の関連付けは，ステレオタイプで行う．要求・応答を含むシーケンス図の例を**図 2**に，シーケンス図から要求と応答を抽出した例を**表 2**に示す．

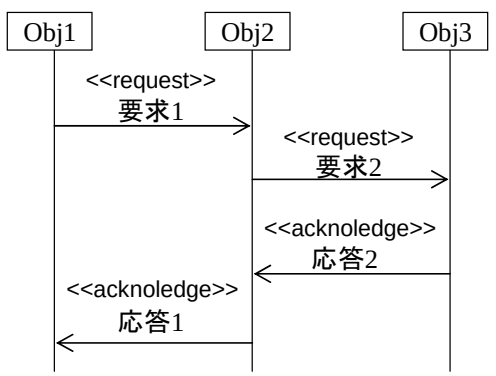


**図2　要求・応答を含むシーケンス図**

**表2　要求・応答の抽出**

| 番号 | 要求 | 応答 |
|---|---|---|
| 1 | 要求１ | 応答１ |
| 2 | 要求２ | 応答２ |

(2) 要求・応答の順序性

シーケンス図から要求・応答の前後関係を抽出し，記憶する．**図 2**に示すシーケンス図から要求・応答の前後関係を抽出した例を**表 3**に示す．

**表3　要求・応答の前後関係抽出**

| 番号 | 前 | 後 |
|---|---|---|
| 1 | 要求１ | 要求２ |
| 2 | 要求２ | 応答２ |
| 3 | 応答２ | 応答１ |

(3) エラー応答の成立

エラー応答が発生する状況として，次の 2 つを想定する．

- 要求のタイムアウトを一定回数繰り返した場合
- エラー応答を受信し，他の構成要素にエラー応答を送信する場合

シーケンス図に記述されたエラー応答に対し，エラー応答の直前の振舞いが上記の 2 つのいずれであるか識別し，エラー応答の条件として抽出・記憶する．要求のタイムアウトの判別は UML Testing Profile**[4]**で行い，エラー応答の判別はステレオタイプで行う．

エラー応答を含むシーケンス図の例を**図 3**に，シーケンス図からエラー応答の条件とエラー応答を抽出した例を**表 4**に示す．

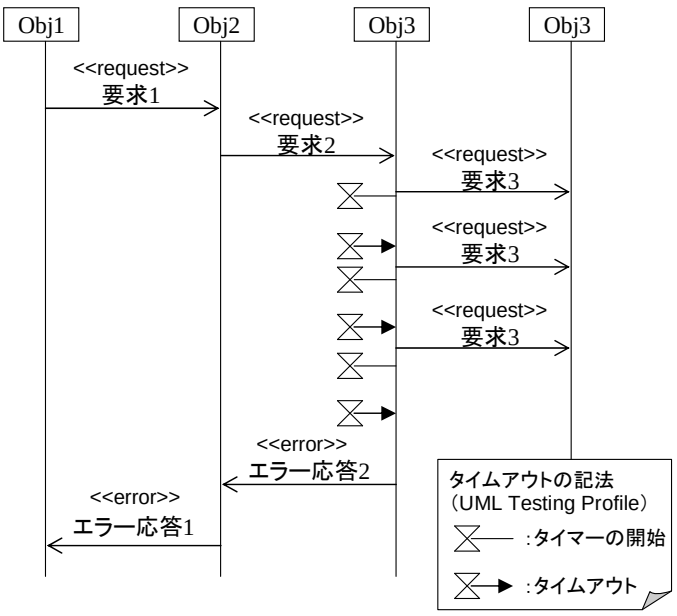


**図3　エラー応答を含むシーケンス図**

**表4　エラー応答の条件とエラー応答の抽出**

| 番号 | 条件 | エラー応答 |
|---|---|---|
| 1 | 要求３，<br>送信回数：３ | エラー応答２ |
| 2 | エラー応答２ | エラー応答１ |

### 3.2.2 検証すべき性質を検査項目に変換する方法

3.2.1で抽出した各性質に対して，時相論理式のテンプレート**[5]**を定めることで検査項目を機械的に生成する．各項目の変換方法を**表 5**にまとめる．なお，時相論理に用いる記号の意味は次のとおりである**[6]**.

- [] p　　：常に命題 p が成り立つ
- <> p　　：将来のある時点で命題 p が成り立つ

**表5　検証すべき性質から検査項目への変換**

| 項目 | 性質 |
|---|---|
| (1)要求・応答の成立 | [](要求) -> <>(応答))<br>(**表 2**の例の場合)<br>[](要求１ -> <> 応答１)<br>[](要求２ -> <> 応答２) |
| (2)要求・応答の順序性 | [](要求/応答) -> <>(要求/応答))<br>(**表 3**の例の場合)<br>[](要求１ -> <> 要求２)<br>[](要求２ -> <> 応答２)<br>[](応答２ -> <> 応答１) |
| (3)エラー応答の成立 | ・[](要求回数) = = 一定数&&<br>(応答なし) -> <> (エラー応答) )<br>・[](エラー応答) -><>(エラー応答))<br>(**表 4**の例の場合)<br>・[](要求２回数 = = 3 && 応答なし<br>-> <> エラー応答２ )<br>・[](エラー応答２-><>エラー応答 1) |

### 3.2.3 抽象化モデルを検査モデルに変換する方法

システムを構成する各要素の状態遷移モデルを基に，一つの検査モデルを記述する．例えばモデル検査ツール SPIN では独自の仕様記述言語 Promela で検査モデルを記述する．Promela ではシステム構成要素の状態遷移を個別のプロセスとして記述することが可能であるため，状態遷移モデルからプロセス記述への変換規則を定めることで，検査モデルを機械的に生成することができる．Promela によって記述した検査モデルの構成を**図 4**に示す.

### 3.2.4 検証実行と仕様整合性確認の方法

3.2.2，3.2.3で作成した検査項目と検査モデルをモデル検査ツールに入力し，検証を実行する.

モデル検査ツールが検査項目の違反を報告しなかった場合は，本手法が定める性質（**表 1**）において，要求仕様と制御仕様の整合性が保たれていると判断できる.

モデル検査ツールが検査項目の違反を報告した場合は，報告結果をもとに，検査項目の違反が要求仕様の誤りによるものか，制御仕様の誤りによるものかを分析・修正する．検査項目の違反を検出しなくなるまで仕様の修正と検証を繰り返す.

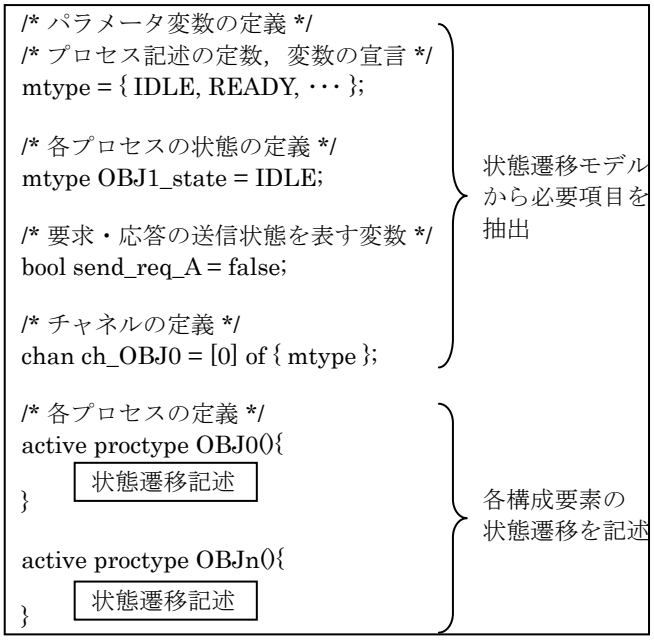


**図4　Promela で記述した検査モデルの構成**

## 4. 適用評価

本章では，簡単な例題に対して本手法を適用し，効果を評価する．以下の適用評価では，表 1 に示す性質のうち(1)要求・応答の成立に関して検証を行うものとする.

### 4.1 例題の仕様

#### 4.1.1 要求仕様

OBJ0，OBJ1，OBJ2，OBJ3 の 4 つの要素で構成される．OBJ0 は OBJ1，OBJ2 にそれぞれ通知メッセージ（notify）を送信する．OBJ1 は通知メッセージを受け取ると，OBJ2 に要求（req_A）を送信する．OBJ2 は通知メッセージを受け取ると，OBJ3 に要求（req_B）を送信する.

要求仕様を表すシーケンス図を**図 5**に示す.

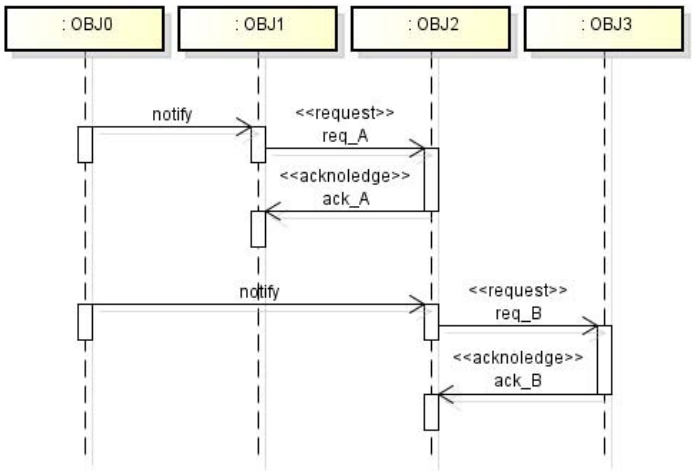


**図5　要求仕様を表すシーケンス図**

#### 4.1.2 制御仕様

OBJ0 は状態遷移を持たない．OBJ1，OBJ2，OBJ3 の初期状態は IDLE であり，他の構成要素からメッセージを受信すると計算状態に遷移し，計算が完了すると再び IDLE に遷移する．また，ある状態で受け付けられないメッセージを受信した場合，これを破棄する．OBJ1，OBJ2，OBJ3 の状態遷移モデルを**図 6**に示す．

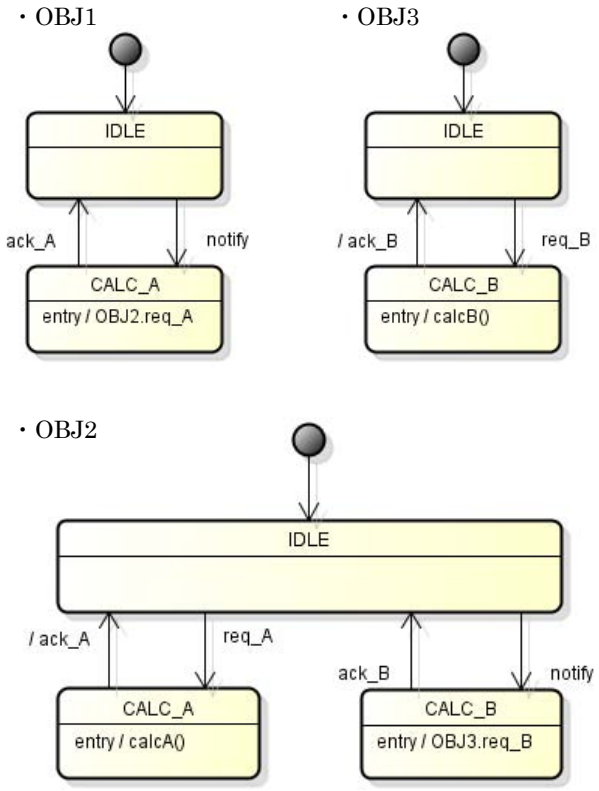


**図6　制御仕様を表す状態遷移モデル**

### 4.2 検証結果

表 1 に示す性質のうち(1)要求・応答の成立に関して検証を行った．検査項目は，**図 5**に示すシーケンス図から以下の 2 つが生成された．

[]( req_A -> <> ack_A )
[]( req_B -> <> ack_B )

検証の結果，以下の検査項目違反を検出した．

・違反した検査項目
[]( req_A -> <> ack_A )
・違反が発生した動作シナリオ
OBJ1 が notify を受信，CALC_A 状態に遷移
→OBJ2 が notify を受信，CALC_B 状態に遷移
→OBJ2 が OBJ3 に req_B 送信，OBJ3 が受信し CALC_B 状態に遷移
→OBJ1 が OBJ2 に req_A 送信，OBJ2 は受信するが，CALC_B 状態のため req_A を破棄
→OBJ3 が OBJ2 に ack_B 送信，OBJ2 が受信し IDLE 状態に遷移
→終了

検査項目違反の原因は，OBJ2 の状態が CALC_B の時に OBJ1 から要求 req_A を受信し，これが受け付けられなかった点にある．

OBJ1 が，応答 ack_A を受け付けるまで req_A を繰り返し送信するように変更すると，この検査項目違反は検出されなくなる．

### 4.3 評価

例題への適用結果で示すように，本手法を用いることで，要求が正常に受け付けられなかった場合の動作シナリオを導出することができる．検証結果を基にエラー発生時の処理を検討・追加することで，要求仕様を満たす制御仕様を記述することが可能となる．

また，計算時間については例題の規模で数秒程度であると見込んでいる．今回の検討では検査項目および検査モデルの作成機能が開発途上であり正確な計測は出来ていないが，これまでに開発した同等の変換機能を持つ検証ツール**[7]**の実績から上記のように見込んでいる．なお，SPIN による探索は状態数 1454，遷移数 4153 に対して探索時間は 0.188[秒]であった．

### 4.4 今後の課題

今回の検討では検査可能な性質の枠組みとして通信に関する基本的な 3 つの性質をとりあげた．今後，通信仕様の検証に必要と考えられる項目を洗い出し，要求仕様からの抽出方法，検査項目への変換方法を検討する．

また，今回の検討では制御仕様の抽象化を範囲外としたが，本手法を実開発に適用する上では，検証する性質に合わせて制御仕様を抽象化する方法を明確にする必要がある．

## 5. おわりに

本稿では，要求仕様に記述された要素間の通信仕様が制御仕様に正しく記述されているか検証する手法を検討・評価した．本手法の特徴は，シーケンス図で記述した要求仕様から要求・応答に関する検査項目を自動的に生成することである．例題を用いた評価では，要求が正常に受け付けられない場合の動作シナリオの導出に効果を確認した．

今後は，通信仕様の検証に必要な項目を洗い出すことと，制御仕様を抽象化する方法を体系化することが課題となる．

### 参考文献

[1] 経済産業省 商務情報政策局 情報処理振興課: 高度情報化社会を見据えた情報システム・ソフトウェアの信頼性向上に向けて, 2009.
[2] 佐藤弘明: トヨタ,日産,ホンダは ECU を何個使っているのか, 日経エレクトロニクス 2010 年 12 月 27 日号, 2010, pp.175-186.
[3] 大貫智洋: 制御ソフトウェアの仕様整合性検証技術の調査と分析, 情報処理学会 研究報告, Vol.2012-SE-175 No.10, 2012.
[4] Object Management Group: UML Testing Profile Version 1.0, 2005.
[5] Matthew B. Dwyer, George S. Avrunin, James C. Corbett: Property Specification Patterns for Finite-State Verification, Proceedings of the Second Workshop on Formal Methods in Software Practice (2008)
[6] 中島震: SPIN モデル検査, 近代科学社, 2008.
[7] 大貫智洋, 上野浩一郎, 磯田誠: 状態遷移を持つオブジェクト間通信のモデル検査技術, 情報科学技術フォーラム講演論文集 9(1), 2010, pp.307-308.