3L-08

組み込みシステムにおける設計制約を検証するための UML モデル検査手法

小林　雅弥†　　松浦　佐江子‡

芝浦工業大学　システム理工学部　電子情報システム学科†‡

1. はじめに

複数機器で構成される組み込みシステムは、各機器が協調して目的を達成する。要求分析の段階で、その目的を達成するためのシステムの振舞いを UML アクティビティ図で表したワークフローにより表現することができる。各機器の振舞いはアクションのフローにより、機器間の協調はシグナル送受信で表現することで、各機器の処理フローとその協調を定義する。しかし、その協調が実際にうまく動作し、目的を達成できるかは UML モデル図だけでは分からない。それは各機器の通信やハードウェアに関わる設計制約の情報が不足している可能性があるからだ。情報が不足した状態で開発を進めると下流工程で大きな手戻りが発生する事になる。

そこで、開発早期の段階でワークフローが要求だけでなく設計制約も満たすことができるかを調べるために、モデル検査を利用する。モデル検査はシステムの振舞いを有限状態遷移図（Finite State Machine, FSM）に、その性質を時相論理に基づく論理式（検査式）に表すことで性質を満たすかを網羅的に検査する事ができる技術である。組み込みシステムのような並行動作するシステムを表現、検査する事ができる。性質を満たさない場合はその時の FSM のパス（反例）を表示する。検査可能な性質は「全てのパスで必ず P にならない」（安全性）「いつかは Q を満たすパスが存在する」（活性）などである。今回はモデル検査ツール UPPAAL[1]を使用した。これは並行動作に加え、時間制約を表現、検査する事ができる。UPPAAL における FSM をシステムモデルといい、検査式と合わせて UPPAAL モデルと呼ぶ。

本稿では、設計制約について問題がある既存の組み込みシステムのワークフローを UPPAAL モデルに変換して、モデル検査を実行し、その結果から得られたワークフローの不足した情報を導く。この情報をワークフローにフィードバックする事でワークフローを洗練し、手戻りを防止する事を目指す。

A Verification Method of UML Requirement Models for Design Constraints of Embedded System

†Masaya KOBAYASHI ‡Saeko MATSUURA

†‡Department of Electronic Information System, Collage of System Engineering and Science, Shibaura Institute of Technology

2. 調査実験

2.1. 概要

芝浦工業大学で実施しているシステム開発実習で作成された「荷物自動搬送システム」のワークフローを手動でモデル検査ツール UPPAAL のモデルに変換し、検査と修正を行い、ワークフローの問題を明らかにする。このシステムは実際に下流工程において手戻りが発生しており、ワークフローに何らかの問題がある事が分かっている。

「荷物自動搬送システム」とは LEGO MINDSTOMS NXT(以降 NXT)と PC で構成されている組み込みシステムである。全体の動作は、受付所（PC）で荷物を受け取り、収集担当ロボット（NXT）がそれを中継所（NXT）に運ぶ。そして、配達担当ロボット（NXT）が中継所で荷物を受け取り、受取人宅（PC）で荷物を受け渡す。この一連の流れの中で、受付所と中継所は本部（PC）に荷物の配達に関わる受付時間や発送開始報告などのデータを送信する。

各機器は Bluetooth またはソケット通信によってデータの授受を行っている。Bluetooth は NXT から PC へ通信確立要求を出す事はできないという制約がある。

2.2. 前提条件

UML であるワークフローと UPPAAL モデルは概念が異なる。前者は処理を基準に後者は状態を基準に作成される。そこでワークフローの 1 つの処理ステップを UPPAAL モデルの 1 つの状態に対応するよう変換した。

ワークフローには機器を表すパーティション、1 つの処理を表すアクションノード、条件分岐を表すディジションマージノード、通信を表すシグナル送受信ノード、システムが扱うデータを表すオブジェクトノードなどの要素がある。

UPPAAL モデルでは、1 つの非同期に動作する状態遷移図をテンプレート、そのインスタンスであるプロセス、1 つの状態をロケーション、ロケーション間の遷移をエッジとして表す。エッジには遷移可能な条件を表すガード、テンプレート間の同期をとるチャネル、変数の値を変化させる更新と呼ばれる項目がある。さらに、状態遷移図が性質を満たすかを検査する検査式をロケーションや変数等を使用して定義できる。

これらを踏まえて、変換に関する前提条件を示す。

- パーティション内に独立したフローがある場合は非同期で処理すると見なし、別のテンプレートと

Copyright ©2015 Information Processing Society of Japan.
All Rights Reserved.

して定義する。その際、同一機器には同一の機器 ID を振って関連付ける。

- アクションノード、ディジションマージノードをロケーションに変える。
- シグナル送受信ノードの対をチャネルに変える。

2.3. 検査項目：処理の完了

当初は「すべてのプロセスがある状態から遷移できない」というデッドロックをしない事を検査していた。しかし、ワークフローには基本フローと例外フローが存在し、それぞれが別の終了ノードを持っていたり、ループしないプロセスが存在したりする。したがって、処理が途中で停止しない事を調べるには、単純にデッドロックしない事を調べるだけでは不十分であった。検査上、状態爆発の原因になるため、ループしているフローにはループ回数を制限した。また、検査式については終了ノードに到達するかを検査するようにした。

UPPAAL では「全てのパスで必ずデッドロックしないか、または終了ノードに到達するか」を検査した、結果は成功した。これによって、ワークフローは途中で停止しない事を確認できた。

2.4. 検査項目：通信確立

NXT の Bluetooth の制約で NXT から PC へ通信確立を要求できない。これを満たしているか検査する。

UPPAAL モデルにおいて同一の機器と通信のやり取りをしている部分を特定、その最初のロケーションの前に通信確立を表す処理、最後に通信切断する処理を挿入した。

通信確立要求を出す側はチャネルによって通信相手と同期をとる。同時に通信確立したかを表すフラグを立てる。もし、通信確立が失敗したら通信確立が失敗したフラグが立ち、機器 ID も変数に記録される。この通信確立要求が終わった後のエッジのガードで通信確立が成功したかを判定、失敗していたらここで遷移が停止する。

ワークフローには通信確立要求を明示する記述がないため、どちらが要求を出しているか分からない。そのため、先に送信をしている機器が通信確立要求を出していると仮定した。

UPPAAL では「全てのパスで必ず通信確立が失敗したフラグが立っていない」を検査した結果は満たされなかった。

収集担当ロボット―受付所、中継所―本部、配達担当ロボット―受取人宅の通信において NXT 側のガードの条件が偽となり、遷移が停止している。これは仮定していた通信確立要求を出す方向が間違っているために起きた。このようなミスを減らすために表 1 のような情報をワークフローに追加する事を提案する。

表 1　通信確立についての制約記述

符号	種別	主な指定箇所	意味
device type	タグ	パーティション	機器の種類
connection capacity	タグ	パーティション	最大同時接続数
request sending	ステレオタイプ	アクションノード	通信確立要求を送る
request waiting	ステレオタイプ	アクションノード	通信確立要求を待つ
use bluetooth	ステレオタイプ	アクションノード	Bluetoothを使う
use socket	ステレオタイプ	アクションノード	ソケットを使う

2.5. 検査項目：情報の伝達

荷物自動搬送システムでは、荷物の配達状況が確認できることが要求されている。そこで通信によってあるデータが目的の機器に到達するかを検査した。

まずワークフローでオブジェクトノードが生成される箇所を特定し、データ ID を振る。以降、オブジェクトノードは名前と型が等しい場合、同一と見なした。そして、機器ごとに用意したデータの到達を表すフラグを立てる関数を用いて、データのやり取りを UPPAAL モデル上で表現した。

UPPAAL では「受取人宅に荷物が到達するパスが存在するか」「本部に依頼情報、荷物番号、受付時間、発送開始、中継所到着、配達開始、受け取り時間、配達完了が到達するパスが存在するか」を検査し、結果は失敗した。

原因は本部が複数の通信を受け付けている事と送信データと受信データの整合性が取れていない事ある。シグナル送受信ノードの入場動作や本部の更新項目のノートに書かれた説明を頼りにデータ ID を振り直し、本部の受信部分を分割し、1 対 1 通信に直した。再度検査すると検査は成功した。

実験から、送信するオブジェクトノードをシグナル送信ノードの前などに登場させる事、本部のシグナル受信ノードを送信ノードの数に合わせる事を提案する。

3. 考察

設計制約はワークフローに記述する基準を定義したという意味で有意義だろう。例えば、ワークフローの作成時には基準に従って書く事で、レビュー時は基準に従っているかを確かめる事でワークフローの問題を早期に発見する事ができる。すなわち、下流工程での手戻りを防止する事ができる。

制約記述の適応範囲について考察する。今回提案した制約記述は通信を双方向にできないという性質のような汎用的な制約を満たすために使う。したがって、他のシステムに対しても制約記述を適応する事が可能である。

4. まとめ

本稿では、複数機器で構成される組み込みシステムのワークフローについてモデル検査ツールを用いて、ワークフローに不足した設計制約を満たすための情報を明らかにし、ワークフローにそれらの情報を記述する方法を提案した。

今後は他のシステムのワークフローについても同様の調査実験を行い、制約記述の体系を構築したい。

5. 参考文献

[1] UPPAAL, http://www.uppaal.org

Copyright ©2015 Information Processing Society of Japan.
All Rights Reserved.