ソフトウェアの実行前検査の研究

2005MT067 宮地 昭裕
指導教員　蜂巣 吉成

1　はじめに

我々の研究室は，組込みソフトウェアのアスペクト指向ソフトウェアアーキテクチャスタイル (以後，E-AoSAS++) を提案している．E-AoSAS++ は，組込みソフトウェアを並行に動作する状態遷移機械 (以後，CSTM) の集合と規定する．E-AoSAS++ は各 CSTM ごとに設計を行い，複数の CSTM を合成してソフトウェアを実現する．開発者は各 CSTM の動作は把握できるが，合成後の動作まで把握して設計するのは困難である．よって，合成後に開発者が意図した動作を行わない可能性があるので，設計したソフトウェアが要求仕様を満たさない場合が発生し得る．このような問題が実装後に発見されると，設計段階へ手戻り作業が発生する．そこで，設計段階で実行前検査を行い問題の早期発見をする必要がある．

本研究の目的は，要求仕様と設計したソフトウェアとの整合性を確認する検査手法を提案することである．検査手法として，軌跡を用いたモデル検査手法を提案する．軌跡とは，モデル化したソフトウェアの処理の順序を表したものである．この軌跡を用いて，要求仕様の軌跡が設計したソフトウェアの軌跡に含まれているかの検査を行うことにより，整合性を確認する手法になると考えた．設計段階で実行前検査を行い問題を早期に発見することで，設計段階への手戻りを軽減させることができる．

2　モデル検査

モデル検査とは，対象となるモデルの状態を網羅的に探索することでソフトウェアに問題があるかを検査する．本研究では CSP[1] の理論を基にしたモデル検査ツール FDR[2] を使用する．

CSP は並行プログラムの相互作用のパターンを記述する言語の 1 つである．並行システムをモデル化したプロセスがあり，プロセスが発生させたイベントの有限記号列が軌跡となる．例として，プロセス P が a→b とイベントを発生した時，軌跡は，⟨⟩ で囲まれて表現され，プロセス P がとり得る全ての軌跡の集合を traces(P) とすると，traces(P) =$\{\langle\rangle,\langle a\rangle,\langle a,b\rangle\}$ となる．

FDR で検査する項目の一つに Traces Refinement がある．Traces Refinement は 2 つのプロセスの軌跡の集合おいて，片方の軌跡の集合に，他方の軌跡の集合が含まれているかを検査する機能である．この機能を用いて，要求仕様の軌跡が設計したソフトウェアの軌跡に含まれているかの検査を行う．

3　E-AoSAS++ の概要

E-AoSAS++[3] は，組込みソフトウェアを並行に動作する状態遷移機械の集合と規定する．CSTM が互いイベントを送りあい，連動して動作することで組込みシステムを実現する．

本検査手法において必要な E-AoSAS++ の図式は，CSTM のインスタンスの関係を示すオブジェクト図．CSTM の動的な振る舞いを表現する状態遷移図．アクションを表すシーケンス図．要求仕様を表現するソフトウェア全体の挙動を表すシーケンス図である．

4　軌跡を用いた実行前検査方法

本章ではまず，どういうものを要求仕様として抽象化し，どう表現するかを明示し，軌跡を用いた検査を行いたいので CSP に変換する規則を提案する．次に，設計した E-AoSAS++ のソフトウェアを CSP に変換する規則を提案する．変換した CSP から TracesRefinement を用いて，要求仕様の軌跡が設計したソフトウェアの軌跡に含まれているかの検査を明示する．

4.1　要求仕様の表現方法と CSP への変換規則

要求仕様を表現する際に，ソフトウェアにおける全ての処理を表現することは困難である．そこで，ソフトウェアに対し確認したい一部の動作を要求仕様として表現する．確認したい一部の動作は，E-AoSAS++ の CSTM 間のイベント通知のみとし，これが発生する正しい順とする．この動作を，E-AoSAS++ のソフトウェア全体の挙動を表すシーケンス図に記述する．

また，複雑なソフトウェアの動作を 1 つのシーケンス図で記述することは困難である．そこで，シーケンス図の表記方法の 1 つである相互作用オカレンスを用いて，複数のシーケンス図に任意に分割する．シーケンス図名を CSP 記述のプロセス名とし，シーケンス図のメッセージが発生する順に CSP のイベントとして記述する．この変換規則を図 1 に示す．この CSP との対応関係をもたせることにより，要求仕様として表現されたソフトウェアの動作を CSP で表現する．また，相互作用オカレンスを用いて分割されたシーケンス図では，参照先のシーケンス図を CSP へ変換し，参照元に挿入することで結合させる．図 1 の例では，プロセス spec が要求仕様のプロセスとなり，要求仕様のプロセスの軌跡の集合は traces(spec) となる．

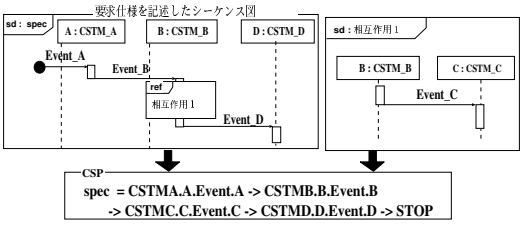

図 1　シーケンス図から CSP へ変換する例

4.2 E-AoSAS++ の図式表現から CSP への変換規則

E-AoSAS++ に基づいて設計したソフトウェアの振る舞いとは，各 CSTM の連動した振る舞いであるので，モデル化するには，CSTM の動的な振る舞いを表す図から検査言語に変換する必要がある．よって，E-AoSAS++ の図式表現で検査に用いる図式は，オブジェクト図と，各 CSTM における状態遷移図，他の CSTM に通知するイベントを記述するアクションを表すシーケンス図である．次に，CSTM の振る舞いを CSP で表現する対応関係を述べる．まず，CSTM の状態遷移図における各状態から，CSP のプロセスを 1 つ作成する．この変換方法を図 2 に示す．また，状態遷移図に記述してあるアクションは，アクションを表すシーケンス図とオブジェクト図から，CSP でアクションを表すプロセスとして作成する．この変換方法を図 3 に示す．この変換規則により，図 2 と図 3 のプロセスから CSTM の振る舞いを CSP で表現する．

しかし，要求仕様の変換規則では，CSTM 間のイベント通知のみを CSP の軌跡として記述する．E-AoSAS++ に基づいたソフトウェアの設計においても，CSTM 間のイベント通知のみを軌跡とする必要があるので，他のイベントは隠蔽しなければならない．図 2 と図 3 のように各 CSTM の振る舞いを CSP で表現し，これらを合成してソフトウェア全体の動作を表現する．表現したソフトウェアのプロセスを Q とおき，プロセス Q の軌跡の集合は traces(Q) となる．

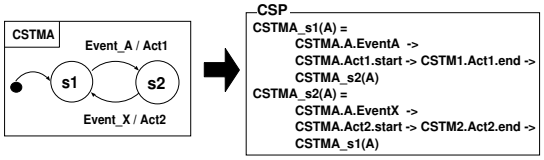

図 2　状態遷移図を表現した CSP 記述の例

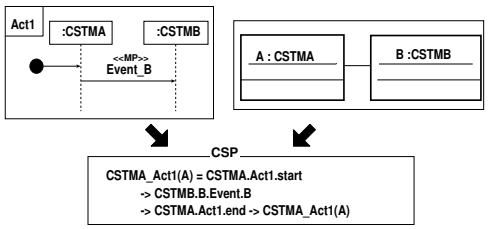

図 3　アクションを表現した CSP 記述の例

4.3 FDR を用いた検査

FDR の Traces Refinement を用いて，4.1 章で変換した要求仕様としてのプロセスの軌跡の集合 traces(spec) が，4.2 章で変換した設計したソフトウェアとしてのプロセスの軌跡の集合 traces(Q) に含まれているかを検証する．2 つの集合の関係は，次のような包含関係になる．

$\text{traces(spec)} \subseteq \text{traces(Q)}$

これより，要求仕様との整合性を確認する検査手法となると考えた．この関係が成立すると設計したソフトウェアは，表現した要求仕様を満たすことを保証する．

5 考察

5.1 要求仕様の表現について

本研究で提案した要求仕様の表現について考察する．例として，自動販売機のソフトウェアを挙げる．要求仕様を表現する際に，自動販売機のソフトウェアにおける全ての動作を記述することは困難である．そこで，ソフトウェアに対して確認したい一部の動作を，本検査手法で必要な要求仕様として表現する．例えば，お金を投入したら商品が出てくる．このように確認したい一部の動作に着目することで，要求仕様として表現できると考えた．この要求仕様を基にソフトウェアの動作として，お金が入る，ランプが点灯する，ボタンが押される，商品を出すという処理の順番をシーケンス図に記述する．ソフトウェアの全ての動作を記述することは難しいが，確認したい動作ごとに要求仕様とすれば表現できると考えた．

5.2 事例を用いて検査手法を検証

本研究で，要求仕様と設計されたソフトウェアとの整合性を確認する検査手法を提案した．この手法が有効であるかをスイッチライトの例を用いて検証する．スイッチが押されたらライトが点灯するスイッチライトのソフトウェアがあり，このソフトウェアは問題のないものとする．確認したい一部の動作を，スイッチを押したらライトがつくまでとし，これを要求仕様として表現する．ソフトウェアに問題はないので，要求仕様と整合性はある．本検査手法を用いたところ，正しい結果が得られた．よって，本研究で提案した検査手法は有効であると考えた．

6 おわりに

本研究で実行前検査の一つとして，要求仕様と E-AoSAS++ に基づいて設計したソフトウェアとの整合性を確認する検査手法を提案した．提案した検査手法により，設計したソフトウェアの動作が表現した要求仕様を満たすかを発見することができる．

今後の課題として，モデル検査言語の習得や，図式から検査言語へ変換することは手間がかかるので，検査言語へ自動的に変換できる自動生成ツールを作成することが挙げられる．また，シーケンス図で表現できる要求仕様を整理し検討してくことが必要である．

参考文献

[1] C.A.R. Hoare, *Communicating Sequential Processes*, Prentice-Hall, Apr. 1985.

[2] Formal Systems (Europe) Ltd, *FDR2 User Manual,* 2005; www.fsel.com/documentation/fdr2/html/fdr2manual.html.

[3] M. Noro, A. Sawada, Y. Hachisu, and M. Banno, “ E-AoSAS++ and its Software Development Environment,” *Proceedings of the 14th Asia-Pacific Software Engineering Conference(APSEC 07)*, Dec. 2007, pp. 206-213.