状態遷移表検証ツールの適用

7N－3

柴山　武彦*、古山　真司*、藤波　武起**
*日立中部ソフトウェア（株）、**（株）日立製作所　ソフトウェア開発本部

１．はじめに

通信系ソフトウェアを中心に、状態遷移表を用いた開発が盛んである。状態遷移表は、プログラミング言語に比べ、対象の動きを状態と事象により整理出来る点で優れている。特に、網羅性の点では、表形式で記述することから漏れが少ない。一方、ソフトウェア製品の開発では、状態遷移表の規模が大きくなったり、複数の状態遷移表が相互に関連する場合が多い。このような時、状態遷移表は、シーケンス図等に比べると、視覚性に乏しく、設計者が意図した通りの状態遷移表を作成できたかチェックするのが難しかった。

これに対し、我々は状態遷移表設計支援ツールを開発し、製品開発に使用されている状態遷移表に適用した。本報告では、この適用について述べる。

２．状態遷移表設計支援ツール

状態遷移表は、網羅性に優れている。これは、設計対象のすべての状態と事象を表形式で記述し、状態と事象により一意に決まるボックスに対して設計対象の動作を記述することによる。

しかしながら、表形式であるため、設計対象の動きを視覚的に理解することが困難である。この点では、シーケンス図の方が優れている。また、状態遷移表に記述された状態と事象の組み合わせは、本来遷移すべきでない組み合わせへ遷移したり、デッドロックに陥る可能性がある。特に複数の状態遷移表を用いて開発する通信システムでは、これらの不良が発生する可能性が高い。しかし、これらの不良は到達可能性解析により解析可能である[ref.1]。

状態遷移表設計支援ツール（図１）は、これらの点を支援する。ツールの持つ機能は以下の通り。

（１）シーケンス図生成機能

複数の状態遷移表からシーケンス図を生成する機能。本機能により、状態遷移表の設計レビュー等を支援する。

（２）検証機能

到達可能性解析により、複数の状態遷移表を対象とした検証を行う機能。実際には起こり得ない状態と事象の組み合わせに遷移してしまう不正遷移や、デッドロック等の不良を含む問題点を検出できる。

３．ツールの適用

本支援ツールを開発後、製品として流通している通信系ソフトウェアを対象とし、状態遷移表の検証作業を実施した。対象としたプロジェクトの状態遷移表は以下の通り。

表１　適用対象としたプロジェクト

ﾌﾟﾛｼﾞｪｸﾄ名		proj-A	proj-B	proj-C	proj-D
ﾏﾄﾘｸｽ 数		2	4	6	11
ﾏﾄﾘｸｽ規模	状態数	8～10	7～11	6～36	2～45
	事象数	9～11	9～22	6～32	6～36

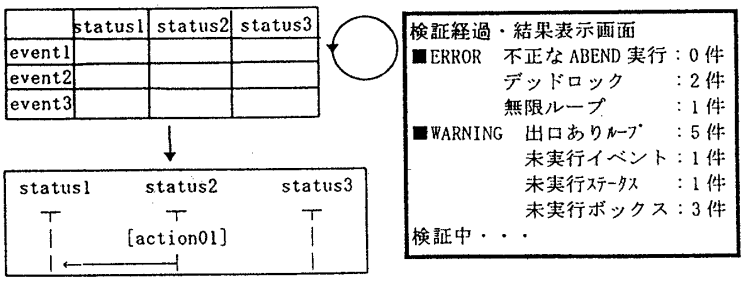

図１　状態遷移表設計支援ツール

Applying verifying tool to status matrix
Takehiko Shibayama, Shinji Koyama, Takeki Fujinami
Hitachi-TS; Software Development Center,Hitachi, Ltd.
5030 Totsuka, Totsuka,Yokohama, Kanagawa 245,Japan

4．適用結果

表１に示したプロジェクトで作成した状態遷移表を、状態遷移表設計支援ツールの検証機能に適用した。適用結果を表２に示す。

表２　適用結果

検出項目	proj-A	proj-B	proj-C	proj-D
不正遷移	０件	**２件**	０件	４件
ﾃﾞｯﾄﾞﾛｯｸ	０件	**３件**	０件	０件
無限ﾙｰﾌﾟ	０件	０件	０件	０件
ﾙｰﾌﾟ(出口あり)	１３件	０件	０件	０件
未実行ｲﾍﾞﾝﾄ	０件	０件	０件	０件
未実行ｽﾃｰﾀｽ	０件	０件	**８４件**	５５件
未実行ﾎﾞｯｸｽ	１１件	０件	０件	０件

表２に示した適用結果を解析したところ、ツールで検出した問題点は、以下の２つの問題により発生していることがわかった。

問題ａ）メッセージ順序

検証機能では全てのメッセージ順序をシミュレートする。そのため、現実には起こり得ないメッセージ順序により、動作不正を検出するケースである。proj-B はこれにあたる。

問題ｂ）状態遷移表の記述方法

本来、状態遷移表に記述すべき事項を、別の手段により記述し実現したために、検証ツールが問題点として検出したケースである。例えば、複数の状態に分けるべき箇所を、状態遷移表上は１つの状態にまとめてしまい、プログラミング言語により状態分けする記述を行ったという例があげられる。proj-C はこれにあたる。

なお、proj-A、proj-D は上記２つの問題の組み合わせである。

これらのうち、問題ａ）について以下に示す。問題ａ）が発生するメッセージのシーケンス図の例を図２に示す。

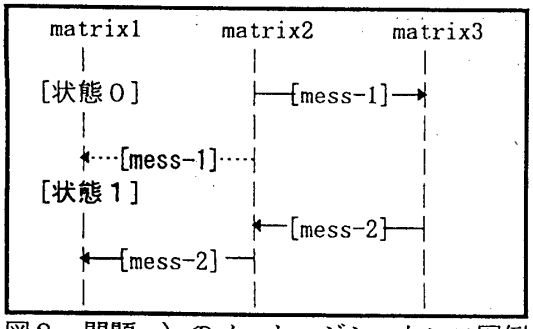

図２　問題ａ）のメッセージシーケンス図例

図２での本来の動作は以下の通り。

動作１）matrix2 からメッセージ mess-1 を matrix3 に出す。

動作２）matrix2 からメッセージ mess-1 を matrix1 に出す。matrix1 は mess1 を受け取ると状態０から状態１へ遷移する。

動作３）matrix1 は matrix3 から matrix2 を経由して送られた mess-2 を受け取る。

しかしながら、検証機能では検証精度をあげるため実際には起こり得ないメッセージ順序もシミュレーションする。図２で言えば、動作２と動作３の順序を逆にした場合も検証する。この場合は、matrix1 は mess1 を受け取っておらず、従って、状態も状態０のまま遷移していない。そのため不正遷移が発生したことを検出する。

5．評価

問題ａ）のケースは、製品として開発したソフトウェアでは、図２の例で示したようなメッセージ順序の逆転が発生しない仕掛けがあり、実用上の問題はない。しかし、現実問題として、設計者が全く意図しないメッセージの送受信が発生することがあり、ツールでこのようなケースをシミュレートし、その結果をレビューする意義は大きい。

問題ｂ）のケースは設計支援環境での支援対象を状態遷移表のみとしたため、現段階では適用対象外の問題であると考える。しかし、実用的なソフトウェア開発の支援を行っていく上で、無視できない問題である。

6．まとめ

状態遷移表設計支援ツールを開発し、その検証機能を、製品として流通しているソフトウェアで使用されている状態遷移表に適用した。その結果、幾つか問題を検出したものの、実用上、問題にならないことが判明した。しかしながら、今回検証対象としたのは既に製品として流通しているソフトウェアの開発段階で設計された状態遷移表である。設計支援を充実させていくためには、今後、不良を含む可能性の高い設計途中の状態遷移表の検証を行っていくことが重要である。また、問題ｂ）への対応も必要であると考える。

参考文献

[1]西木、他：到達可能性解析に基づく通信ソフト検証方法の提案；情報処理学会第 44 回全国大会論文集（1992）