4L-04

UML 状態遷移図とコードのリアルタイム双方向変換ツールの開発

Development of real-time bidirectional conversion tools between UML state transition diagrams and code

萩原　俊†　　佐々木　晃†

法政大学　情報科学部†

1. はじめに

統一モデリング言語(UML)の一つに，様々なオブジェクトの状態がどのように移り変わるかを表す図として状態遷移図がある．状態遷移図の作成は状態遷移の一覧性や仕様の漏れや抜け，修正箇所の発見の容易性の向上に繋がるというメリットを持っている．そのため，ソフトウェアの設計をする際に予め状態遷移図を作成し，それを基にコーディングをする場面は多くある．その手間をなくすため，状態遷移図から自動的にステートマシンのコードを生成するツールの開発は興味深い研究とされ，これまで様々なツールが提案されている．

本研究では、図の更新が即座にコードに反映されるツールを開発する。また、生成されたコードを編集した際にも、図の更新が行われるツールを目標とする。既存のツールの，コード編集後の状態遷移図を確認することができないという問題点を解決し，よりソフトウェア設計に活用しやすいツールを提案する．

2. 研究背景

状態遷移図によるモデリングはソフトウェアを設計することに役立ち，モデリングからコードの骨格を生成することも可能である．既存の状態遷移図を描写できるツールでは，状態遷移図の要素とプログラム構造に対応がなく，状態階層や状態遷移をコードとして生成する機能を持っていない．そのため，これまで状態遷移図から自動的にステートマシンのコードを生成するツールの開発は注目され，Rhapsody [1]や dCode [2], SMConverter [3]などのツールが提案されている．

しかし，既存のツールでは基本的に状態遷移図からコードへの変換のみであるため，コードの変更が状態遷移図に反映されないという欠点を持っている．コード生成後に状態名や遷移イベント名，遷移先の変更が必要となるような，図とコードを照らし合わせながらコードを書き換える場面も想定されるため，このようなツールにはコードから図への変換機能があることが望ましい．

3. 目的と手法

本研究はステートマシンに基づくソフトウェアシステムの設計・実装支援として，状態遷移図に基づいたソフトウェア設計の実現を目的とする．本研究では，特に，実用的なソフトウェアシステム開発に利用可能な状態遷移図の表現を扱えることを目標とする．また，状態遷移図とコードのリアルタイム及び双方向の変換を実現することで，コーディングに関して，既存のツールよりソフトウェア設計に実用的なツールを提案する．

ソフトウェア設計に利用可能なツールを作るためには4つの要素が必要であると考える．

1 つ目は十分に状態遷移図を表現できることである．状態階層や状態の並列性等の状態遷移図に必要な性質や要素を十分に表現できるようにし，また状態遷移図での各要素の位置の自由度を上げることで，状態遷移図の表現能力を広げる．

2 つ目は生成されるコードの利便性が高いことである．それぞれの状態をオブジェクトとして考えるオブジェクト指向のコードを生成することで，生成されたステートマシンのコード及びそれを用いるシステム側のコードで状態が扱いやすくなる．本研究では SMConverter で生成されるコードを参考に，システム側のコードから扱いやすいように再検討したものを採用している．

3 つ目はコードの変更が状態遷移図に反映されることである．本ツールの有用性の一つとして，コード作成と同時に状態遷移図を残し，その後の設計で参考にできるという点があるが，コード生成後にコードを編集してしまうと図とコードの互換性が失われてしまう．そのため，図とコードの対応が取れることが必要となってくる．そこで，本ツールではコードから図への変換を実装することで，いずれの変更においても図とコードの互換性を失わないようにする．

4 つ目は生成されるコードの骨格を理解しやすくすることである．生成されたステートマシンのコードはシステム側のコードから用いるため，ステートマシンのコードの骨格の理解が推奨される．ステートマシンのコードの理解のために，図とコードのリアルタイムでの変換及び変更箇所の強調を実現する．それにより，状態遷移図の構成と同時進行でのコードの生成が見て取れる．

4. 提案ツールの概要

本研究で提案するツールでは基本的に一つのウィンドウで UML 状態遷移図を表現し，コードの編集を行う．

4.1. ツール画面

提案ツールの画面を図 1 に示す．ウィンドウは大きく 2 つに分かれており，左側は状態遷移図を構築する GUI, 右側はコードを生成，編集するテキストエディタである．

GUI の画面には上部に 3 つのボタンを配置しており，それぞれ状態，領域を分ける棒(以下，「領域棒」とする)，履歴を GUI のキャンバス上に生成するボタンとなる．また，一定条件で初期状態を指す円(以下，「初期円」とする)を生成する．生成したコンポーネントを組み合わせて状態遷移図を構成する．

4.2. 機能の概要

本ツールには大きく分けて 2 つの機能が備わっている．

1 つ目は状態遷移図の構築である．ボタンや自動で生成される状態遷移図の要素を GUI 上で組み合わせることで基本的な状態遷移図は表現することができる．状態遷移図の構築を進めると，そのステートマシンのコード及び簡易的なシステム側のサンプルコードがテキストエディタ上に生成される．

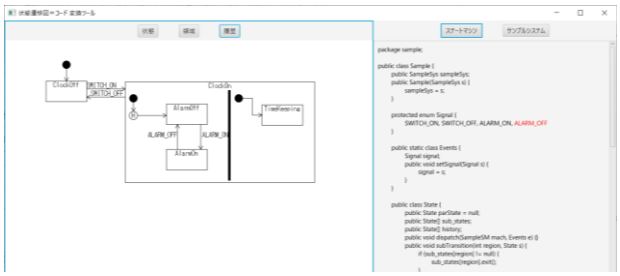

図 1　実装したツール画面

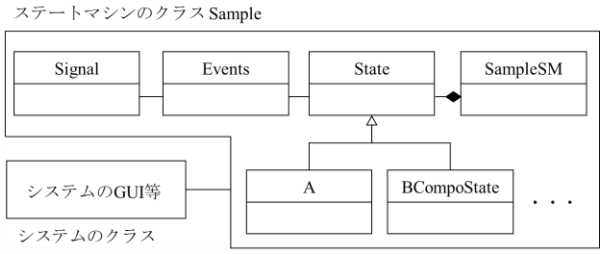

図 2　ソフトウェア設計におけるクラス図

2 つ目はコードの編集である．GUI の操作で生成されたステートマシンのコード及びシステム側のサンプルコードを編集することができる．ただし，ステートマシンのコードに関しては，遷移先や遷移イベント名などの部分的な変更が考えられる箇所のみ編集が許されており，書き換えることでその変更が状態遷移図に反映される．

5. 提案ツールの実装

本研究では，Java によって提案するツールを作成した．また，図より生成されるコードはいずれも Java である．

5.1. GUI

本ツールの GUI では，UML 状態遷移図の基本的な要素を表現することができる．(図 1 の左)

状態は四角形で表し，中に別の状態を含むと，状態の親子関係を示す複合状態となる．状態の他に縦棒で表した領域棒や黒丸で表した初期円，アルファベットの「H」で表した履歴を中に含む．領域棒は状態内を区分し，初期円は初期状態を指し示し，履歴はその領域の状態を保存することができる．

各コンポーネントに対し，いくつかの操作を可能にすることで状態遷移図構築の自由度を高めている．

コンポーネントはマウスのドラッグアンドドロップで 3 種類の操作を行うことができる．コンポーネントの内部，四隅，上下左右をドラッグすると，それぞれコンポーネントの移動，拡大縮小，遷移の矢印を引く操作となる．

状態に関してはマウスの右クリックでさらに「削除」，「entry」，「exit」という 3 種類の操作ができ，それぞれ状態の削除，各状態の入出時アクションの設定ができる．

また，これらの操作により状態遷移図上で変更があった時に，コードが書き換わるのと同時に変換された箇所を次の操作があるまで赤色で強調する機能を持っている．

5.2. 生成コード

本ツールでは GUI で状態遷移図を構築することで，ステートマシンのコード及びシステム側のサンプルコードを生成する．

前提として本研究では，ソフトウェアをステートマシンのクラスとそのステートマシンを用いるシステムのクラスの最低 2 クラスで設計することを想定しており，本ツールではその 2 クラスをそれぞれ 1 つのファイルとして生成する．それらのクラスの関係図は図 2 のようになっている．

本ツールで生成されるステートマシンのコードの構成には，既存ツールの SMConverter で生成されるコードの不要なメソッドや処理を省き，各状態の管理方法を変更した，オブジェクト指向のコードを採用している．ステートマシンを表すクラスではテンプレートとして State，Signal，Events を内部クラスとして持ち，また，ステートマシンの基盤として全状態を管理するクラス(～SM)を持ち，ユーザーの入力に応じて，状態クラスを増やす．

5.3. コードの編集

生成されるコードはテンプレートに沿って構成されているため，遷移先や遷移イベント名などの部分的な変更が考えられる箇所のみ編集が許されている．本ツールでのコードから図への変更可能箇所は以下のようになっている．

- 使用するイベント名の事前追加・消去
- 各状態名
- ステートマシン全体での初期状態
- 複合状態内の各領域の初期状態
- 履歴使用時のデフォルト遷移先
- 状態間の遷移イベントとその遷移先

6. 議論

単純な状態遷移図と複雑な状態遷移図でのソフトウェア設計を考えると，複雑になればなるほど，ステートマシンのコード記述は長く，複雑なものになるのに対し，本ツールでの状態遷移図の構築の難易度は大きくは変わらないものと考えられる．そのため，扱う状態遷移図が複雑であるほど，本ツールでのコード生成の恩恵が大きくなると言える．

また，本ツールの課題は，状態遷移図の表現能力をさらに広げることである．UML 状態遷移図では状態間の遷移に本ツールで用いているイベント名の他に，ガード条件や時間経過などが存在する．そのため，それらを実装することでよりソフトウェア設計に実用的なツールにすることができる．

7. むすび

本研究では状態遷移図とステートマシンのコードをリアルタイムで双方向に変換するツールを開発した．ツールでは UML 状態遷移図の基本的な要素を表現でき，その構成によりオブジェクト指向のコードを生成できる．本ツールは既存のツールと比べ，コードから図への変換機能や変更箇所を強調する機能等を加えることにより，ソフトウェア設計に実用的なツールとなった．

文　献

[1] I.-L. Inc, "Rhapsody," [Online], Mar. 2010.

[2] JAUHAR ALI, 田中二郎, “オブジェクト指向方法論(OMT)に基づく動的モデルからの Java コード生成”, 情報処理学会論文誌, vol.39, no.11, pp.3084-3096, Nov 1998

[3] S. E. V. and P. SAMUEL, "Automatic Code Generation From UML State," IEEE, vol.7, pp.8591-8608, January 23, 2019.