シーケンス図とステートチャート図の相互変換に着目したソフトウェア分析・設計に関する研究

長谷川 英男 † 高田 眞吾 † 土居 範久 †‡
† 慶應義塾大学大学院理工学研究科 ‡ 中央大学理工学部

1 はじめに

ソフトウェア開発において，要求分析段階と設計段階は反復して行う場合がある．要求分析段階では，システムに対する顧客の要求を分析し，シーケンス図によってモデル化する．設計段階では，要求分析で得られたモデルを用いて実装するアーキテクチャに基づいた設計を行い，クラス図やステートチャート図によってモデル化する．

本研究では，要求分析段階と設計段階における問題点を分析し，それを解決するための方法としてシーケンス図とステートチャート図を相互に変換する方法を提案する．

2 シーケンス図とステートチャート図

シーケンス図 シーケンス図とは，オブジェクト間のメッセージのやり取りを記述する図である．シーケンス図と呼ばれるものとして，Message Sequence Charts (MSC) [1] や UML シーケンス図などがある．本研究ではシーケンス図は MSC を用いた．MSC では，bMSC と hMSC と呼ばれる 2 つの図からなる (図 1)．bMSC はオブジェクト間のメッセージのやりとりを表す．hMSC は四角で表したノードが bMSC への参照を表し，bMSC 間のつながりを矢印で表す．また開始ノードは下向きの三角形，終了ノードは上向きの三角形であらわす．hMSC を用いることで分岐やループなども明示することができる．

ソフトウェア開発において，要求は一般的には自然言語で記述する．開発者は要求を分析し，MSC を記述しモデル化する．図 1 は，銀行の ATM の認証に関する MSC である．ユーザがカードを挿入する場面において 2 通りの bMSC (insert_valid_card と insert_invalid_card) を記述した．この例のようにユーザの振る舞いややりとりされるメッセージなどの各状況の相違に基づいて，複数の bMSC を記述することが一般的である．その結果，一つの顧客の要求を分析するためには，複数の bMSC を必要とする．

ステートチャート図 ステートチャート図とは，オブジェクトのイベントやアクションによる状態遷移を記述する図である．ソフトウェア開発では設計モデルを表現するためにステートチャート図を一般的に利用する．シーケンス図があらかじめ得られている場合，ステートチャート図はシーケンス図を元に作成することができる．

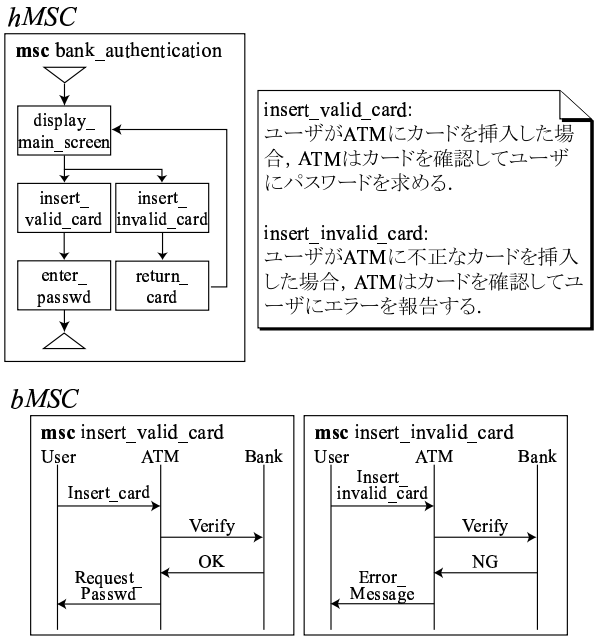

図 1: ATM の認証を MSC で記述した例

3 問題点

ソフトウェア開発では，前述のように要求分析段階で複数の bMSC が得られる．このため， 2 つの問題が発生する．

問題点 1 要求分析段階から設計段階への移行に関する問題

MSC が複雑になるにつれ，設計への移行時にステートチャート図に書き直すべき bMSC が多くなる．そのため，要求分析の結果を設計へ反映することが困難になる．

問題点 2 設計モデルの変更に関する問題

開発者は設計を進めるに従い設計モデルを変更する．設計モデルの変更が顧客の要求を満たすかどうかをチェックするため，設計の変更を要求分析段階にフィードバックする必要がある．しかしながら，MSC が複雑であるほど，設計段階における変更をフィードバックすることが困難になる．

"A Study on Software Analysis and Design based on the Iterative Development of Sequence Diagrams and Statecharts", Hideo Hasegawa, Shingo Takada and Norihisa Doi, Keio University

4 シーケンス図とステートチャート図の相互変換

本研究では前述の問題点を解決するため，シーケンス図とステートチャート図の相互変換を行なう機構を提案する．本機構はシーケンス図として MSC を用い，ステートチャート図として UML Version 1.5 のステートチャート図を用いた．

4.1 概要

提案する機構は，3 節で述べた 2 つの問題点を解決するため，以下の 2 つの操作を行うツールを提供する．

変換プロセス 設計への移行 (問題点 1 に対応)

要求分析段階において開発者は要求を分析し MSC を記述する．ツールは変換プロセスにより，MSC からステートチャート図を生成する．

反映プロセス 設計からのフィードバック (問題点 2 に対応)

設計段階では，開発者はツールを用いてステートチャート図を修正する．ツールは反映プロセスにより，ステートチャート図への変更を MSC に反映する．開発者は反映がなされた MSC を用いて，要求分析段階に戻り開発を続行する．

提案するシステムの全体像を図 2 に示す．本ツールを用いて， MSC とステートチャート図を整合性を保ちつつ交互に変換と反映を繰り返すことで，要求分析段階と設計段階の反復的な開発を行う．これによって徐々にモデルが精練され，最終的には初期に記述された MSC よりも精練された MSC とステートチャート図を得ることができる．

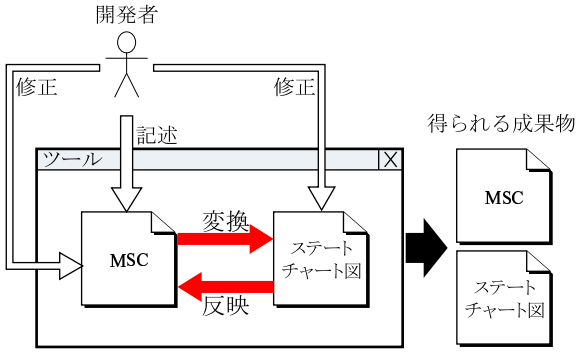

図 2: 提案するシステムの全体像

Whittle ら [2] は，複数の独立したシーケンス図からステートチャートを生成した．本研究は [2] の用いた手法を拡張し，さらにステートチャートからシーケンス図へのフィードバック機構を構築した．

次に，変換プロセスと反映プロセスの詳細について述べる．

4.2 変換プロセス

変換プロセスでは，MSC とその事前・事後条件を入力として受け取り，MSC の矛盾を検出した上で，ステートチャート図を出力する．主な手順は次のとおりである．

まず開発者は，各 bMSC が持つメッセージに対して状態変数による事前・事後条件を設定する．事前条件とはメッセージ送信時に満たすべき条件で，事後条件とはメッセージ受信時に満たすべき条件である．設定した事前・事後条件を用いたデータフロー解析により bMSC および hMSC の矛盾を検出する．矛盾が検出された場合，開発者はそれを修正する．

次にステートチャート図を生成する．生成はオブジェクトごとに行い，1 つのオブジェクトにつき 1 つのステートチャート図を生成する．はじめに，各 bMSC ごとに (部分) ステートチャート図を生成する．MSC において，あるメッセージの受信終了から次のメッセージの送信開始までの間を状態とみなして，状態変数を割り当てる．状態間を遷移で結ぶ．入力メッセージをイベント，出力メッセージをアクションとみなし，遷移にラベル付けを行う．次に各部分ステートチャート図間に hMSC に従い ϵ 遷移を用いて統合する．後処理として，状態の入れ子化などを行う．

4.3 反映プロセス

反映プロセスでは，ステートチャート図の修正部分を入力として受け取り，修正部分を反映させた MSC を出力する．修正は，遷移の追加のみを想定した．主な手順は次のとおりである．

はじめに，ある状態から別の状態へ何らかの遷移が発生することを指定する．次に状態遷移中に発生するメッセージを，部分 bMSC に記述する．部分 bMSC は追加する遷移の内容を指定するために用いる．

次にツールは開発者が記述した部分 bMSC 図を MSC に自動的に反映する．ツールは，bMSC へのメッセージの追加や，hMSC への bMSC の追加，それに伴う hMSC への分岐の追加を自動的に行う．

5 まとめ

本研究では，要求分析段階と設計段階における反復的な開発をサポートするための，シーケンス図とステートチャート図の相互変換を行なう機構を提案した．

謝辞

本研究は，文部科学省科学技術振興調整費「環境情報獲得のための高信頼性ソフトウェアに関する研究」の支援による．

参考文献

[1] ITU-T, Recommendation Z.120, 1999.

[2] J. Whittle and J. Schumann. “Generating Statechart Designs From Scenarios”, Proceedings of the 22nd International Conference on Software Engineering, pp.314-323, 2000.