要求分析と基本設計間のトレーサビリティ確保のためのユースケース記述変換ツール

吉野魁人[†1]　松浦佐江子[†2]

概要：要求分析段階におけるユースケース分析では，一般にテンプレート形式の自然言語により，ユースケースを記述する．記述要素は，事前条件，事後条件，アクター，基本フロー，代替・例外フローである．ユースケース記述に基づき，シーケンス図を用いて記述内容を分析し，振舞いをクラスに割り当てる．しかし，人手による解釈の違いによってはこの振舞いの割り当てに漏れや誤りが発生する可能性がある．本研究ではユースケース分析においてアクティビティ図を利用することにより曖昧性を排除し，ユースケース記述と設計段階で作られるシーケンス図間のトレーサビリティを確保するために，ツールを用いた図の自動変換の手法を提案する．

A Conversion Tool for Ensuring Traceability between UML Requirements Analysis Model and the Basic Design Model

KAITO YOSHINO[†1]　SAEKO MATSUURA[†2]

Abstract: In the use case analysis at the requirements analysis stage, the use case is generally described by the natural language of the template form. This descriptive elements are preconditions, postconditions, actors, basic flows, and alternative / exception flows. Based on the use case description, software designer analyzes the description content using the sequence diagram and assigns the behavior to the class. However, depending on difference in interpretation of software developer, leakage and errors may occur in this behavior assignment. In this study, the authors eliminate the ambiguity in use case analysis by using the activity diagram. Furthermore, we propose a method of automatic conversion to secure the traceability between the activity diagram and the sequence diagram created at the design stage by using the tool.

1. はじめに

ソフトウェア開発における要求分析では，システムの境界におけるユーザや他システムとのインタラクションに着目して，システムのユースケースを分析する．ユースケース分析では，一般にテンプレート形式の自然言語により，ユースケースを記述し，シーケンス図を用いて記述内容を分析し，振舞いをクラスに割り当てメソッドを決定する．要求分析で定義した振舞いフローを踏襲して，クラス定義に基づき，各クラスのメソッドを定義することにより，要求分析と設計間のトレーサビリティが確保できる．

しかし，人手による解釈ではモデルの観点が異なるため設計の段階に漏れや誤りが発生する可能性がある．本稿では，UML（Unified Modeling Language）を用いて，インタラクションを実現する振舞いフローを定義したアクティビティ図とデータを定義したクラス図によりユースケース分析を行ない，設計の起点となるシーケンス図をツール上で生成することにより，要求分析と設計間のトレーサビリティ確保を支援する手法を提案する．

2. 要求分析・設計段階モデルのトレーサビリティ確保の問題点

ユースケース分析では，一般にテンプレート形式の自然言語により，ユースケースを記述する．記述要素は，事前条件，事後条件，アクター，基本フロー，代替・例外フローである．ユースケース記述に基づき，シーケンス図を用いて記述内容を分析し，振舞いをクラスに割り当てメソッドを決定する．しかし，各フローの記述や条件も自然言語の文であるため，メソッドを決定するには曖昧性を排除し，ユースケースとして明確なアクション系列により，レビュー効率や検証可能性を向上させる必要がある．

そこでわれわれはユースケースの明確性の向上と実現可能性を検証するために，要求分析において，システムのユースケースをアクティビティ図とクラス図を用いて分析を行っている[1][2]．ユースケースにおける振舞いを「アクター」「インタラクション」「システム」のパーティションに分け，アクターとシステムのやり取りを主体と役割を明確にしたアクション系列として定義する．一方，システムにおけるデータはクラスとその属性として整理できる．これらが見えてくると，アクションとデータの関係も見えてくるので，オブジェクトノードとしてアクティビティ図に記述することができる．これにより，設計段階では，各クラスの責務に合わせたメソッドの割り振りを考えやすくすることができる．

ユースケースに記述したアクティビティ図の振舞いをクラスのメッセージシーケンスとして表すことを，アクティビティ図のシーケンス図へのマッピングと呼ぶ．マッピングを行うことによりユースケースを要求分析の段階から設計の段階へとトレーサビリティを確保して移行することが出来る．

マッピングを行う際にはユースケース記述として適切な内容が定義されている必要があるが，アクティビティ図のアクションは本来自由形式の自然言語記述であり，処理内容が一意に解釈出来ない曖昧なものを書くことがある．この場合，解釈の違いによっては誤りや漏れが発生したマッピングを行う可能性がある．また，アクションの対象となるデータをオブジェクトノードとして書かずにアクション内にのみまとめて書くこともある．このため，

†1 †2 芝浦工業大学
Shibaura Institute of Technology

マッピングを行う際に振舞い元や役割が不明となる場合がある.

3. 本研究のアプローチ

設計者は本来どのように対応付いているかを考えシーケンス図へのマッピングを行う必要があるが，既に要求分析の段階で必要と判明したクラスに対するアクション系列は分析されている．これをそのままシーケンス図にマッピングすることで，要求分析で記述された振舞いは漏れ・誤りがなく設計の段階へ持ち込める．そこで本研究ではこのマッピングを自動化するツールを作成する．

マッピングの自動化を実現するためには，3 節で述べた問題点を解決するためにアクティビティ図の記法をより形式的にする必要がある．松井ら[2]は，ユースケース記述として書かれるアクティビティ図の形式の定義と以下の意味的な評価を行っている．

- 語句が記述されたパーティションの行為として妥当でなければならない.
- 基本・代替・例外は，システムが何らかの条件を判断し，判断結果に応じてつぎの行為を定めることで，そのフローが定義できる．この際，判断結果は，判断するアクションにつづくディシジョンノードと分岐フローのガードとして記述すべきである.

などが挙げられている．以上の点に加えて，

- アクションがあるデータに対して CRUD（Create, Read, Update, Delete）の振舞いや入出力を行っているということを明確にするために，アクションから続けて繋いでデータ（オブジェクトノード）を書く.
- ディシジョンノードを用いて条件分岐をした場合はマージノードで分岐を対応付けて終了を表す.

これらをユースケース記述としてのアクティビティ図の記法として定義する.

4. ユースケース記述変換ツール

本ツールは UML モデリングツールの一つである astah* Professional[3]内のプラグインとして実装した.

4.1 変換規則の概要

ツール上で行うマッピングを本研究では変換と呼ぶ．アクティビティ図の変換の規則を意味的な観点から表 1 のように定義した．ディシジョンノードのガード条件は条件分岐の際に具体的な条件を書くためのものである.

表 1　アクティビティ図の変換規則

アクティビティ図の要素	シーケンス図の要素
アクション	メッセージ
パーティション	ライフライン
オブジェクトノード	
ディシジョンノード（ガード条件付き）	複合フラグメント

4.2 ライフラインの生成規則

アクションが属するパーティションはそのアクションの振舞い元となる．これらパーティションから対応するライフラインを生成する．さらに，後述するメッセージの生成ではアクションの先にあるオブジェクトノードからライフラインを識別する必要があるため，このオブジェクトノードにあるクラスを用いてもライフラインを生成する．変換の規則は表 2 のように定義した.

表 2　ライフラインの変換規則

アクティビティ図の要素	ライフラインの種類
システムパーティション	コントロール
アクターパーティション	バウンダリ
インタラクションパーティション	
アクション先のオブジェクトノードクラス	エンティティ（クラスと同名）

4.3 メッセージの生成規則

アクティビティ図のアクション内に書かれる文を読み取り，シーケンス図内のシグネチャが無い状態のメッセージとして変換する．メッセージの送信先対象とするライフラインはアクションに続くオブジェクトノードを読み取ることで判断する．このとき，オブジェクトノードがアクションの先に無い場合は自分自身（アクションの属するパーティション）に対応したライフラインにメッセージを送信する．また，アクション内で特定の動詞を使用した場合はさらに詳細なメッセージとして記述する．今回は頻繁に用いられる create メッセージに着目した. 具体的な例をあげると，「**を生成する」という動詞を読み取り，オブジェクトの生成と判断することでシーケンス図に create メッセージを発生させる.

4.4 ツールによる生成物

本ツールを用いることにより，シグネチャは定められていないがアクティビティ図のアクションに対応したメッセージが作られたシーケンス図が出力される．これにより要求分析の結果を踏襲した設計モデルを用いて, 最終的な設計モデルの作成を支援する．利用者はクラス毎にシグネチャの未定義部分（戻り値．パラメータ）を決定し，要求分析の段階では分析が難しいクラスの委譲を追加する作業のみを行えばよいことになる.

5. おわりに

本稿では要求分析段階で得られたアクティビティ図を踏襲したシーケンス図を自動生成することによる基本設計の支援方法を提案した．これにより，各段階で作られたモデルのトレーサビリティの確保が容易になると考えられる.

今後は，ツールの未完成部分の完成とアクティビティ図の記法を省略して書いた場合も同様に変換が行えるようにする方法の考案を課題とする．また，基本設計後の詳細設計で更新されたシーケンス図とアクティビティ図間においてもトレーサビリティが確認できるような仕様を検討する.

参考文献

[1]松浦佐江子．ソフトウェア設計論 ―役に立つ UML モデリングへ向けて―．コロナ社, 2016.
[2]松井，奥田，野呂，岡田，加藤，渡辺，松浦．モデリング能力育成を目的としたユースケース記述の自動評価手法．電子情報通信学会論文誌, VOL.J97-D, NO.3, p.465, 2014.
[3]astah* Professional, http://astah.change-vision.com/ja/product/astah-professional.html, (参照 2018-07-20).