B-001

複雑なモデルに対応した形式的ソフトウェア合成システムの提案

A Proposal of Formal Software Synthesis System Considering Complex Models

田中 涼介† 織田 健†
Ryosuke Tanaka Takeshi Oda

1 はじめに

ソフトウェアの大規模化や複雑化に伴う開発コストの増大や信頼性の低下に対し、我々は形式手法に基づいた部品を再利用することでソフトウェアの合成を行う MSSS 手法を提案している [1]。MSSS 手法によるソフトウェア合成システムでは、モジュール構造を持つ要求への対応とユーザインタフェースへの考慮が課題であった [2]。本稿ではこれらの課題に対応するための手法の拡張とシステム構成を提案する。

2 背景と目的

2.1 形式手法 B Method

B Method は形式手法の一種で、抽象的な仕様を表すモデルと、これを段階的に詳細化したリファインメントや実装を集合論と一階述語論理に基づいて記述し、それぞれの無矛盾性と段階間の整合性を検証しながらソフトウェアを開発する [3]。詳細化は詳細化前後の変数等が満たす条件 (リンク不変条件) を記述することで行う。複数のモデルを用いてモジュール構造を構成し、各々を個別に詳細化することで複雑なソフトウェアを開発できる。

2.2 MSSS 手法

MSSS 手法は、B Method に基づいたソフトウェア部品を生成し再利用することで、要求を記述したモデルを満たすソフトウェアを合成する手法である [1]。ソフトウェア合成は図 1 のように、要求モデルの細分化、モデルの一致する部品の検索、結合可能な部品の組み合わせの選択、部品の結合の 4 つの工程からなる。また、要求が単一のモデルで表される場合を対象として、実際に合成を行うシステムを開発した [2]。

2.3 部品の結合

2 つの部品が結合できるためには、それらに共通する変数の詳細化方法が一致している必要がある。部品の含む識別子名は要求モデルに合わせることで取得時に部品間で統一され、結合可否の判定が可能となる。この条件を満たしていれば、リファインメントの段数が異なっていてもその追加や統合により段数を揃え、それぞれの制約条件や操作を連結することで結合できる。

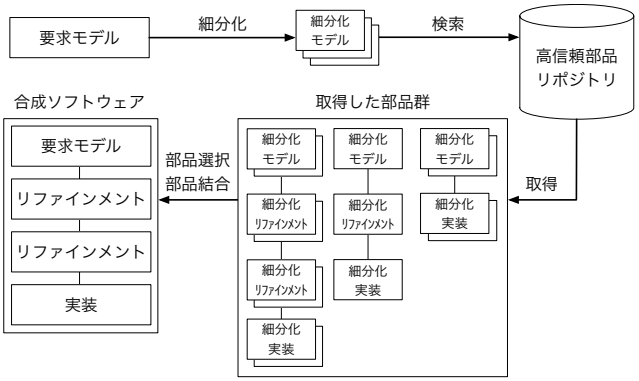

図 1: ソフトウェア合成の概要

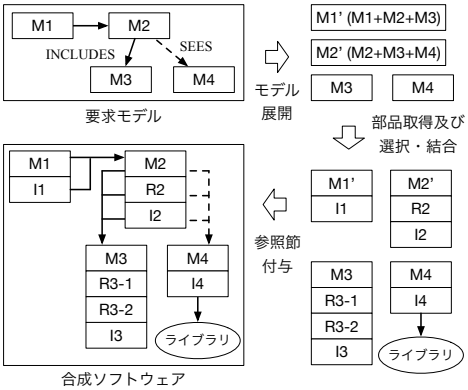

図 2: モデル展開を取り入れたソフトウェア合成の概要

2.4 モデル展開

MSSS 手法において要求がモジュール構造を持つ場合には、図 2 のように要求の各モデルに対して参照先の変数の宣言と制約条件の展開をする手法が提案されている [4]。展開前後でモデルの振る舞いは同一であると保証されているため、展開により単一のモデルの場合と同様にソフトウェア合成ができるようになると言える。合成後は要求に基づいて再び参照関係を付与し、展開した変数や制約条件を除去することでモジュール構造を構築する。

2.5 不足部品の雛形生成

細分化されたモデルを満たす部品が不足した場合には、他の部品から得られる情報を書き入れた部品の雛形を生成し、ユーザが部品の実装を記述する [5]。雛形には細分化モデルが含む変数や定数の他の部品における制約条件や、操作の入出力などが記述される。

2.6 システム構築における課題

システム構築にはいくつかの課題がある。まず、従来のモデル展開では推移的な参照関係である INCLUDES のみが考慮されていて、非推移的な参照関係である SEES が未考慮である。また、リファインメントを含む部品の雛形生成手法が未提案である。これらを考慮して MSSS 手法を拡張した上で、実際の利用の流れを整理し、利用性を考慮したシステムを構築する必要がある。

2.7 研究目的

本研究は、単一のモデルを対象としていた形式的ソフトウェア合成システムにモデル展開を組み入れることで複雑なモデルに対応し、さらに利用の流れを考慮したユーザインタフェースを持つシステムの提案を目的とする。

†電気通信大学大学院情報理工学研究科情報学専攻

3　複雑なモデルに対応した形式的ソフトウェア合成システム

3.1　モデル展開の拡張

INCLUDES と SEES の参照範囲の違いを考慮したモデル展開は、以下の手順で行う。

1. SEES を用いている要求モデルに対して、SEES による参照先の変数、定数が含まれる制約条件を除去した SEES 除去済み要求モデルを作成する。
2. SEES 除去済み要求モデルに対して、INCLUDES による参照を辿って全ての情報を展開する。
3. 元の要求モデルの情報を SEES 除去済み要求モデルに重複を除いて結合する。
4. 各モデルの SEES による参照先の情報を展開する。

3.2　不足部品の雛形生成の拡張

リファインメントを含む部品を考慮した不足部品の雛形生成は、以下の手順で行う。

1. 選択された部品を全て結合する。雛形は結合後部品と同じ数のリファインメントを持つ。
2. 部品が不足した細分化モデルが含む変数と定数に関する宣言、型制約、リンク不変条件を結合後部品の各リファインメント及び実装から抜き出して雛形に記述する。
3. 細分化モデルが含む変数以外の変数を含まないその他の制約条件を抜き出す。
4. 2-3. で追加した条件が含む定数に関する宣言や制約を抜き出す。
5. 雛形に詳細化を行っていないリファインメントがあれば削除する。

3.3　システムの利用の流れ

システム利用の際には合成の状況を保持する必要があると考えられるため、1 組の要求モデルごとにプロジェクトディレクトリを作成し、その中で処理を行う方針とする。システムの利用は以下の手順で行う。

1. ユーザの指示でシステムがプロジェクトを作成する。
2. ユーザが要求モデルのパスを指定し、システムはモデルをプロジェクトにコピーする。
3. システムがモデル展開・モデル細分化・部品検索・部品の組み合わせの列挙を順に行う。
4. システムがモデルに含まれる変数を表示し、ユーザは変数を実装するライブラリを指定する。
5. システムはユーザの指定に合致する組み合わせを表示し、ユーザは組み合わせを 1 つ選択する。
6. システムは不足部品の雛形を提示し、ユーザは雛形を元に部品を記述、検証する。
7. 不足部品がなくなったらシステムは部品結合と参照関係の付与をし、合成ソフトウェアを出力する。

3.4　システム構成の提案

図 3 にソフトウェア合成システムの構成を示す。字句・構文解析を伴い形式仕様を書き換える処理には Standard ML (SML) を用い、その他の処理には Python を用いる。取得した部品をオブジェクトとして扱い、これに対する操作を順に行う方針でソフトウェアを合成する。また、`SML Runner` は SML の対話環境の起動と必要な関数の読み込みをあらかじめ行い、Python 側からはこの環境に対してコマンドを送る。利用時にはユーザのコマンドを `msss` が処理し、`MSSS Project` が指示に応じて 3.3 節のようにソフトウェア合成の処理を行う。

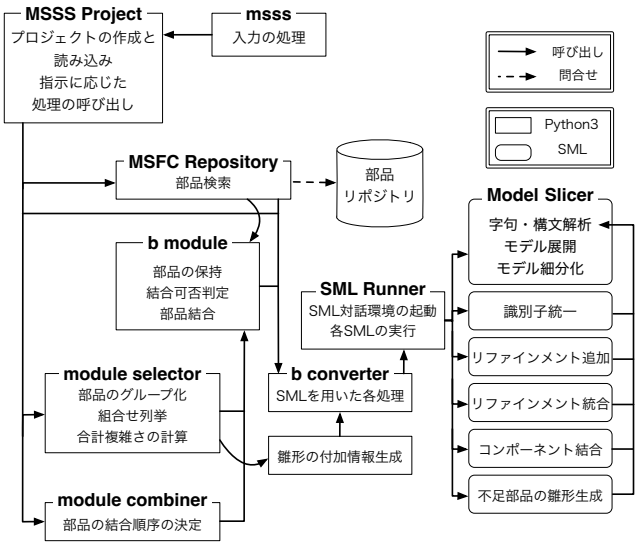

図 3: ソフトウェア合成システムの構成

4　考察

先述のシステムにより、モジュール構造を持った要求を満たすソフトウェアを合成できると考えられる。本研究の内容はシステムの提案に留まっているため、今後は実際にシステムを構築しソフトウェア合成実験を行うことで、システムが要求を満たすソフトウェアを合成することを検証する必要がある。実験では、INCLUDES や SEES を用いた様々なモジュール構造だけでなく、リファインメントの数や変数の詳細化方法のバリエーションを持つ部品や要求モデルを用いることが望ましい。

5　おわりに

本稿では、従来の形式的ソフトウェア合成システムをモジュール構造を持つ要求に対応させるための、MSSS 手法の拡張とシステム構成について提案した。今後はシステム構築と、様々な部品や要求モデルを用いた実験を行うことで手法の妥当性を検証する必要がある。

参考文献

[1] 中村丈洋. B Method における部品再利用によるソフトウェア合成と高信頼ソフトウェア部品の整備. 電気通信大学 電気通信学研究科 博士 (工学) 学位論文, 2013.
[2] 田中涼介, 檜垣廉, 織田健. 小規模なモデルを対象とした形式的ソフトウェア合成システムの構築. 情報処理学会第 86 回全国大会講演論文集. 2024, vol.1, p.147-148.
[3] 来間啓伸. B メソッドにおける形式仕様記述. 近代科学社, 2007.
[4] 松田蓮, 織田健. モジュール構造と段階的詳細化に対処した形式的ソフトウェア合成手法. 第 21 回情報科学技術フォーラム論文集. 2022, vol.1, p.201-202.
[5] 大久保稜, 織田健. 抽象データ型に対応した不足部品の自動生成手法. 第 21 回情報科学技術フォーラム論文集. 2022, vol.1, p.203-204.