B-024

形式的ソフトウェア合成システムの構築と妥当性検証

Construction and Validation of a Formal Software Synthesis System

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く部品の再利用によりソフトウェアの合成を行う MSSS 手法を提案している [1]。MSSS 手法では、B Method で記述できる複雑なソフトウェア構造への対応手法が個別に提案・検証されてきたが、それらを同時に考慮した検証は行われていなかった。本稿では手法を具体化して構築したシステムとその妥当性検証について述べる。

2 背景と目的

2.1 形式手法 B Method

形式手法の一種である B Method では、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 [2]。各記述は制約条件と、状態を変化させる操作からなり、詳細化は詳細化前後の変数等の関係を表す条件 (リンク不変条件) を記述することで行う。複数のモデルによりモジュール構造を構成し、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 に基づく既存ソフトウェアから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みたす新規ソフトウェアを合成する手法である [1]。ソフトウェア合成の具体的な手順は 3 章で述べる。

2.3 研究目的

MSSS 手法には、段階的詳細化やモジュール構造といった複雑なソフトウェア構造への拡張手法がいくつか個別に提案されてきた。それらに対応してシステムを構築しソフトウェア合成手法の検証を行った結果、モジュール構造を扱う場合の部品検索手法と参照構造を付与する工程に誤りを発見した [3]。本研究では改善した手法の提案と総合的な妥当性の評価を行うことを目的とする。

3 ソフトウェア合成の手順

3.1 モデル展開とモデル細分化

ソフトウェア合成 (図 1) では、初めに要求モデルに含まれる各モデルの参照先の制約条件 (c_n) を展開して参照関係を断ち切る。次にモデル細分化では要求モデルをその操作の 1 代入文を単位として分割し、各代入文に関係する制約条件や宣言と合わせて細分化モデルとする。これが部品の仕様を表す。このとき、数学的に等価な記述が文字列上で可能な限り一致するように変換や推論を施すことで、文字列一致による部品検索が可能となる。

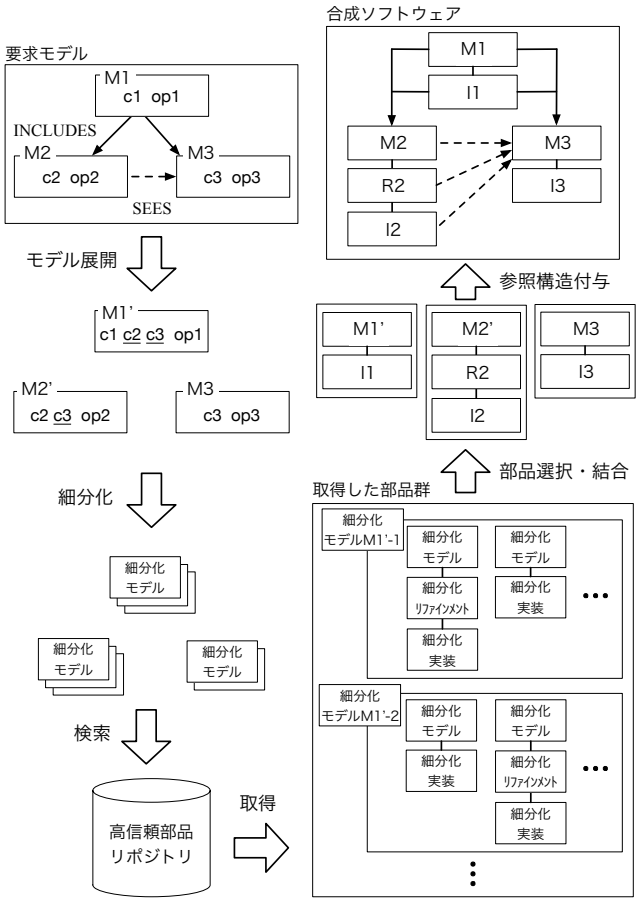

図 1: ソフトウェア合成の概要

3.2 部品検索

部品は細分化モデルとそれをみたすリファインメントや実装、そしてリンク不変条件等を抽出した付加情報の組である。部品検索では要求の細分化モデルと制約条件や代入文が一致するモデルを持つ部品をリポジトリから取得する。ただし操作の事前条件については要求より緩い部品も利用可能であるため取得し、モデル展開で展開された変数が異なる部品は利用不可であるため除外する。

3.3 部品選択

検索された部品は識別子が抽象化されているので、まず要求に合わせて識別子を置換する。次に、ある変数について異なる詳細化を行う部品同士は結合することができないため、部品の付加情報を用いて可能な部品の組み合わせを列挙する。このとき、部品に優先順位をつけることで計算コストを下げる。

3.4 部品結合

部品の組み合わせを選択したら、各部品の制約条件や代入文をそれぞれ結合することで元の要求をみたすソフ

†電気通信大学大学院情報理工学研究科情報学専攻

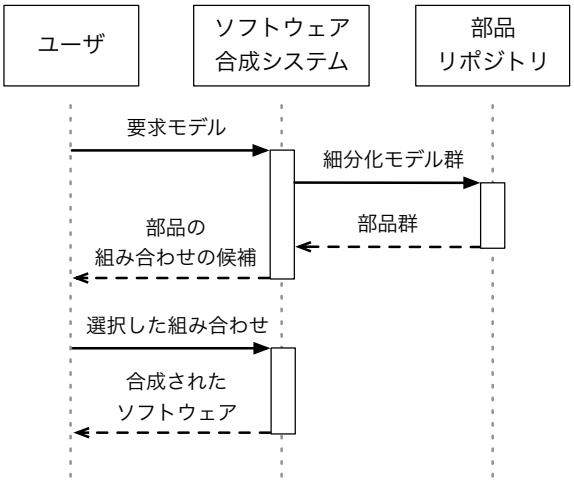

図 2: ソフトウェア合成システムの処理の流れ

トウェアが合成できる。結合する部品のリファインメントの段数が異なる場合は、詳細化を行わない段の追加や、2 段に渡る詳細化の統合により段数を揃える。実装の代入文の順序はモデルと整合するように定める。

3.5 参照構造付与

各リファインメントや実装は対応する要求モデルと同じ参照を持つようにする。初めに要求モデルと同じ参照先を異なる名前を付けて参照し、参照先の変数の一致を示すリンク不変条件を記述する。次に付けた名前に対応して利用している変数名を置換する。これにより要求に合わせたモジュール構造を再構築することができる。

4 形式的ソフトウェア合成システム

3 章の手法に従ってソフトウェア合成を行うシステムを構築した。システムは図 2 のように、初めに要求を元に部品の取得と組み合わせの提示を行い、ユーザの選択した組み合わせの部品を用いて要求をみたすソフトウェアを合成する。コードは Standard ML が 14,000 行程度、Python が 2,000 行程度となった。

5 システムの妥当性検証

5.1 実験方法

システムの妥当性を評価するために実験を行った。初めにモジュール構造を持つ 3 個のソフトウェアを記述・検証し、それらを用いて部品生成およびリポジトリへの登録を行った。これらの部品は従来手法では合成への利用に失敗したものを含む。検証には B Method による開発環境である Atelier B を用いた。次に新規要求を表すモデルを記述・検証し、これをシステムへの入力としてソフトウェア合成を実行した。最後に、出力されたソフトウェアを検証し要求をみたすかどうか確かめた。

5.2 実験結果

表 1 のように部品の生成と登録を行った。4 つのモデルからなる要求 VarietyStore を入力してシステムを実行すると 17 個の細分化モデルが生成され、それぞれに部品を取得してソフトウェアを合成することができた (図 3)。合成には Shop からの部品が 9 個、Survey からの部品が 5 個、Lane からの部品が 3 個利用された。検証

表 1: 部品生成の結果

プロジェクト	モデル数	細分化モデル数	登録部品数
Shop	6	15	14
Survey	3	8	6
Lane	2	3	3

要求モデル

```
MACHINE Trade_Controller
INCLUDES
   Food, Book, Goods_TYPE
ABSTRACT_VARIABLES
   takings
INVARIANT
   takings : NATURAL
INITIALISATION
   takings := 0
OPERATIONS
   sell_food(ff, nn) =
   PRE
      ff : FOOD &
      nn : NAT &
      food_num(ff) >= nn
   THEN
      remove_food(ff, nn) ||
      takings := min({
       takings + food_value(ff) * nn,
       MAXINT
      })
   END; …
END
```

合成された実装

```
IMPLEMENTATION Trade_Controller_i
REFINES Trade_Controller
IMPORTS
   Inst001_i.Food, Inst002_i.Book, Goods_TYPE
CONCRETE_VARIABLES takings_i
INVARIANT
   takings_i : INT &
   takings_i = takings &
   food_value = Inst001_i.food_value &
   food_num = Inst001_i.food_num & …
INITIALISATION
   takings_i := 0
OPERATIONS
   sell_food(ff, nn) =
   BEGIN
      VAR lv001 IN
         lv001 <-- Inst001_i.get_food_price(ff) ;
         IF
            takings_i + lv001 * nn > MAXINT
         THEN
            takings_i := MAXINT
         ELSE
            takings_i := takings_i + lv001 * nn
         END
      END ;
      Inst001_i.remove_food(ff, nn)
   END; …
END
```

図 3: 合成されたソフトウェア (抜粋)

の結果、証明責務は全て証明できた。

6 考察

5.2 節の結果から、以前の研究 [3] で不備が発見されたモジュール構造を持つ要求への対応手法が改善されたと言える。特に、実験に用いた要求モデルは部品が元々持っていたモジュール構造とは異なる構造を持っており、モジュール構造を持つソフトウェアから汎用的な部品を生成し再利用できたと言える。よってモジュール構造に対応するためのモデル展開や参照構造付与などの手法には妥当性があると考えられる。一方で、5 章の実験ではリファインメントを持つ部品への対応や、異なる実装を持つ複数の部品の検索・選択といった従来手法まで全て網羅しているとは言えないため、さらに部品や要求モデルを新たに作成して実験を進める必要がある。

7 おわりに

本稿では MSSS 手法によるソフトウェア合成を行うシステムを構築し、実際に部品を用いて要求をみたすソフトウェアを合成する例を示した。今後はさらに実験を進めることで MSSS 手法の検証や改善を行う。

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] 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.
[3] 田中涼介, 織田健. B Method におけるモジュール構造に対応したソフトウェア合成システムの構築. 情報処理学会第 87 回全国大会講演論文集. 2025, vol.1, p.285-286.