Vol.2012-SE-175 No.8
2012/3/15

ツールを使用した形式仕様作成の事例研究

井上　心太[†1]　　大森　洋一[†2]　　荒木　啓二郎[†2]

形式手法を用いて仕様を厳密に記述することで，上流工程からソフトウェアの品質を向上できる．しかし，自然言語記述から形式仕様への変換は，その記述体系の相違から容易ではない．本研究では，自然言語記述から形式仕様への変換を補助するドメイン辞書管理ツールを用いた形式仕様への変換手順を提案する．本稿では，組み込みシステムの仕様書に対して実施した開発の事例研究の内容を示す．開発の結果，モデルの抽象度の設定の問題やクラス構成の抽出方法の問題が発見された．開発で発生した問題から自然言語記述の形式仕様への変換の手順の改良，およびツールの使用法を考察する．

The Case Study of Creation of the Formal Model Which Uses a Tool

Shinta Inoue,[†1] Yoichi Omori[†2] and Keijiro Araki[†2]

The software quality can be improved from the early stage of software development by strictly description of the specification using Formal Method. However, the conversion to formal model from natural language description is not easy from a difference of the description system. In this research, the conversion procedure to the formal model is proposed, using the domain dictionary management tool with which the conversion to formal model from natural language description is assisted. This paper shows the details of the case study of development which is carried out to the specification of an embedded system are shown. The problem of a setup of the degree of abstraction of the formal model and the problem of the extraction method of class composition have been discovered as a result of development. Improvement of the procedure of the conversion to the formal model of natural language description and the directions of a tool are considered from the problem which occurred in development.

1. はじめに

ソフトウェアの品質の向上への要求が高まっている．この問題を解決する方法として形式手法による仕様の検証が注目されている[1)]．形式手法を用いて仕様を厳密に記述し，仕様の曖昧さや不備を取り除くことでソフトウェアの品質を向上できる．しかし，自然言語で記述された仕様書の形式仕様への変換は，その記述体系の相違から容易ではない．

自然言語記述の形式化の試みは過去の研究[2)3)]で行われている．これらの研究では自然言語の意味を一意に定めることで仕様を厳密に記述することに成功している．しかし，限定されたドメインを対象としているため一般性に欠けている部分がある．この問題の解決のため文献 4) では形式仕様作成をサポートするドメイン辞書管理ツール（以下本研究では辞書ツールと呼ぶ）を使った形式仕様作成の手順を提案している．しかし，この作成手順は仕様書から形式仕様への変換を完全に支援しているとは言い難く，実際の開発に適用できるかは不明である．また，ツールの評価も十分ではない．

本研究では,「話題沸騰ポット（GOMA-1015 型）要求仕様書第 7 版」[5)]（以後本研究ではポット仕様書と呼ぶ）を対象に，先行研究[4)]で示されている手順（以後本研究ではツール使用手順と呼ぶ）に沿って辞書ツールを使用した開発の事例研究を行う．ポット仕様書は SESSAME[6)] が提供する電子ポットを題材にした組込みシステム分析・設計のための要求仕様書である．また，開発で発生した問題を分析し，これを解決するためのツール使用手順の改良を提案する．さらに，辞書ツールを使用して発見したツール自体の不備やその評価についても考察する．

事例研究における形式仕様の記述には VDM++を使用する．VDM++は形式手法の一種である VDM（Vienna Development Method）の形式仕様記述言語である[7)]．VDM はモデル規範型と呼ばれ，モデルの状態を記述していくことに重点を置いている．また，VDM を適用した開発の成果[8)]も報告されており，その有用性が実証されている．

本研究の以降の章の構成は以下のとおりである．まず，第 2 章では，ドメイン辞書管理ツールの概要について述べる．第 3 章では，事例研究の内容を，辞書ツールを用いた形式仕

†1 九州大学工学部電気情報工学科
Department of Electorical Engineering and Computer Science, School of Engineering, Kyushu University
†2 九州大学大学院システム情報科学研究院
Faculty of Information Science and Electrical Engineering, Kyushu University

Vol.2012-SE-175 No.8
2012/3/15

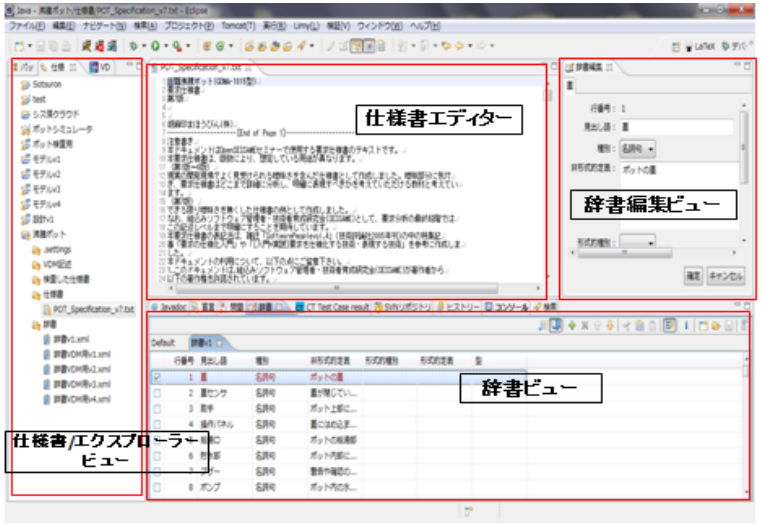

図 1 ツールの外観
Fig. 1 Appearance of a tool

様作成から実装までを順を追って述べる．第 4 章では，開発で発生した問題の分析を行う．第 5 章では，ツール使用手順の改良についての考察とツールの評価について述べる．最後に第 6 章では，本研究のまとめを述べる．

2. ドメイン辞書管理ツール

辞書ツールは，仕様記述から形式仕様への変換をサポートするツールである．辞書ツールは，使用すれば自動で形式仕様を作成するというものではなく，形式仕様の作成を補助するツールとなっている．辞書ツールは大きく分けて，以下の 3 つの機能を持っている．

- 辞書管理機能
- 単語の色づけ機能
- モデル出力機能

以降の節では，これらの機能の詳細について述べる．

2.1 辞書管理機能

辞書ツールでは自然言語記述のファイルを読み込み，その記述内の単語をマウスなどで指定し抽出ボタンを押すことで辞書に登録することができる．自然言語記述内の任意の単語を指定することができ，辞書の見出し語として追加される．

登録された見出し語に対して，種別，自然言語による非形式的定義，形式的種別，形式仕様記述言語による形式的定義を追加できる．種別は名詞句，動詞句，状態から選択し，形式的種別ではクラス，変数，定数，関数，手続きから選択する．

また，複数の辞書を作成することができ，辞書にはそれぞれ，対象領域，利用組織，入力言語，出力言語を記述するフィールドが存在する．対象領域はその辞書の専門領域を示すもので，辞書の種類を識別するための情報である．また，組織による識別は利用組織で行う．入力言語と出力言語はそれぞれ見出し語の記述言語，形式モデルの記述言語である．さらに，辞書には優先度をつけることことができ，単語の色づけを優先すべき辞書を指定できる．

2.2 単語の色づけ機能

検査ボタンを押すことで，辞書に登録された見出し語に対して自然言記述中の同一の文字列すべてに色付けを行う．色づけは最初に登場した単語とそれ以降に登場した単語で色を分けることができる．また，辞書の種別の欄で設定されている名詞句，動詞句，状態といった区分によって異なる色を指定できる．複数の辞書から色づけを行うことも可能である．その際に複数の見出し語が自然言語記述内で一致した場合，

(1) 優先度の高い辞書
(2) 同一辞書内の最長一致
(3) 同一辞書内の先頭に近い単語

の順の優先度で色づけされる．

2.3 モデル出力機能

形式的定義と形式的種別の欄を記入済みの辞書がある場合，モデル出力ボタンを押すことで，指定されたフォルダに VDM モデルを出力する．この際，VDM モデルはクラス毎に別のファイルとして出力される．また，ツールは形式的定義に記述された内容を整形して出力するだけである．ツールでは VDM++の構文の検査は行われないため，VDMTools などのツールで別途構文チェックを行う必要がある．

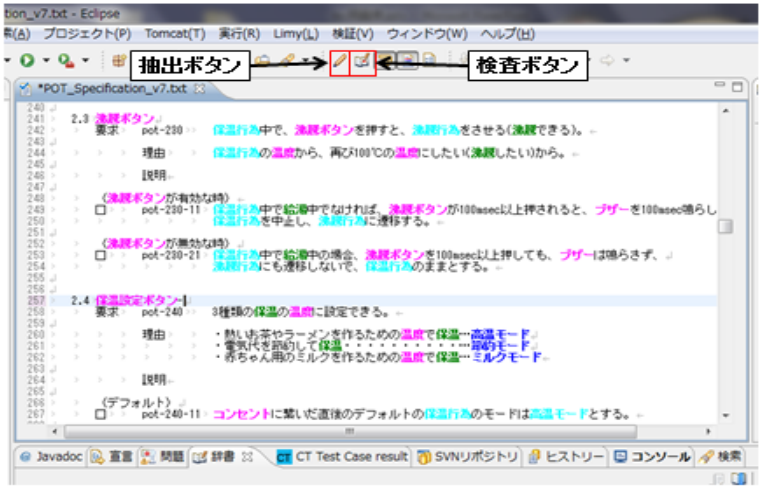

図 2 単語の色づけ機能
Fig. 2 The coloring function of the words

3. 事例研究

本章では，ポット仕様書を対象に辞書ツールを使用した事例研究の内容を示す．以降では，事例研究での開発を辞書ツールを使用した VDM モデル作成，VDM モデルの検証，実装の 3 工程に分けて説明する．

3.1 辞書ツールを使用した VDM モデル作成

ツール使用手順に沿ってポットの仕様書を VDM モデルへ変換した．手順の詳細と実施した内容を以下に示す．

(1) 仕様書中の単語を辞書に登録
ポット仕様書を辞書ツールに読み込み，仕様書中のシステムの構成要素となる単語を辞書に登録していった．登録時にはこまめに単語の色づけ機能を使用して，単語の抜けや漏れがないか確認を行った．これにより「センサ」と「センサー」といった細かい表記の揺れを発見することができた．この段階では，75 個の単語を辞書の見出し語として登録した．

(2) 名詞句の整理
登録した辞書中の見出し語から名詞句となるものを抽出し，整理した．ここでは，75 個の見出し語から 41 個の名詞句を抽出した．

(3) 類義語・多義語の整理
抽出した名詞句の中から類義語・多義語をまとめ，類義語は必要に応じて削除した．例として，「水温」と「温度」という 2 つの見出し語を登録していたがこれらは同一の意味だと判断し，「温度」を辞書から削除した．

(4) クラス分け（グループ化）
整理された名詞句の中からクラスの候補を洗い出した．クラスの候補以外の名詞句に関しては，どのクラスに属するかを吟味してグループ分けを行った．この時点ではクラスの候補は「話題沸騰ポット」，「操作パネル」，「貯水部」の 3 つの見出し語とした．

(5) 状態と動詞句に分類される単語の整理
名詞句とみなさなかった見出し語を状態と動詞句に分類した．ここで，状態 25 個，動詞句 9 個となった．これにより辞書に登録した 75 個の見出し語を名詞句 41 個，状態 25 個，動詞句 9 個に分類したことになる．

(6) 登録した単語を具体的な変数として定義
グループ分けされた名詞句や状態の中で変数とするものには，データ型や具体的な定義を与えた．例として，「モード」という見出し語は以下のように VDM++で記述した．

```
private 「モード」: <高温モード>|<ミルクモード>|<節約モード>
```

ここで＜と＞で囲まれた型のことは VDM++では引用型と呼び，任意の名前の値を設定することができる型である[9]．上記の例では，「モード」は引用型の集合として定義されており，このような型のことを列挙型と呼ぶ．本研究で作成した VDM モデルでは変数のほとんどはこの例のような形式で定義した．これは具体的な値の決定を後の段階に繰り延べることができるためである．また，「private」という記述があるが，これはアクセス修飾詞と呼ばれる修飾詞である．アクセス修飾詞は，可視性とスコープの規則であり，モデルのどの地点で特定の要素が使えるかどうかを決定する[7]．VDM++には以下の 3 種類のアクセス修飾詞が存在する．

- public
  モデル中のすべてのクラスのすべてのオブジェクトから見ることができる．

- protected
  そのクラスあるいはサブクラスの中からのみ参照することができる．
- private
  定義されたクラスの中でのみ参照することができる．

この段階では，アクセス修飾詞に関してはあまり深く考えずに記述し，後の手順である VDM モデルの詳細化の段階で決定していった．

(7) VDM++の記述の書き込み
以上の手順によってある程度まとまったクラス分けがなされているので，これをもとにして辞書の形式的定義の欄に VDM++の記述を書き込んだ．この際，VDM モデルに必要無い見出し語を削除し，新たに必要になった見出し語を追加した．また，操作の記述に関しては具体的な定義は与えず，以下のように VDM++で記述した．

```
水量異常の判断 : () ==> bool
水量異常の判断() ==
       is not yet specified;
```

操作の本体の「is not yet specified」の部分は，操作の具体的な処理内容を与えずに操作を定義する表現方法である． これにより，操作の具体的な定義を後の段階へと繰り延べることが可能となる．

(8) VDM モデルへ変換
辞書ツールの形式モデル出力の機能を使い辞書から VDM モデルに変換した．これにより,「話題沸騰ポット」,「操作パネル」,「貯水部」の 3 つのクラスファイルが作成された．

(9) 変換された VDM モデルを段階的に詳細化
出力された VDM モデルを詳細化した．特に操作に関しては具体的な処理内容を与えていないため，この段階で具体的な記述を行った．この詳細化については 3.1.1 節で詳しく述べる．

3.1.1 VDM モデルの詳細化

VDM モデルを段階的に詳細化する際には第三者に VDM モデルをレビューしてもらいながら記述を行った．レビューは合計 4 回行われ，レビュー毎に指摘された部分の加筆修正を行った．このレビューにより VDM モデルの問題点だけでなく，ツール使用手順の問題点も浮かびあがってきた．以降では各レビューで指摘された内容について述べる．また，

表 1 モデルのクラス構成の変遷
Table 1 Changes of the class composition of a model

時期	1 回目のレビュー後	2 回目のレビュー後	3 回目のレビュー後	4 回目のレビュー後
モデルのクラス構成	「話題沸騰ポット」	「話題沸騰ポット」	「話題沸騰ポット」	「話題沸騰ポット」
	「センサ」	「センサ」	「センサ」	「センサ」
	「タイマ」	「タイマ」	「タイマ」	「タイマ」
	「ボタン」	「ボタン」	「ボタン」	「ボタン」
	「ハードウェア」	「インジケータ」	「インジケータ」	「インジケータ」
	–	「ランプ」	「ランプ」	「ランプ」
	–	–	–	「パネル」

レビュー毎の VDM モデルクラス構成を**表 1** に示す．

1 回目のレビュー VDM モデルのクラス構成が適切でない点，モデルの抽象度が低い点を指摘された．これらの指摘をもとにクラス分けからやり直し，操作に関しては具体的な処理を記述することは避け，出来るだけ抽象的な記述を心がけてモデルの修正を行った．レビュー前は 3 つだったクラスを表 1 に示すように 5 つのクラスへ再構成した．また，モデルの抽象度を高くするために，必要最低限の構成要素のみをモデルに記述し，モデルを簡単化した．

2 回目のレビュー システムに必要な機能が省かれている点，ポットのシステムを表現するうえで不足している操作がある点を指摘された．前者はユーザにポットの状態を知らせる機能（ブザーとランプ）をポットの振る舞いに影響しないと判断して省略していたことが原因であった．後者は実際にポットを動かす際に必要なコンストラクタや蓋の開け閉めなどのユーザのポットに対する操作の抜けが原因であった．これらの指摘をもとに，省かれていたブザー，ランプ，インジケータに関する記述および不足していた操作の記述を追加した．この際，ブザーは変数，ランプとインジケータはクラスとして定義した．

3 回目のレビュー ポットの状態遷移に関係する事前条件・事後条件を確認すべきであることを指摘された．モデルを確認した結果，必要な事前条件・事後条件が抜けていて，正しく状態が遷移しないことが分かった．また,「インジケータ」クラスを「ランプ」クラスのサブクラスと定義していたが，現段階ではインジケータとランプは別のクラスとして定義した方がよいという指摘も受けた．これは，インジケータは本質的には水位を外部に知らせる機能を持つ部品であり，その他のランプとは機能が異なるためである．これらの指摘を受けて，事前条件・事後条件の追加と「インジケータ」クラスの再定義を行った．また，コンセントと蓋は変数として定義していたが，どちらも on と off という状態を持つ

一種のボタンとして機能するという共通点があるためボタンクラスのインスタンスとした．ここではクラス構成の変更は行わなかった．

4 回目のレビュー　仕様中に記述されていないエラー後の復帰処理についての記述の抜けを指摘された．今回は仕様書にエラーの復帰処理の記載がないため仕様書の内容に従いモデルには記述しないものとした．また，ポットの振る舞いに直接関係ないため省略していた温度とモードの表示機能を「パネル」クラスとして追加した．

以上のレビューの指摘からモデルに必要な見出し語の取捨選択や見出し語をどう VDM モデルに対応付けるかがツール使用手順では不足していることのではないかと推測する．

3.2 VDM モデルの検証

前節で作成された VDM モデルをポットの状態遷移に注目して検証を行った．具体的には，ポットの状態遷移表を作成し，状態が仕様書の意図どおりに正しく遷移するか，遷移条件が正しく記述されているかの 2 点について検証した．実際に作成した状態遷移表を**表 2** に示す．表の縦の状態は遷移後の状態を表しており，横の状態は遷移前の状態を表している．表中の項目には次の状態に移るための操作が記述されている．斜線の部分については，その状態間の遷移が存在しないことを表す．本研究で作成した VDM モデルでは加熱，カルキ抜き，保温，給湯の 4 つの操作がポットの状態を遷移させる．例えば,「加熱中かつエラーなし」状態のときに「加熱 ()」操作を呼び出すと「アイドル」状態に遷移する可能性があることを示している.「加熱 ()」操作は,

```
public 加熱 : () ==> ()
加熱() ==
(状態 := <加熱中>;
 --温度制御行為をしない場合の処理
 if (蓋センサ.参照() = <OFF> or 水量異常の判断() = <異常>)
    then(ヒータ := <停止中>;
         状態 := <アイドル>;
         沸騰ランプ.消灯();
         保温ランプ.消灯()  )
        ・・・・・
 )
```

と定義されていて，蓋センサが OFF のときあるいは，水量が異常であるときにアイドル状態に遷移することが分かる．これが,「加熱中かつエラーなし」状態から「アイドル」状態への遷移の条件となる．この条件を仕様書の記述と照らし合わせて検証を行った．このような検証を表 2 中のすべての遷移について行い，遷移の条件を確認した．検証の結果から温度エラーが発生するときに蓋が閉じていなければならないという条件の書き漏らしを発見した．これを受けて，温度エラーの処理の部分の修正を行った．

表 2　検証用のポットの状態遷移表
Table 2　The state transition table of the pot for verification

遷移後の状態 ＼ 遷移前の状態	アイドル	加熱中かつ エラーなし	カルキ抜き中 かつエラーなし	保温中かつ 給湯停止中かつ エラーなし	保温中かつ 給湯中かつ エラーなし
アイドル		加熱 ()	カルキ抜き ()	保温 ()	給湯 ()
加熱中かつ エラーなし	加熱 ()			加熱 ()	
加熱中かつ 高温エラー	加熱 ()	加熱 ()		加熱 ()	
加熱中かつ 温度上がらずエラー	加熱 ()	加熱 ()		加熱 ()	
カルキ抜き中かつ エラーなし		カルキ抜き ()			
カルキ抜き中かつ 高温エラー		カルキ抜き ()	カルキ抜き ()		
カルキ抜き中かつ 温度上がらずエラー		カルキ抜き ()	カルキ抜き ()		
保温中かつ 給湯停止中かつ エラーなし			保温 ()		給湯 ()
保温中かつ 給湯停止中かつ 高温エラー			保温 ()	給湯 ()	給湯 ()
保温中かつ 給湯停止中かつ 温度下がらずエラー			保温 ()	給湯 ()	給湯 ()
保温中かつ 給湯停止中かつ 温度上がらずエラー			保温 ()	給湯 ()	給湯 ()
保温中かつ 給湯中かつ エラーなし				給湯 ()	

3.3 実　　装

前節で検証された VDM モデルをもとに実装を行った．実装では設計とプログラミング言語 Java によるコーディングの 2 つの作業を行った．

設計では引き続き VDM++を使用して記述した．今回作成したモデルの大部分は設計として使用することができた．ここではモデル中で定義された操作の具体的な処理内容に関して記述を追加した．また，必要に応じてクラスの追加も行った．

コーディングでは Java の Swing を用いて実装を行った．Swing とは，Java の GUI（Graphical User Interface）の開発のための API（Application Program Interface）セットのことである．ポット仕様書は組み込みシステムの仕様書であり，そのままソフトウェアとして実装はできない．したがって，ユーザがボタン操作でポットの振る舞いを確認できるポットのシミュレータとして実装を行った．実装には，VDM モデルの検証で状態遷移の条件を明確にしていたため，ポットの状態遷移に関する問題は発生しなかった．実装時に発生した問題の多くは，Swing 特有の処理やスレッド処理が原因のものであった．具体的には，並行動作しているスレッドが同一のコンポーネントの状態を変更してしまい意図していない表示になってしまう問題などが発生した．

4. 開発で発生した問題

本章では，第 3 章の辞書ツールを用いた開発の各工程で発生した問題の分析を行う．

4.1 モデル作成

この工程では，3.1.1 節で述べた VDM モデルのレビューで指摘されたモデルの欠陥が問題として挙げられる．以降ではそれぞれの問題についての分析を行う．

モデルの抽象度が低い　ツール使用手順ではモデルの抽象度に関しては特には触れていない．したがって，モデル作成者によって抽象度に差が出ると考える．この原因はレビュー前のモデルではプログラムのように処理を具体的に記述していたことであると考える．また，インスタンス変数の不変条件や操作の事前条件と事後条件を仕様書からうまく抽出できていないことも原因として挙げられる．

クラス構成が適切ではない　ツール使用手順にはクラスの候補にする見出し語の選定基準がないことにより発生したと考えられる．また，見出し語のグループ化においても明確な指針がなく，モデル作成者に依存する．これは，ツールで抽出した見出し語とモデルの構成要素との間の対応付けが不明であることが原因であると考える．

図 3　シミュレータの外観
Fig. 3　Appearance of a simulator

不足している記述がある　ポットの振る舞いに影響しないと判断してポットの機能を省略したために発生した問題である．省略した機能に関係する見出し語は辞書から削除した．これにより，新たに辞書に見出し語を登録しなければならなくなった．辞書ツールのモデル出力機能を使用するためには，辞書内のすべての見出し語に対して形式的定義を記述しなければならない．これにより，モデルで使用しないと判断した単語を辞書から削除しなければならなかったことが原因である．

4.2 モデルの検証

モデルの検証ではポットの状態遷移の検証を行った．ここでは，ポットの状態遷移の条件をモデルから抽出する部分に問題があった．遷移の条件は操作の事前条件や操作の本体に記述しており，満たすべき条件の抽出が困難であった．これは，VDM++では多様な表現が可能で，遷移の条件を操作の事前条件と本体のどちらでも記述可能であることが原因であると考える．

4.3 実　　装

実装では仕様書中に記載されていないが，実装するうえで必要になる項目の扱いが問題となった．実装では，Swing の API を使用したため，設計時には想定していなかったクラスを定義しなければならなかった．これは実装言語に依存する部分とモデルとの対応付けを設計で明確にしなかったことが原因であった．

4.4 問題の分析

以上の問題を整理する．

まずモデル作成においては抽象度の設定が必要であること，見出し語の取捨選択とグループ分けの明確な指針がないことが問題となっている．前者に関してはモデル化をする際に重要になる問題で，モデルを作成する前にモデルの検証内容により抽象度を設定するべきある．後者については辞書の見出し語の整理とモデルとの対応付けがツール使用手順では不足している可能性があることを示唆している．

次にモデルの検証では，モデルに必要な制約を系統的に抽出できていないことが問題となっていた．VDM では抽象化のために事前条件や事後条件を用いて制約を記述することが重要なポイントとなるため，仕様書から制約を系統的に抽出する必要がある．

最後に実装では，VDM モデルと設計の間の対応付けを明確にしていない点が問題となっていた．VDM モデルは仕様書の検証のために作成される．一方，設計は実装のために作成されるソフトウェアシステムの構想である．両者の間には共通する部分もあるが，作成する目的が違うため相違点が多い．よって，VDM モデルから設計を行う際にはこの相違点を考慮する必要があると考える．

5. 考　　察

本章では，4 章で分析した問題をもとにツール使用手順の改良を提案する．また，ツールの不備の指摘とツールの評価についても述べる．

5.1 辞書ツールを使用した VDM モデルへの変換手順の改良

4 章で述べたように文献 4) で示されているツール使用手順では決定できない項目が存在する．この項目には，抽象度の設定，見出し語の取捨選択およびグループ分けの 2 つが該当する．これらの項目は以下に示す方法により解決できると考える．

5.1.1 抽象度の設定

モデルの抽象度は検証する内容に左右されるため一般化することが難しい．しかし，始めは抽象度を高く設定し，モデルの詳細化の段階で必要に応じて抽象度を低くしていくというプロセスが良いと考える．これにより，詳細化の段階で検証したい内容に合わせてモデルを記述していくことができると考える．これは 3.1 節で実際に行った内容である．

注意する点としては，モデルの記述では具体的な値を使用しないことと，関数や操作は事前条件と事後条件に着目して記述することである．また，関数と操作では陰定義として記述するという方法も考えられる．これは具体的な関数や操作の中身については考慮せずに満たすべき条件に注目してモデルを記述でき，抽象化しやすいからである．

5.1.2 見出し語の取捨選択およびグループ分け

辞書に登録した見出し語をモデル内で使用するかどうかの判断については難しいと考える．現時点で必要のない見出し語であっても後になって必要であることが判明したり，構成を変更する場合には必要な見出し語も変わってしまうからである．辞書ツールでは形式的定義を記述していない見出し語が辞書に存在するとモデル出力が行えない．よって，モデル中で必要がなくなった見出し語に関しては辞書から削除しなくてはならない．

以上の問題を解決するために，単語の抽出を行う辞書と形式的定義を記述する辞書を別に作成する方法を提案する．まず，仕様書中の単語を辞書に一通り登録し，その辞書をもとにして形式的定義を記述するための辞書を作成する．新たに作成した辞書に形式的定義を書きこみ，必要がないと判断した見出し語は削除する．もし削除した見出し語が後で必要になった場合，単語を抽出した辞書から見出し語を再登録すればよい．また，モデル作成後にクラス構成などを大幅に変更する必要がある場合は，単語の抽出を行った辞書をもとに新たな辞書を作成することで対応する．さらに，クラスを再構成する場合，単語を抽出した辞書に見出し語のグループ分けの結果を反映させることで，グループ分けをスムーズに行うこともできる．これにより，見出し語のグループ分けに失敗してもグループ分けを繰り返す過程で洗練できると考える．

以上により，ツール使用手順では決定できない項目を決定することができると考える．また，文献 4) では詳しく述べられていなかった辞書の管理に関する手順を加えることを提案した．

5.2 辞書ツールの評価

辞書ツールを用いた開発を通して発見した辞書ツールの不備とツールの評価について述べる．

5.3 ツールを使用した開発で発見したツールの不備

ツールを用いた VDM モデルの作成においてツールの不備を発見した．その内容を以下に述べる．

データ型を辞書の種別の欄で指定できない VDM++ではデータ型を定義することができる．辞書中の見出し語からデータ型を作成することが 3.1 節のモデル作成中にあったが，辞書の種別の欄にはデータ型に関する項目は存在しない．よって，本研究ではクラス定義の中に型の定義を書きこむという形で対処した．辞書は見出し語とモデルの間の関連を示すものであるため，データ型の項目がないのは不自然であると考える．

インスタンス変数の不変条件の記述位置の問題 VDM++の文法ではインスタンス変数の不変条件の定義は，インスタンス変数定義の末尾にまとめて記述しなければならない．辞書でこの不変条件を記述する際，クラス内で一番最後に登録されている変数の形式的定義の欄にまとめて不変条件を記述しなければならない．よって，不変条件が記述された部分の形式的定義が煩雑になってしまう．また，不変条件を追加する際も記述する場所が分かりにくくなり不便である．

5.4 ツールの評価

ツールを使用することで，ドキュメントの単語の表記の揺れのチェックは十分に行えることが本研究の開発から確認できた．しかし，辞書管理，モデル出力機能に関しては前節で述べた不備が存在し，ユーザが配慮しなければならない点が多くあると感じる．よって，VDM++の記述をサポートする機能の必要性を感じた．例えば，形式的種別の欄へのデータ型の項目の追加や，インスタンス変数の不変条件を記述できる欄の追加などが考えられる．

6. おわりに

本研究では辞書ツールを使用した形式仕様の作成から実装までを実際に行うことで，辞書ツールを使用した開発の事例研究を行った．また，開発で発生した問題から辞書ツールを使用した仕様書から形式仕様への変換の手順の改良を提案した．

これによって，辞書ツールを使用した仕様書から形式仕様への変換の手順を適用することで発生する問題点とこの手順を適用する際に考慮すべき点が発見されたと考える．

謝辞 ツール開発に協力して下さった吉村康晴氏をはじめとする九州ビジネス株式会社のみなさまに感謝申し上げます．本研究では，NPO 法人組み込みソフトウェア管理者・技術者育成研究会の配布する「話題沸騰ポット（GOMA-1015 型）要求仕様書第 7 版」を利用しています．また，本研究の一部は，科学研究費補助金（21300009），基盤研究（B）「ソフトウェア開発の現場で使えるフォーマルメソッドに関する研究」の助成を受けたものである．

参 考 文 献

1) 中島 震：ソフトウェア工学の道具としての形式手法，*NII Technical Report* (2010).
2) 西田富士夫，高松 忍，谷 忠明：要求仕様における日本語表現と形式表現間の相互変換，情報処理学会論文誌，Vol.29, No.4, pp.368–377 (1988).
3) 小林吉純：日本語による電話サービス仕様記述における表現の多様性と意味の同一性の認識，電子情報通信学会論文誌，No.8, pp.1035–1048 (1999).
4) 大森洋一，荒木啓二郎：自然言語による仕様記述の形式モデルへの変換を利用した品実向上に向けて，情報処理学会論文誌，Vol.3, No.5, pp.18–28 (2010).
5) SESSAME: 話題沸騰ポット要求仕様書，SESSAME（オンライン），入手先 ⟨http://www.sessame.jp/workinggroup/WorkingGroup2/POT_Specification.htm⟩（参照 2012-1-30）.
6) SESSAME: http://www.sessame.jp/.
7) ジョン・フィッツジェラルド，ピーター・ゴルム・ラーセン，ポール・マッカージー，ニコ・プラット，マーセル・バーホフ：VDM++によるオブジェクト指向システムの高品質設計と検証，翔泳社 (2010). 酒匂 寛 訳.
8) 栗田太郎：携帯電話組込み用モバイル Felica IC チップ開発における形式仕様記述手法の適用，情報処理， Vol.49, No.5, pp.506–513 (2008).
9) CSK: VDM++言語マニュアル，CSK（オンライン），入手先⟨http://www.vdmtools.jp/uploads/manuals/langmanvice_a4J.pdf⟩（参照 2012-1-30）.