演習コースⅡ 形式手法と仕様記述

「形式手法と仕様記述」 実施報告

Report on “Formal Methods and Specification Description”

主査　　：栗田　太郎（フェリカネットワークス株式会社）
副主査　：石川　冬樹（国立情報学研究所）
研究員　：
伊藤　淳（NEC ソリューションイノベータ株式会社）
蛸島　昭之（株式会社デンソー）
田邉　昭（株式会社野村総合研究所）
内藤　聡（テックスエンジソリューションズ株式会社）
宮本　陽子（株式会社メタテクノ）

報告概要
仕様をはじめとした開発上流の成果物における品質確保のため，国内産業界でも，形式手法への注目が高まっている．しかし一般の開発者にはまだその実際は馴染みがない．加えて技術を学ぶことができたとしても，プロジェクトの性質など状況に応じた適切な活用方法を定めることは難しい．本演習コースにおいては，参加者はまず，形式手法の一つ VDM の学習を通し，形式手法における原則を実感した．その上で，各自の要望，悩み，興味に応じ，学んだ手法の活用のための研究提案や，様々な手法の学習や活用模索を行った．

Abstract For quality assurance of early deliverables in development, especially specifications, formal methods have recently attracted attentions of the Japanese industry. However, they are still “unknown” for most ordinary developers. Moreover, even if technology is obtained, difficulties lie in deciding proper usages according to contexts such as project characteristics. In this exercise course, participants first studied VDM, one of formal methods, to catch principles in formal methods. Each of the participants then worked for research proposals to leverage the methods, or studied specific methods and their applications, according to their requirements, concerns and interests.

1. 仕様記述における様々な問題

開発上流工程の成果物における品質確保は，効果の大きさ，効率の高さの双方の観点から非常に重要とされる．逆に，上流工程に起因する不具合が，発見，解消されないまま後の工程に引き継がれると，その修正コストは上流での修正コストの何十倍にもなる[Feiler09]．特に仕様は，「何を作ろうとしているのか（何を作ったのか）」を記述，維持するものであり，複数の組織・チーム・人をまたがって設計・実装，テスト，運用・保守等の拠り所となるものである．このため，仕様の品質確保は非常に重要なのである．

一方，仕様の記述においては，様々な種類の難しさが混在し，それらに起因する多種多様な問題が生じる．その代表的な例として，下記が挙げられる．

【厳密さ・可読性に関する問題】　発注者や設計・実装担当者，将来の仕様担当者など，想定する読み手が，容易に理解し，また一意に解釈できるように，指針を定めて記述や確認を行っていない．このため，誤解が発生し手戻りの原因となり，後の保守や派生開発も非常に困難になる．

【整合性・正当性に関する問題】　並行動作するオブジェクトの状態遷移や，データの読

み書きなどによるモジュール間の依存関係が非常に複雑であるが，それらが整理，検証されていない．このため，特定のケースでのみ影響が現れる不具合が残る．一方，実装後のプログラム・システムは，様々な側面を含み，実行環境や状態が複雑すぎて，再現や理解，修正ができない．

【合目的性・必要十分性に関する問題】　仕様全体やその中の各項目が定める「ゴール」と，その上位の「ゴール」（正当性や妥当性の基準）が結びついておらず，仕様が必要であり十分であることが検証されていない．このため，仕様項目の漏れが発生しやすくなるとともに，後に適切な変更内容を定めることが難しくなる．

設計や実装，テスト，ソフトウェア保守・進化（派生開発）などにおけるトラブルの根幹には，仕様に関するこういった問題があることが多い．

2.　形式手法

仕様に関する問題の解決には様々なアプローチがあるが，国内産業界では近年「形式手法」が注目されている（詳細は，[MRI11, DSF11, IPA10] などにまとめられている）．
それでは，「形式手法とはどういうものなのか」という問いを投げかけると，人によって次のように様々な答えが返ってくるだろう．

- 数理論理学に基づいた手法のことである．
- プログラミング言語のように，文法や意味論が定まった言語で記述するので，記述の表す内容・意味が一意に定まり，定義の不整合や不足もツールで確認できる．
- テストとは異なり，バグがないことを証明できる．
- スレッドの切り替えのタイミングや通信の成否などにより分岐する，大量で複雑な状態遷移の可能性を，網羅的に自動検査してくれる（モデル検査）．
- 様々な例を自動生成したり，シミュレーションしたりして，ユーザや開発者の確信度を高めたり，漏れに対する気づきを促したり，テストケースを生成したりすることができる（モデル発見，仕様アニメーション）．
- 原子力や航空などのミッションクリティカルな領域において，コストをかけて高信頼性を確保するためのアプローチである．
- 様々な領域において，品質確保のためにかけるコストを上流に移すこと（フロントローディング）により，手戻りによるコスト増大を防止し，全体のコストを下げたり，実装・テスト段階に負荷が集中することを避けたりするためのアプローチである．

これらの答えそれぞれは，正しいとも言えるし間違っているとも言える．というのも，形式手法という言葉は総称にすぎないからである．具体的な手法やツールは多種多様である（VDM, B, Event-B, Alloy, SPIN, UPPAAL など）．さらに，それらの手法・ツールを直接利用しなくとも，その裏にある原則，考え方を，日本語仕様の記述規約や DSL（Domain Specific Language）の文法，レビュー方法やレビュー基準などに埋め込むことにより，手軽に活用できることも多い．

結局のところ，個々の手法，ツール，その裏側にある原則，考え方に対し，それが対象とする問題と効果，限界を十分に理解，実感した上で，組織やプロジェクト，開発対象の性質に応じた活用方法を定める必要がある．加えて，形式手法の利用によってある問題に対処できそうだとしても，学習，移行や運用の課題もあれば，他にも考えなければならない問題が多々あるため，総合的な施策の整理，構築が求められる．

3.　演習コースⅡにおける取り組み

本演習コースでは，前述の背景を踏まえ，下記 2 つの観点からの取り組みを行う場を参加者全員で作り上げることを目指している．

（1）　形式手法の考え方も踏まえての，仕様記述における問題解決の模索と議論
（2）　特定の手法・ツールの学習と活用に向けた検討

まず準備段階として，5～6 月においては，最も手軽な手法として国内での知名度が高い VDM を中心として講義，演習を行った． VDM は，構造化プログラミングやオブジェクト指向に基づいてのモデリングや，解釈実行を通したテストなど，一般の開発者にとって馴染みのある記述・検証方法を用いる手法である．また日本語でのツール利用や情報取得が行いやすく，国内における適用事例もよく知られている［VDMTools，Kurita10］．なお，VDM およびその他の手法について，8 月に追加のセミナーも 2 回行った．

このように，VDM を一例として形式手法全体に関する理解と実感を得つつ，7 月の合宿以降は，各参加者の要望，興味，悩みに応じてグループ分けと取り組みテーマの決定を行い，実際の取り組みを行った．上記（１）については，研究の取り組みとして，課題を分析し，達成目標とアプローチを定め取り組んだ.（２）については，各自で対象とする手法・ツールを定め学習や取り組みを行った．

各自の取り組み概要を下記に示す．

（１）に関する取り組み：Ａチーム（田邉・内藤・宮本）

(1.1) テーマ

情報伝達文書における自然言語・図・VDM の特徴．読み手と書き手の立場での考察と議論．

(1.2) 背景

我々の所属する開発現場では，要求や仕様を伝える文書や，前任の担当者から後任に向けて，業務の流れを文章と図で説明するような，いわゆる引継ぎ資料で下記のようなコミュニケーションエラー（認識のずれ）が発生し，課題となっている．

- 要求仕様書では，自然言語と図を用いて表現しているが，記述不足・用語のゆれ・意味のわからない用語があるため，書き手と読み手に認識のずれが生じている．複数の解釈をゆるす曖昧な文や図も，誤解の要因となっている．
- 引継ぎ資料では，前任者の暗黙知が充分に記述されていないことがあり，後任がなかなか理解できないことがある．
- 引継ぎ資料に，曖昧な用語やわかりにくい記述があった場合，後任である読み手は業務知識（ドメイン知識）を持たないため，何が正しいのかを判断できず，読み間違いや書き間違いに気づかないまま引き継いでいることがある．

これらの情報伝達時のコミュニケーションエラーは，我々の開発現場のみで起きていることではなく，様々な開発現場に共通する課題である．自然言語と図で書かれた要求仕様書の曖昧さは，システム開発における重要な問題であるため，これまでにいくつもの手法が提案されている．とくに，厳密な仕様記述に関しては，国内では形式手法の一種である，VDM が有効な手段として，既に活用されている[Kurita・Araki10][Sahara11]．これまでの本演習コース II においても，VDM の利活用は，継続して取り組んでいるテーマのひとつである[Hada・Wada11] [Miyamoto11] [Miyamoto・Kusakabe12]．

一方，引継ぎ資料の認識ずれや曖昧さの問題に対しては，我々の知るかぎりでは，解決に向けた策が見つけられていない．前任者である書き手の記述不足や，曖昧さに起因する読み手の理解不足が発生していたとしても，後任者がある程度成長しないと，記述不足や理解不足に気づくことができないという検知の難しさがある．この問題に対して，厳密な仕様記述に有効性のある形式手法 VDM を用いることで改善できないかと考え,本年度の研究課題とした．

なお，VDM（Vienna Development Method）は，IBM のウィーン研究所にて開発された，モデル規範型に属する形式手法の一種である．言語としては，ISO で標準化されている VDM-SL（VDM Specification Language)と，それに対して主にオブジェクト指向の拡張を行った VDM++などがある．

(1.3) 研究課題

本研究では，まず，ある業務の流れの一部を題材とし，その題材を自然言語・図・形式仕様記述言語 VDM の 3 種で記述することで，形式仕様記述言語 VDM の適用方法を学びながら，手段によって，読み手の理解や書き手の心理にどのような違いがみられるか，を考察・議論することにした．具体的には，下記の 2 点である．

- 自然言語・図・形式手法 VDM，それぞれの特徴を整理すること
- 手段によって，書き手と読み手の立場でどのような違いがあるか

(1.4) 研究方法

ここでは，研究対象とした具体例と，研究の方法をのべる．

今回の記述対象とした業務内容は，ある店舗情報誌の広告営業担当者の業務で，社内システムを利用した広告原稿の作成から，顧客への提案作業を経て，社内関係者とのシステムを介した広告原稿申請するまでの流れ[図 1]である．

研究の方法は，次のような手順で行った．

業務の引継ぎのシーンを想定し，業務内容（前述）を，下記の３つの表記法で記載し，それぞれの取組みを通して，見られた差異や結果を整理する．

①自然言語と(自作の)図による表記
②業務フロー図による表記
③VDM による表記

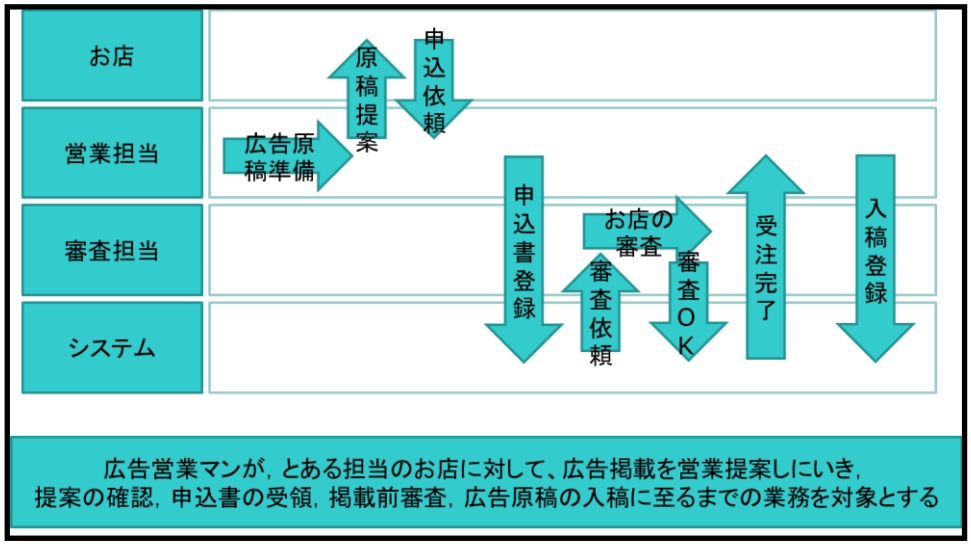


図 1 題材にした「店舗情報誌の広告営業担当者の業務」

(1.5) 研究結果

ここでは，①～③の結果として，表記内容および書き手と読み手の所感を示す．

①自然言語と(自作の)図による表記

No	誌面に広告が掲載される条件	流れ
1	広告原稿と申込書が揃ったら	広告原稿 ＋ 申込書 ⇒ 掲載
2	広告原稿と申込書が揃い、原稿を入稿完了したうえで、締切に間に合ったら	広告原稿 → 入稿完了 ＋ 申込書 → 締切 ⇒ 掲載
3	OL提案し、OLが内容承諾した広告原稿と申込書が揃い、原稿を入稿完了したうえで締切に間に合ったら	広告原稿 → 初稿提案 ↓ ↑ 内容承諾 ＋ 申込書 → 入稿完了 → 締切 ⇒ 掲載
4	OL提案し、OLが内容承諾した広告原稿と申込書が揃い、原稿を入稿完了したうえで締切に間に合ったもののみを対象に広告原稿を印刷用画像化したら	広告原稿 → 初稿提案 ↓ ↑ 内容承諾 ＋ 申込書 → 入稿完了 → 締切 → 印刷用画像化 ⇒ 掲載
5	OL提案し、OLが内容承諾した広告原稿と申込書が揃い、原稿を入稿完了したうえで 締切に間に合ったもののみを対象に広告原稿を業務ルールに沿って掲載順を決め、広告原稿を印刷用画像化したら	広告原稿 → 初稿提案 ↓ ↑ 内容承諾 ＋ 申込書 → 入稿完了 → 締切 → 掲載順決定 → 印刷用画像化 ⇒ 掲載
6	OL提案し、OLが内容承諾した広告原稿と、申込OLを業務ルールに沿って審査し、審査OK申込のみ受注完了とし、受注完了した申込書が揃い、原稿を入稿完了したうえで締切に間に合ったもののみを対象に広告原稿を業務ルールに沿って掲載順を決め、広告原稿を印刷用画像化したら	広告原稿 → 初稿提案 ↓ ↑ 内容承諾 ＋ ↑ 申込書 → 審査 → 受注完了 → 入稿完了 → 締切 → 掲載順決定 → 印刷用画像化 ⇒ 掲載
7	OL提案し、OLが内容承諾した広告原稿と、申込OLを業務ルールに沿って審査し、審査OK申込のみ受注完了とし、受注完了した申込書が揃い、原稿を入稿完了したうえで事前に必要になる用紙枚数を、原稿数と広告サイズから概算し不足しない用紙を発注しておく、締切に間に合ったもののみを対象に広告原稿を業務ルールに沿って掲載順を決め、広告原稿を印刷用画像化したものを納品された掲載用紙に印刷・製本したら	広告原稿 → 初稿提案 ↓ ↑ 内容承諾 ＋ ↑ 申込書 → 審査 → 受注完了 → 入稿完了 →原稿数把握 →掲載用紙発注 → 用紙納品 ↓ → 締切 → 掲載順決定 → 印刷用画像化 → 印刷・製本 ⇒ 掲載

【①の所感】

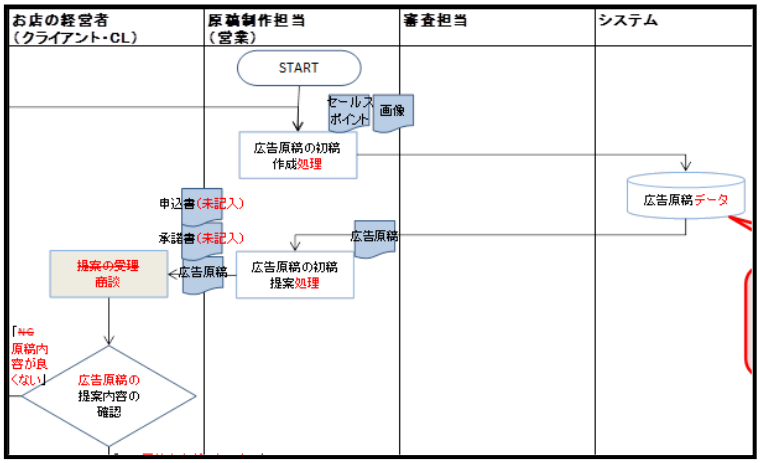
書き手：

- 記述スタイルが細かくルール化されていないので，書き手の思いつくままに自由に記述にできる．
- わかりやすさを意識すると，曖昧になり，厳密さを意識すると，わかりにくくなる．
- 書いた直後は正しく書いたつもりだったが，あとで読み返すと自分でもわからない部分があった．
- 記述内容を変更したい場合に，差分を比較しにくいので，変更箇所の管理が難しい．

読み手：

- 曖昧さ以前に，ドメイン知識がないと，ひとつひとつの用語がわからない．
- 用語の意味がわかっていても，背景や文脈がわかっていないと，短い文章であっても，用語の関係性が捉えられず，内容がわからない．
- 自作の図はどのような表記ルールとなっているかがわからないため，読み解けない．
- 記述内容が変更された場合に，差分を比較しにくいので，変更箇所がわかりにくい．
- 気になる点があっても，記述スタイルの好みの問題なのか，内容の問題なのかが判別できず，指摘しにくい．

②業務フロー図による表記



【②の所感】

書き手：

- 図で詳細情報まで記述すると，重なってしまうなど，読みづらくなる．
- 内容を整理してから書き始めても，見やすい配置か，線が重ならないか，など見やすさに気を配るので，作成に時間がかかる．
- 図の表記ルールを確認しながら記述するが，用語のゆれや記述内容に間違いがあっても自分では気づきにくい．

読み手：

- 直感的でわかりやすい．しかし，「なんとなくわかった気」になってしまうと，詳細まで理解せずに読みとばしてしまうことがある．
- 図の表記ルールに合致していない場合は指摘しやすいが，それが反映されたとしても，記述内容が妥当かどうかはわからない．
- 更新されたファイルを受け取った場合に，差分比較をしても，記述内容が変わったのか，図の配置のみが変わったのか，わかりにくい．

③VDM による表記(一部抜粋)

```
class 原稿制作担当 is subclass of CommonType
instance variables --インスタンス変数定義
  public i管理システム : 管理システム;
operations --操作定義
  --構成子
  public 原稿制作担当 : 管理システム ==> 原稿制作担当
  原稿制作担当(a管理システム) == i管理システム := a管理システム;

  public 初稿作成処理 : 顧客名 * セールスポイント * 画像 ==> ()
  初稿作成処理( a顧客名,aセールスポイント, a画像 ) ==
    i管理システム.初稿作成( a顧客名, aセールスポイント, a画像 );

  public 初稿提案処理 :( 顧客名 | 原稿NO ) ==> 管理システム`広告原稿マップ
  初稿提案処理( a検索キー ) ==
    i管理システム.初稿検索( a検索キー );

  public 確認結果の反映処理 :( 顧客名 | 原稿NO ) ==> 管理システム`広告原稿マップ
  確認結果の反映処理( a検索キー )  ==
    i管理システム.原稿状態更新( a検索キー, <内容承諾> );
end 原稿制作担当
```

```
class 管理システム is subclass of CommonType
types
  public 顧客情報マップ = inmap 顧客ID to 顧客名;
  public 広告原稿マップ = inmap 原稿NO to 広告原稿;
instance variables --インスタンス変数定義
  private i顧客情報マップ : 顧客情報マップ := {|->};
  private i広告原稿データ : 広告原稿マップ := {|->};
  private i発行済顧客ID : 顧客ID := 0;
  private i発行済原稿NO : 原稿NO := 0;
operations --操作定義
  public 初稿作成 : 顧客名 * セールスポイント * 画像 ==> ()
  初稿作成( a顧客名, aセールスポイント, a画像 ) ==
  (
    dcl x顧客ID : [顧客ID] := 顧客IDを得る(a顧客名);
    dcl x広告原稿 : 広告原稿;
    if x顧客ID = nil--顧客名が未登録の場合
    then
      x広告原稿 := new 広告原稿(i発行済原稿NO,a顧客名,
                        顧客登録(a顧客名),aセールスポイント,a画像)
    else
      x広告原稿 := new 広告原稿(i発行済原稿NO,a顧客名, x顧客ID,
                        aセールスポイント, a画像);
    i広告原稿データ := i広告原稿データ munion
                         {i発行済原稿NO |-> x広告原稿};
    i発行済原稿NO := i発行済原稿NO + 1;
  )
  post
  -- 広告原稿データの数が１増えていること
    card dom i広告原稿データ = card dom i広告原稿データ~ + 1;
:
end 管理システム
```

【③の所感】
書き手：
・ 用語のゆれなどの記述ミスは，VDM ツールにより，即座に検知できる．
・ VDM で書くためには，文法やツールの使用方法だけでなく，対象とするシステムへの深い認識が必要．(システムの責務，境界，用語の定義など)
・ 具体的なテストケースを考えることが，記述対象への理解を深める助けとなる．想定したテスト結果になるまで，VDM のテストを何度も実行し，VDM の記述を見直すことで，確信をもちつつ記述することができる．
読み手：
・ VDM はツールの使用方法や文法を学ぶことで，比較的短期間で読めるようになった．
・ テストケースを実行したり，実行結果の C0 カバレッジが確認できたりするため，記述が妥当なのか，テストで確認できていない部分がないかを具体的に把握することができる．
・ ソースコードと同じように，差分を比較できるので，変更箇所がわかりやすい．さらに，テストを回帰実行できるため，変更したことに影響をテスト結果で確認できる．

なお，VDM++で，処理と具体的なテストケースを記述し，記述内容の動作確認を行ったことにより，自然言語や図の表現では見つからなかった問題点を 19 件指摘できた(表 1)．

表 1 VDM 作成とテストケース実行を行ったことによる指摘内容の分類

分類	件数
①用語のゆれ （2種類以上の用語が同じ意味で用いられている）	4
②記述不足 （記述モレ，説明不足，暗黙知）	8
③曖昧 （定義が不明確で，複数の解釈ができる）	5
④その他 （文章そのものへの疑問など）	2

(1.6) 考察
上記の記述結果を通して，記述手法ごとの特徴を整理し，考察する(表 2)．

【自然言語・(自作の)図】
書き手が自己の認識を表現する手始めとして，とっつきやすく，扱いやすいと言える．その一方で，読み手の語彙レベルや背景・文脈の理解度が低い場合は，内容が伝わりにくい．記述スタイルが細かくルール化されていないので，書き手は自由に書きやすいが，読み手は気になる点があっても，記述スタイルの好みが違うのか，内容に問題があるのか判断がつかず，指摘しにくい．記述スタイルに自由さがあるため，変更箇所を差分比較したり，変更点を細かく管理したりするのは難しいと言える。

【業務フロー図】
書き手が記述ルールを守ることが必要である．読み手も記述ルールがわかれば，全体をイメージしながら，情報の流れを把握しやすい．全体が把握しやすい反面，詳細な情報を記述すると複雑になり，読みやすさが損なわれるため，一つの図で何もかも表現することはできない．詳細な情報を表現するには，複数の図を使いわけたり，具体例を別途あげたりするなど，工夫が必要である．表記ルールに合致していない点については指摘しやすいが，流れの妥当性については，具体例やドメイン知識がないと指摘しにくい．差分を比較しても記述内容の変更か，配置が変わっただけで内容には変更がないのか，わかりにくいと言える．

【VDM】

VDM で記述するには，文法やツールの習熟に加え，記述方針や記述範囲を定めたり，ひとつひとつの用語を整理し定義を確認したり，情報をどのような概念で扱うか設計を行ったり，と様々な観点から深く考える必要がある．そのため，不明確なドメイン知識では書きたくても書けないというジレンマに陥る．けれども，それを乗り越え書き出すことができれば，用語のゆれや未定義の用語はツールの静的チェックで即座に指摘されるため，内容の意味や関係に集中して記述することができ，問題点に気づきやすい．読み手は，記述量が多いと全体を把握しにくいと思われるが，用語の定義だけでなく，処理内容の妥当性をテストケースの実行で確認することができるため，詳細まで把握でき，テスト結果にもとづいた問題指摘もできる．プログラムソース同じように版管理や差分比較ができ，テストケースの回帰実行により，変更による影響範囲もわかりやすいと言える．

表 2 記述手法ごとの特徴

	書き手の扱いやすさ	読み手の全体把握のしやすさ	問題箇所への気づきやすさ	変更管理のしやすさ
自然言語・図	◎	△	△	△
業務フロー図	○	◎	○	△
VDM	△	△	◎	◎

(1.7) 結論

研究を通して得た結論を以下にまとめる．

- 人によって語彙力や読解力，背景や興味が異なるため，現実世界のような複雑な情報を 100%間違いなく伝達することは，困難である．
- まず伝える側(書き手)が，伝えられる側(読み手)の語彙力，業務経験，文化的背景，興味の対象などを把握するよう努める必要がある．
- 書き手は，伝えること・わかりあうことの困難さを意識したうえで，それでもできる限り伝える努力をしなくてはならない．そのためには，１つの表現手段だけに頼ることなく，読み手の理解に応じた表現形式を適宜使いわけることが求められる．
- 読み手は，認識のずれがあることを想定し，具体例をふまえて理解したことを言い換えるようにするとよい．

（２）に関する取り組み：Ｂチーム(蛸島)

(2.1) テーマ

形式手法 Alloy を用いた状態遷移モデルの安全性分析検討．

(2.2) 取り組み概観

組み込みシステム開発では状態を持つシステムを対象とすることが非常に多く，状態遷移図による設計は大変重要である．だが，状態遷移図には遷移のヌケ・モレを見つけるのが困難という課題がある．この課題は，状態遷移表を併用することで解決することができる．しかし，システムの安全分析を行う場合，状態遷移表による遷移のヌケ・モレ検査だけでは不十分である．安全分析では，車の走行中はドアロック解除が発生しないかといった検証が必要となる．そこで，軽量形式手法ツールの 1 つである Alloy Analyzer を用いて

状態遷移モデルの安全性分析が行えるか検討を行った．また，同じ状態遷移モデルから N スイッチカバレッジを満たすテストケースが生成可能かを検討した．現状では，それぞれの目的には別の状態遷移モデルを作成する必要がある．今後は，安全性分析と N スイッチカバレッジ・テストケース生成の両方が行える統一的なモデル記述方法を構築したい．(蛸島)

4. まとめと展望

ここまで述べたように，形式手法と一口に言っても多種多様な側面を扱っている．また仕様記述から，システム分析における妥当性確認，設計の検証，テストとの連動など，様々な活用の可能性がある．限られた時間において，様々な可能性を模索したり，特定のアプローチをしっかり使いこなせるようになったりすることは難しい．しかし本コースでの経験を基に，参加者が継続的に適応，進化を続けていって欲しい．

コース自身のあり方としては，「各参加者が成長した」ということだけでなく，取り組みにおける成果物を積み重ね，コース全体として成長し成果物を出していくことが重要と考えられる．いずれにしても，主査，副主査も含めメンバ全員でアプローチを議論し，楽しく進めていきたい．

（文責：石川 冬樹）

参考文献

[Feiler09] Peter H. Feiler et al (2009). System Architecture Virtual Integration: An Industrial Case Study. Technical Report CMU/SEI-2009-TR-017, Carnegie Mellon University

[MRI11] 三菱総合研究所・経済産業省 (2011). フォーマルメソッド導入ガイダンス. http://formal.mri.co.jp/

[DSF11] Dependable Software Forum (2011). 形式手法活用ガイドなど. http://www.nttdata.com/jp/ja/dsf/index.html

[IPA10] IPA (2010). 形式手法適用調査 . http://www.ipa.go.jp/sec/softwareengineering/reports/20100729.html

[VDMTools] SCSK 株式会社. VDM information web site. http://www.vdmtools.jp/

[Kurita10] 栗田 太郎 (2010). モバイル FeliCa のソフトウェア開発における品質確保のための構造と実践　抽象度の制御やコミュニケーションの活性化に向けて. 情報処理学会デジタルプラクティス Vol.1 No.3

[Kurita・Araki09] 栗田 太郎，荒木啓二郎 (2009). モデル規範型形式手法 VDM と仕様記述言語 VDM++ －高信頼性システムの開発に向けて－. 日本信頼性学会.

[Sahara11] 佐原 伸 (2011). 構造化日本語仕様書としての VDM 仕様. SEC Journal

[Hada・Wada11] 羽田 裕・和田 圭司 (2011). VDM 仕様とテスト設計による仕様の問題発見に関する評価. 日本科学技術連盟第 27 年度ソフトウェア品質管理研究会成果報告集

[Miyamoto11] 宮本 陽子 (2011). 形式仕様記述言語の利用による仕様書の改善 －USDM と形式仕様記述の考察を通して－. 日本科学技術連盟第 27 年度ソフトウェア品質管理研究会成果報告集

[Miyamoto・Kusakabe12] 宮本 陽子 (2011). VDM と USDM を組み合わせた仕様記述方法－VDM による USDM 仕様記述の改善提案－. 日本科学技術連盟第 28 年度ソフトウェア品質管理研究会成果報告集