2015 年度「形式手法と仕様記述」 実施報告

Report on “Formal Methods and Specification Description” FY2015

主査　　：栗田　太郎（ソニー株式会社）
副主査　：石川　冬樹（国立情報学研究所）
研究員　：臼井　眞一（富士フイルムソフトウエア株式会社）
　　大村　林太郎（アズビル株式会社）
　　木本　俊（日本プロセス株式会社）
　　田邉　昭（株式会社野村総合研究所）
　　宮本　陽子（株式会社メタテクノ）

研究概要
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]．また他の手法として LTSA[LTSA]などのモデル検査ツール，およびモデル発見ツールを持つ手法 Alloy[Alloy]について８月に追加のセミナーも行った．

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．

4. 各取り組みの概要

以下では 2015 年度に行った 4 つの取り組みの概要について紹介する．これらのうち 4.1 で述べる取り組みは，上記 3.における分類（１）の取り組み，残りは（２）の取り組みであった．

4.1 チェックシートの改良に関する検討（田邊）

(1.1) テーマ

チェックシートの記述内容の厳密さについて、現状の問題点と今後の改善点の考察

(1.2) 背景

QC７つ道具の１つにチェックシートがある．点検用チェックシートは，トラブルの未然防止・再発防止の道具として，ソフトウェアの業界に限らず，建築業界や航空業界、医療分野でも利用されている．弊社でも様々なチェックシートを作成し，システム開発の現場で活用し効果を挙げている．しかし，必ずしも良質なチェックシートばかりとは言えず，現場に混乱を招くチェックシートも存在している．記述内容が曖昧である，記述内容が不足している,あるいは冗長な表現で可読性が悪く,読み手に複数の解釈を与えてしまうチェックシートや，チェックすることでどのような効果が得られるのか，どのような障害の再発防止となるのか，どのようなリスクが回避・軽減されるのか等，チェックする目的が分からないチェックシートも存在している．それが原因で，現場では，本来意図するものとは異なる，ピントはずれなチェックを行ってしまい，後々開発上の手戻り，あるいは重大な障害を引き起こすといったことも実際に起きている．

(1.3) 研究課題

本研究では，システム開発の現場で実際に利用されている基本設計時のチェックシートの一部を題材として，記述の厳密さの観点から記述内容を分析することで，該当チェックシートの記述上の問題点を明らかにし，今後の改善点を考察する．

(1.4) 研究方法

研究の方法は次の手順で行った．

１）チェックシートの記述内容を分解する．

２）分解内容を整理する．

１）では，現場で使用している基本設計時のチェックシート 34 項目について，記述内容の過不足を確認する為に文節ごとに分解を行った．分解は，形式手法における「記法の型にはめる」という考え方に基づき,「誰が(WHO)」,「いつ(Who)」,「何を(What)」,「どこで(Where)」,「なぜ(why)」,「どのように(How)」,「どうする(動詞)」の５Ｗ１Ｈ＋動詞で行った．５Ｗ１Ｈは，もとは新聞記事を書く際の原則だが，ビジネスの場面では報告書・メールの作成時や，口頭で状況を説明する際にも応用されている．５Ｗ１Ｈに沿って整理し，５Ｗ１Ｈにあたる内容を相手に伝えるようにすると，情報をわかりやすく，もれなく伝達

することができるとされている．しかし，実際のチェックシートの記述内容が，５Ｗ１Ｈをどれだけ充足しているのかの調査を行ってみたところ．当てはまらない文節も多く，それらについては記述内容から，「条件」，「誰に（誰と）」，「その他修飾語」に分類した．また１つのチェック項目に２つ以上のチェック項目が含まれているものについては独立させた．その結果，チェック項目は 50 となった．

２）では，５Ｗ１Ｈ＋動詞と追加３分類の特徴を整理し，考察のための準備とした．

(1.5) 研究結果

１）チェックシートの記述を分解した結果の抜粋を下記に示す．

	2	3	4	5	5増幅
原文	商用環境・加盟店ST環境で提供できる機能を顧客と認識合わせしているか？	今回システム化する要件に関して、目的・背景を理解しているか？	新たな業務、新規データモデルを使用する場合は、その業務及びデータ特性を顧客から連携頂いているか？	既存の業務・データであっても、新たにシステムの処理を追加する場合は、業務内容を理解し、データパターンの整理ができているか？	同左
誰が(Who)					
何を(What) ※目的語	提供できる機能を	目的・背景を	業務及びデータ特性を	業務内容を	データパターンを
いつ(When)					
どこで(Where)					
なぜ(Why) ※チェックする理由					
どのように(How)					
どうする(動詞)	認識合わせすること	理解すること	連携してもらうこと	理解すること	整理すること
条件			新たな業務、新規データモデルを使用する場合	既存の業務、データモデルを使用する場合　かつ　新たにシステムの処理を追加する場合	既存の業務、データモデルを使用する場合　かつ　新たにシステムの処理を追加する場合
誰に（誰と）	顧客と		顧客に		
その他修飾語	商用環境・加盟店ST環境で	今回システム化する要件の			

表 1 チェックシートの記述　分解結果（抜粋）

２）分解内容の整理した結果を下記に示す．

分解内容	存在した項目数
誰が(Who)	3
何を(What) ※目的語	49
いつ(When)	3
どこで(Where)	0
なぜ(Why)　※チェックする理由	1
どのように(How)	9
どうする(動詞)	50
条件	22
誰に（誰と）	22
その他修飾語	16

表 2 分解内容　整理結果

記述の構成は，目的語となる何を(What)＋どうする(動詞)の形式がベースとなっていた．「何を(What)」は 49 項目，「どうする(動詞)」は 50 項目全てに存在した．次いで多かったのが「条件」と「誰に（誰と)」で 22 項目，逆に少なかったのは，「どのように(How)」が 9 項目，「誰が(Who)」と「いつ(When)」が 3 項目，「なぜ(Why)」が 1 項目，「どこで(where)」はゼロ項目であった．

また，使用されている「どうする(動詞)」の種類としては 23 種類存在した．

項番	どうする(動詞)の種類	存在した項目数
1	確認すること	12
2	認識合わせすること	5
3	合意すること	4
4	把握すること（＝明確に押さえること）	3
5	記載すること	2
6	決定すること	2
7	検討すること	2
8	整理すること	2
9	定義しないこと	2
10	同意すること	2
11	理解すること	2
12	依頼すること	1
13	一致させること	1
14	計算すること	1
15	作成すること	1
16	整合性をとること	1
17	設計すること	1
18	定義すること	1
19	変更すること	1
20	明確にすること	1
21	用いること	1
22	連携してもらうこと	1
23	連携すること	1

表 3 使用されている動詞　使用順

（1.6）考察

上記の結果を通して，該当チェックシートの記述に関する問題点と改善方法について考察する．

チェック項目に「どのように(How)」が 9/50 項目存在と少なかった．方法が 1 つである場合やどのような方法でも実施しても結果が同じになる場合には問題とならないが，実施方法によって結果が異なる可能性がある場合には，「どのように(How)」で実施方法を限定すべきと思われる．また、「なぜ(Why)」についても 1/50 項目存在とほとんどの項目で記述が無かった．「なぜ(Why)」が分かることで，最適な「どのように(How)」を選択することも可能であるため，「なぜ(Why)」の記述は重要と言える．「なぜ(Why)」については，記載項目として独立させる等の工夫も必要である．

「誰が(Who)」3/50 項目，「いつ(When)」3/50 項目存在とこちらも少なかったが，シート全体で誰がいつ使用するかについて組織としての共通認識があるため，記述が無くても影響は少ないと思われる．「どこで(Where)」0/50 項目存在については，このチェックシートには記載不要と思われる．

動詞の種類については，「理解すること」，「明確にすること」など何をもって理解したことになるのか，何をもって明確になったと言えるのか，の様に深さの定義が必要な動詞が使用されていることも複数の解釈を生む要因だと思われる．また「認識合わせをすること」や「合意すること」、「同意すること」など，あえて使い分ける必要がある場合は別だが，使用する動詞を集約することも検討すべきである．

（1．7）結論

研究を通して得た結論を以下にまとめる.

- 「なぜ(Why)」は重要. 記載を促すために項目を独立させる等の工夫も必要.
- 「誰が(Who)」や「いつ(When)」など組織として共通認識がなされているものは，あえて記述する必要は無い. 記述すると冗長となり，却って分かり難くなるおそれもある.
- 動詞の使い方に気を配る. 深さの定義が必要な動詞は深さを定義するか，別な表現に置き換える.

今後、ここで得た結論をもとに，作成者，利用者と意見交換を行ってより分かり易いチェックシートが作成されるように作成手順を整理していきたい.

4.2 厳密な記述に関する検討（大村）

(2.1) テーマ

VDM により仕様記述の曖昧さを除去した記述法を学ぶ

(2.2) 取り組み背景

組み込み機器開発において機能仕様書の担当者と機能設計書の担当者が異なる場合,仕様書に書かれた内容が正しく伝わらないことは後工程において設計バグを引き起こす要因となる. テスト工程に携わる中でこの要因に起因する設計バグに出くわすことが多かった私は，バグを誘発しない仕様書の書き方を学ぶことを目的として演習コースⅡに参加した.

仕様記述言語の１つである VDM は曖昧さを許さない厳密な記述を求められる. VDM を学ぶことを通して，仕様書上で読み手と書き手の理解の不一致を誘発する「曖昧さ」とは何かを学ぶ.

(2.3) 取組内容

とある演算機能の仕様文を VDM により記述することで，２つの曖昧さを生む記述を見つけることが出来た.

１つ目は,「1 つの言葉には１つの意味を持たせる」である. １つの意味を異なる用語で表現していたり，１つの用語に異なる意味を持たせていたりしたが，VDM でこれらを変数や関数で定義するときに用語と意味の対応関係が統一されていないことに気が付き，正確に書き分けを行う事ができた.

２つ目は,「複数の解釈ができる言葉を使わない」である. 仕様の中に

・「入力 X_1～X_n に異常がある場合，…」

という記述がある. この文章は VDM にて記述すると，以下の文章のいずれかになるように強制される.

・「X_1～X_n の全てが異常である場合，…」

・「X_1～X_n のいずれかが異常である場合，…」

このどちらの意味を持たしているのかを仕様の記述者に確認を行い，誤解釈を生む恐れのあった記述を修正する事ができた.

(2.4) まとめと今後の活動

仕様を VDM により記述することは仕様書としての文章を書くことに加えての手間がかかる行為である. 工数に制限がある中で VDM をそのまま業務に取り入れることはチームや職場レベルでの改善が必要となりハードルが高くなるが，VDM の習得を通じて曖昧さを除去した文章の記述に努めることは個人レベルでの実施が可能であり，仕様書の質を確かにあげられる行為であると考える.

私が現状見出した記述は上述の 2 点のみだが，これからも VDM を学び仕様の曖昧さを生む記述の観点をさらに増やしていきたいと考えている.

4.3 モデル検査の活用に関する検討（１）（臼井）

(3.1) テーマ

SPIN による不具合早期検出の試行.

(3.2) 取り組み背景

社内の各種プロジェクトの不具合低減を行う必要があり方法を模索していた際に，設計段階でソフトウェアをシミュレーションで動かすことが可能な形式手法であれば，早期に不具合を検出できるのではないかとの考えに至った．目に見えないソフトウェアはその性質の一つとして，動かすことで不具合を検出しやすいという性質があると考えており，経験的に設計工程で見つけられなかった不具合も，テスト工程になると見つけやすくなる．また優れたレビュアに，頭の中でかなり先までソフトウェアを動かしながら指摘をしていると聞いたことがある．形式手法でツールの力を借りて設計段階でソフトウェアを動かすことにより，レビューで指摘できていないような不具合を早期に検出し，不具合低減を行うことを目的として演習Ⅱコースに参加した.

(3.3) 取り組み概要

一言に形式手法と言ってもいろいろあるので，演習コースでは代表的なものとして VDM++, Alloy, LTSA, SPIN について概要を学んだ.時間的制約を考慮して絞り込みを行い，モデル検査の一つである SPIN を詳細に学び使用することとした.

解決に難航している不具合は，並行するプログラムが個々に動作することでシステム全体として想定していない状態に陥ることによるものが多かったが，その種の問題には SPIN が適しているとのアドバイスをいただき,手元にあった不具合事例を SPIN でモデル化してみた．C 言語の知識があったせいか考えていたよりも簡単にモデル化が行えたこと，作成したモデルを使用して不具合が簡単に検出できたことから，SPIN を選択した.

モデル検査とは，形式手法の一つであり，振る舞いの仕様をモデル化し，モデル検査ツールを用いて検査をすることで，システムのとりえる全ての状態を網羅的に探索することで，与えられた性質が成立するかどうか検証する．並行動作するプログラム間の振る舞いなどについて，上流工程に適用することで，システムの不具合を早期に発見できる．SPIN とは，AT&T Bell 研究所で開発されたモデル検査ツールのひとつ．Promela という C 言語に似た言語でシステムの振る舞いを記述し，並列動作をモデル化することができる.

SPIN についてある程度使えるレベルまで習得した後，設計工程が完了しているいくつかのプロジェクトの設計書を対象に適用実験を実施した．その結果，あるプロジェクトにおいて条件によっては，状態矛盾を引き起こし以後の継続処理が行えないという不具合を 1 件検出した．検出した不具合は，テストでは簡単に出せないタイミングの不具合で，それを設計書段階で検出することができた．また，今までの「こういうところがあやしい」という指摘が,「こういう条件で発生する」と明確になり，指摘を受けた側にもわかりやすいという効果もあった．更に，一度モデルを作成すれば比較的簡単にモデルを変更したシミュレーションを行うことができるため，修正確認を簡単に行え，SPIN の活用は有効であると判断した.

(3.4) 考察と今後の課題

効果が上がったケースもあったが,課題も見えてきた.モデルを作成しやすい設計書(状態遷移図等)が既にあれば,少しの労力でモデル検査を実施できる反面,それらがないと,設計書作成から実施する必要があり，工数がかかる．また，工数をかけて設計図を作成してモデル検査を実施するより，レビューの観点を工夫することにより工数の少ないレビューで発見できるケースもあると感じた工数と見合う効果が見込めないことがある.

今後，モデル検査の特性を踏まえた上で，リスクベースで洗い出しを行う等，効果的な

適用箇所の特定方法の具体化，費用対効果を考えた適用判断の基準の検討を行い，モデル検査を有効に活用していきたい．（臼井）

4.4　モデル検査の活用に関する検討（２）（木本）

(4.1) テーマ

形式手法に触れる．　～SPIN によるモデル検査の実践～

(4.2) 取り組み背景

私の所属する開発現場では,複数の装置で構成される並行システムの設計,製造を行っている.装置構成は,ユーザが直接操作する端末,アプリケーションサーバー,制御対象の装置などである.

一般的に並行システムの課題として,装置ごとに状態が存在することによる組み合わせ爆発.各装置の処理タイミングのズレ,応答の有無などが要因となり,想定しきれないケースが不具合を生み出すといったことが挙げられる.

私の担当している機能においても同様に,状態数が多い.割り込み処理やタイムアウトへの対応などが原因で処理が複雑化しており,不具合を誘発しやすい状態になっている.

今後,当該処理を修正するにあたり,本来想定していなければならないケースどれだけあるか,要求仕様としてどうあるべきかを明確にする必要があると考え,形式手法の中でも,起きうる状態変化を網羅的に検査することが可能な SPIN に注目し,有効に活用する方法を得ることを目的とした.

(4.3) 取り組み概要

SPIN は,検査対象のモデルを Promela という言語で作成し,作成したモデルの状態を網羅的に探索することができるモデル検査ツールである.

① 妥当性の検証

はじめに,検証したいシステム用件をすべて満たしたモデルを作成するのではなく,各装置の状態は一意であると仮定し,該当となる処理のシーケンスのみに着目した最小モデルを作成し,そのモデルの妥当性を検証した.

最小モデルとして想定した動作は,ユーザがある端末から操作をすると,サーバとある装置へ制御指示を送信し,装置での処理が終了すると,端末とサーバへ応答を返すという内容だった.

モデルを検証した結果,各装置が応答を返さないケースがあることが結果として得られた.Promela にてモデルを作成する際には,制御指示送信に対する応答の有無も想定したうえでモデルを作成しなければならない認識がなかったため,想定した動きをするモデルになっていなかった.

② 改善モデルの検証

検査対象としている処理では,制御指示に対して応答が帰ってこないことを考慮する必要がないことから,改善モデルとして制御指示を出すと必ず応答するモデルを作成し,改良したモデルが想定した動作となることを確認した.

(4.4) 取り組みにより得た知見

今回の取り組みで,検査モデルを作成するにあたり,通常の処理タイミングでは発生することのない認識していなかったタイミングでの割り込みや,状態の組み合わせを考えなければならない状況に陥ったことから,機能理解を深めるきっかけとすることができた.これは作成したモデルが,検査したい内容を十分に満たす必要があることから,必然的に対象の

処理について十分に記述することを強制させられたためである.状態数と処理の組み合わせの数が複雑な機能では,意識できないケースが発生することがありえるため,設計フェーズで仕様の検証,深度化をするための手段としては有用なものであると感じた.

結果として,今回の取り組みでは Promela に慣れておらず,モデル化を行う作業に時間を有してしまったため,検査対象とした実際のシステムの処理モデルを作成し,検査するには至らなかった.今後の課題としては,言語記述へ慣れるため,より多くの事例へ適応を実践していきたいと考える.

5. 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．

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

[LTSA] LTSA - Labelled Transition System Analyser, http://www.doc.ic.ac.uk/ltsa/

[Alloy] Alloy, http://alloy.mit.edu/alloy/