（鉄道総研月例発表会講演要旨）

段階的詳細化に基づく鉄道信号へのフォーマルメソッド適用法

信号通信技術研究部　信号
副主任研究員　寺田　夏樹

１．はじめに

鉄道信号は安全性が要求される分野である。電子化された信号システムのハードウェアの安全性技術に関してはある程度確立されたものと考えられている。一方，ソフトウェアはシステムの中の比重も規模も年々大きくなっており，高い安全性を確保しつつ，効率よく開発できる手法が望まれている。

我々は以前からフォーマルメソッド（formal methods，形式的手法）によるソフトウェアの高信頼化手法の研究を行ってきた。今回実時間システムについてフォーマルメソッドの適用を試みた所，複雑な制御システムを記述しようとする場合には，単純に形式的仕様記述言語で記述しようとしても仕様の理解が難しいことが判明してきた。そこでしばしばフォーマルメソッドで出てくる段階的詳細化（stepwise refinement）の技法に着目した。これはモデルをいきなり書くのではなく，段階を追って仕様を実装に近づけていく作業である。段階的な記述を行うことにより，仕様の理解がしやすくなる。各詳細化段階が前の段階の条件を満たすことを証明することによってその作業の正当性も得ることができる。

段階的詳細化手法に基づき複雑な制御システムを記述する試みを行い，その有効性を確認した。本報告ではその適用例とともに詳細化手法の有効性について述べる。

２．フォーマルメソッドとは

フォーマルメソッドには様々な定義があるが，数学的論理学的体系を持った仕様記述言語を用いて形式的（数学的）に仕様を記述し，その仕様に対して形式的に検証を行うことによりシステムの品質を上げるための技術の総称である。図１はその開発の１例である。形式的に仕様を記述することにより，仕様のあいまいさを排除できる。このことにより品質向上につながる。また形式的仕様記述はコンピュータにも理解できる形であり，自動テストや自動証明などといった様々な形でのコンピュータの支援により仕様の検証を効率よく行うことができる。仕様の段階での検証を十分に行うことにより信頼性やソフトウェアの生産性をあげるのはフォーマルメソッドの狙いである。

フォーマルメソッドの中にも様々な手法がある。手法ごとの得手不得手があるため，必要に応じて手法を選択し，場合によっては組み合わせて使うのがよい。我々はこれまでは VDM と呼ばれる手法を使用してきた。この手法は VDM-SL あるいは VDM++と呼ばれる言語で仕様を記述し，それをテストなどの検証を通じて，最終的にシステムの高信頼化を図ろうとする手法である。仕様の記述を明確

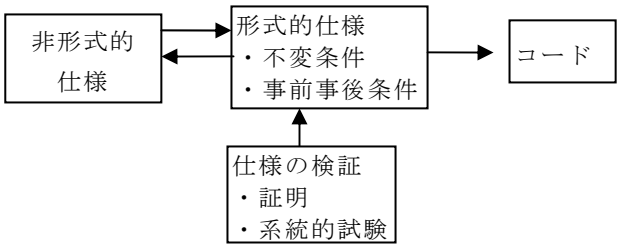

図１　フォーマルメソッドによるシステム開発

にし，実行して確認をするということに主眼をおいており，導入がしやすい手法である。

３．段階的詳細化と B-Method

VDM 等の手法を用いて仕様を記述することは品質向上に有効である。しかし仕様が単に仕様記述言語で書かれればよいのではなく，いかに理解しやすい仕様を作成するかを考える必要がある。具体的なシステムの実装となると抽象レベルでは考えていなかった様々な条件を考慮しなければならない。考慮すべき要素がたくさん存在する時にそれらを一度に記述しようとしても,概して理解しにくい仕様ができやすい。従って最初は簡単に記述し，次第に拡張していき，複雑にしていく手法が考えられる。これが段階的詳細化（stepwise refinement）という考え方である。しかし VDM の開発ツールではそのような開発スタイルはサポートしていないため別のツールの使用を考えた。フォーマルメソッドの１つである B-Method と呼ばれる手法はこの詳細化に則ったものであるため，その適用の検討を行った。

B-Method は VDM と同様のモデル化指向手法に分類されるが，その中身は段階的詳細化と証明という構成であり，VDM とは異なる面が多い（図２）。現在ではフランスの企業が開発ツールを提供しているが，証明を行う機能がその重要な位置を占めており，途中の段階の仕様でテストを行うということは難しい。

B-Method では最初にシステムを抽象機械（abstract machine）と呼ばれる抽象的な制約を持った仕様として記述する（詳細については４章で述べる）。それを段階的に詳細化する。それぞれの段階の仕様はそれ自身が矛盾のないことを証明によって示す必要があるが，この時，詳細化した仕様が詳細化する前の仕様を満たすことも証明する必要がある。最後の段階は具体化仕様(Implementation)と呼ばれプログラムに近い形（B0 と呼ぶ）に記述する。これは C や ADA といった言語に翻訳可能である。ここで各ステップの詳細化では分かりやすい範囲の違いだけを考えることにする。従ってシステムが大規模になれば詳細化の段数は増えるが，何が詳細化によって変わったのかをはっきりさせることができる。

この B-Method を用いてハードウェアの形式的記述を試みることとした。対象としたのは ATC（Automatic Train Control）システムである。この例について紹介する。

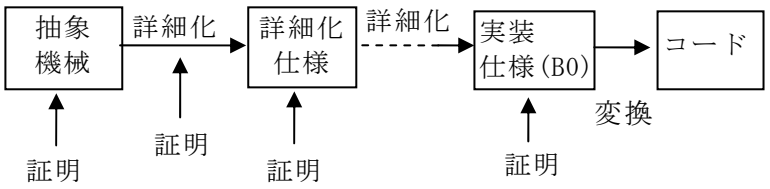

図２　B-Method による開発

４．例題－ATC ブレーキパターンの計算

従来の ATC では，列車の車上に許容速度を指示して制御していたが，最近導入されてきているデジタル ATC 等のシステムでは，車上に停止目標までの距離等を与え，それを基に車上装置が安全に停止できる速度を算出して制御している。車両の位置と許容速度の関係をブレーキ曲線（ブレーキパターン）と呼んでいるが，これを算出するプログラムをこの詳細化によって作成してみる。なお，ここでは先行列車に衝突しないために停止位置までに停止できる速度を示す停止パターンを扱うものとする。

ある停止目標までの距離が与えられた時に，一定の減速度で減速して安全に停止できる速度を計算して出力する機能を考える。このように最も抽象的なレベルの仕様を抽象機械と呼ぶが，関数 distance(sp)を速度 sp からブレーキをかけながら停止するまでに走行す

る距離を出力する関数とすると，この抽象機械は次のように書くことができる。なお、/*と*/で囲まれた部分はコメントである。

```
SPEED = {0,..,255} ∧   /* 速度 */
ta ∈ ℕ∧ lo ∈ ℕ∧ brakespeed ∈ SPEED ∧
/* ta は停止位置，lo は現在位置，brakespeed は 許容速度である */
（ta ≦ lo ⇒ brakespeed  = 0）∧
/* 停止位置が現在位置より手前であれば許容速度は 0 である．  */
(ta > lo ⇒  distance(brakespeed) ≦ta - lo)
/* 停止すべき位置が現在位置より先であれば,許容速度から停止するまでの走行距離は停止すべき位置と現在位置の差に等しいか小さい */
```

これを満たす領域を図３で図示すると①～③となる。 これに対して，詳細化仕様として停止位置の手前に余裕距離を考えた仕様を記述する。式は省略するが,図３では②,③の領域となり，①～③に含まれる。

さらに次の詳細化ではブレーキをかけ始めてから完全に効くまでの空走距離を考える。その結果は図３では③の領域となる。

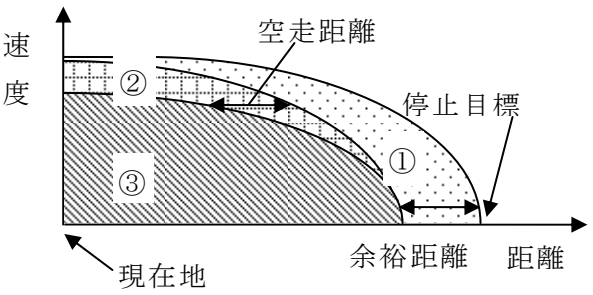

図３　ブレーキ曲線の詳細化

さらなる詳細化では伝送距離等を考慮に入れ，さらに実装上の制約（例えば距離を 16bit の整数で考える，など）を盛り込んでいく。これらの詳細化において自分自身が矛盾のないことを証明するとともに，前の段階の条件を満たしているかどうかを証明していく。

最終的にブレーキパターンに許容される領域は一定の領域になるが，そのなかでなるべく大きな速度になるように領域の中の最大値をとるように仕様を定める。最後にはその計算アルゴリズムを記述する。

この例での詳細化のステップは 12 段になった。整合性を示すために証明する必要がある条件を証明責務と呼ぶが，各段階の証明責務の数を表１にまとめた。概ね 100～300 程度の証明責務が生成されるが，そのうち大部分は自明な証明責務で，実際の証明作業が必要とされたのは１割程度である。それらのうち自動で証明された（表の自動証明の欄）のが半分程度であった。従って本当に手作業で証明しなければならないもの（手動証明の欄）は全体からすれば非常

表１　証明責務の数

モジュール	全体	自明	自動証明	手動証明
抽象機械	142	135	2	5
詳細化1	132	117	7	8
詳細化2	129	124	3	2
詳細化3	160	144	4	12
詳細化4	244	223	8	13
詳細化5	223	182	12	29
詳細化6	320	288	15	17
詳細化7	154	131	12	11
詳細化8	155	143	8	4
詳細化9	229	166	2	61
詳細化10	70	56	3	11
詳細化11	55	47	4	4
実装	348	111	90	147
計	2361	1867	170	324

に少ないため，それに集中すればよいということがわかる。また，最後の詳細化（実装）だけは対話的に証明した証明責務の数が他と比べて多いことがわかる。

５．VDM を用いた段階的詳細化

B-Method は矛盾がなく仕様に忠実なコードを出力するという点で非常に強力なツールである。しかし一方で記述の柔軟性に乏しく，広く適用されるには難しい手法である。従ってもう少し実際に適用しやすいように VDM を取り入れた方法についても検討した。この場合の段階的詳細化については，前の段階の条件を後ろの段階が満たすか満たさないかまでを必ずしも厳密に考える必要はないこととする。これはあくまでも仕様の書きやすさからの提言であり，もし安全上重要な部分であれば，B-Method 等を使用して整合性を証明していくことが必要であると考えられる。こうした検討を実際のモデル化を通じて行った結果，我々が提案する VDM を中心とした信号システムのモデル化手法は以下の通りである。

a) システムを分析し，必要な条件や基本的な要求を洗い出す
b) 最初の段階を VDM にてモデル化する。この際なるべく「何をしてほしいのか」を記載する。モデル化しつつ，そのモデルを検証する。
c) a）の中から必要な条件をピックアップしながら，さらに詳しいモデルを VDM で記述し，検証する。次第に「どうやって実現するか」を記述していくことになる。その際に a）の条件にチェックを入れて，どの条件を考慮に入れたかを把握できるようにする。
d) 仕様がある程度具体化した段階で安全性上，証明が必要な部分を抽出し，その部分については B-Method 等を使用して証明されたコードを生成する。
e) その他の部分については，VDM で詳細な仕様まで記述する。VDM のコードジェネレータあるいは手作業でコード化する。

現時点の VDM の開発ツールでは自動証明機能が提供されておらず、証明すべき条件が出力されるのみである。従って VDM で証明を行う場合に手作業となる。将来的に証明機能が提供されれば d)において証明してコード化するという部分が大きな比重を占めることも考えられる。VDM 仕様に対する証明ツールが提供される可能性があるが，商用に耐えるレベルに達するには相当の時間がかかると思われる。しかしながらそれ以外の部分については比較的導入が容易と思われる。この提案した手法の有効性について今後検討していく予定である。

６．まとめ

プログラムの信頼性を向上する技術として段階的詳細化について紹介し，これを ATC のブレーキパターン計算の例に適用してみた。段階的詳細化を行うことにより，仕様が複雑となっても各段階の違いはわずかであるため，理解がしやすい。そしてその結果得られた曲線は複雑な制約を満たしたものとなった。適用してみると，ブレーキパターンの計算には段階的詳細化という考え方を取りやすいことが判明した。

今後その他の信号システムについても同様の段階的詳細化の手法の適用を検討したいと考えている。この手法が同じようにきれいに適用できるかどうかについては分からない部分も多い。しかし，様々な信号システムに適用できることが判明すれば，その適用の経験を元に鉄道信号へのフォーマルメソッドの適用の指針へとつなげられる可能性があると考えている。そのことにより信号システムの信頼性向上につながることを期待している。