ソフトウェア工学　83－11
（１９９２．２．７）

VDM'91 シンポジューム報告

峰尾欽二
日本ユニシス（株）　研究開発部

1991年10月にオランダでVDM'91シンポジュームが開催された．会議のテーマであるFormal Software Development Methodsが示すように，４回目を迎えたこのシンポジュームは，欧州における形式的ソフトウェア開発法をすべて網羅する方向に踏み出した会議であり，同時に形式的開発法が産業界に移植されつつあることを示す会議でもあった．従来，日本ではこのシンポジュームはほとんど知られておらず，今回の会議に出席して，日本に於ける形式的開発法の研究と実践に役立つ議論が多いことを発見したのでこのシンポジュームを報告する．

The Report of the VDM'91 Symposium

Kinji Mineo
Nihon Unisys Ltd.,　Research and Development

The VDM'91 Symposium was held in the Netherlands in October, 1991. This 4th symposium, though "Formal Software Development Methods" of the theme of this conference suggests, decided to discuss about all formal methods applying in Europe, and also showed many formal methods to has been applying in the industrial area of Europe. Since so far in Japan this symposium has been not well known and in this conference I found many useful discussions in the research and the practice of formal methods in Japan, I will report the discussions in this symposium.

VDM'91 シンポジュームに出席する機会があったので，その報告を行う．VDM は Vienna Development Method の略称で，ソフトウェアの形式的開発法のひとつとして知られている．本論で詳しく触れるが，その名称とは異なりこのシンポジュームは欧州で活躍している数多くの形式的開発法を対象にしている．ところが，日本ではこのシンポジュームの実体がほとんど知られておらず，現に過去の会議に日本人の参加者はほとんどいないようである．筆者は，欧州において形式的開発法がソフトウェア開発現場にどの程度定着しているか，その現状を知ることを目的として会議に参加した．報告も同様の視点から行う．

1 会議の背景

当研究会の周辺でも形式的方法に関する議論がないわけではない．形式的仕様記述言語 LOTOS を基礎にした研究はいくつかある（たとえば，[1]）．方法論を形式化しようとする試みもある [2]．数少ないが，形式的方法を対象とした議論もある [3]．ここ数年米国でも形式的方法が一部に注目され，米国を主力とした学会で教宣が行われている．IEEE がその 3 誌 Computer, Software, Transaction on Software Engineering（1990 年 9 月号）に特集号を組み，VDM,Z などを紹介している．米国経由で欧州生まれの形式的方法が紹介されているわけである（[4],[5] もその例）．

VDM シンポジュームは VDM ヨーロッパが組織し，EC 委員会が後援して一年半毎に開催し，今回で 4 回目になる．その歴史は以下のとおりである．

第 1 回　1987.5　ブリュッセル
第 2 回　1988.9　ダブリン
第 3 回　1990.4　キール（ドイツ）

第 4 回はオランダのアムステルダムから南西 40 キロメートル離れた Noordwijkerhout という田舎町で 1991 年 10 月 21 日 (月)-25 日 (金) に開かれた．会議のテーマは，

第 1 回　A Formal Method at Work
第 2 回　The Way Ahead
第 3 回　VDM and Z
第 4 回　Formal Software Development Methods

である．

テーマが示すとおり，第 1，2 回は VDM を対象にしていたが，第 3 回では Z も含め，欧州における代表的な形式的方法 2 つをその対象に広げた（従来からある Z user meeting はそのまま継続しているようである）．第 4 回ではさらに対象を広げ，欧州で活躍しているその他の方法もシンポジュームに招待している．

これは第 5 回以降では全欧州の（米国もその対象になるかもしれない）形式的方法を対象とするように組織的再編を行う方向に動いているためであり，その意味で今回の会議は歴史的に過渡をなすものである．さらに今回の会議は，実務への適用に特別の関心を示し，プロジェクト報告のセッションと開発ツールの実演ならびにセッションを設けている．その意味でも過渡的な性格を持っている．

実務への適用に特に関心を持って参加した筆者には，論文集からは得られないものを得ることができた．報告は，既に発行されている論文集 [6],[7] からは判読しにくいものに重点をおく．論文集は 2 分冊になっている．これは，プロジェクト報告やツールセッションの項が増えただけでなく，新たに対象にした方法をチュートリアルセッションのなかで紹介しているからである．その意味でこのチュートリアルは形式的方法の初学者を対象にしているのとは異なっている．

勢力分布を知る参考のため参加者の国別分布を示す．

イギリス	36	アメリカ	9
ドイツ	19	カナダ	5
フランス	17	ノルウェー	4
オランダ	16	ポーランド	3
デンマーク	10	ベルギー	3
アイルランド	2	スウェーデン	1
イタリー	2	イスラエル	1
日本	1		

参加登録者数は 129 名であるが，前回までは 70-80 名くらいとのことであるので，今回大幅に参加者数が増えたことになる．それは，実務側の人数が増えたからであろう．

2 会議の構成

会議の最初の2日間はチュートリアルセッションで初級チュートリアルと上級チュートリアルが並行して走り，全部で8セッションから成り立っている．後半の3日間が本会議であり，論文セッションと並行して1日目にツールプレゼンテーションが，2日目にプロジェクト報告のセッションが同時開催された．

2.1 チュートリアル

論文集があるので簡単な羅列程度に紹介したい．興味のある方は文献 [7] を参照されたい．初級チュートリアルは以下の4つのセッションからなる．

1) John Guttag (MIT) の Larch

米国からの招待であり，最終日の招待講演への布石の意味を持っている．ちなみに，VDM'90 では京都大学の中島玲二教授らが招待講演に招かれ様相論理プログラミングの講演を行っている（これが唯一の日本人参加者か?）．個別仕様間のインターフェイスを記述する Larch Interface Language を中心にした話である．ソフトウェアの再利用に役立つことを強調している．

2) J.C.P.Woodcock (Oxford Univ.) の Refinement Calculus

基本的には，C.Morgan, "Programming from Specification", Prentice Hall International(1990) の紹介である．開発プロセスには Z 記法を使っている．

3) Micheal Mac an Airchinnigh (Trinity Coll.) の Irish School of the VDM

アイルランド派 VDM の紹介である．

4) Chris George (CRI) の The RAIZE Specification Language

RAIZE (Rigorous Approach to Industrial Software Engineering) の初心者向け紹介である．

上級チュートリアルも同様に4つのセッションから成り立っている．

1) Ole-Johan Dahl (Oslo Univ.) の Formal Development with ABEL

2) Bernd Krieg-Bruckner (Bremen Univ.) の The PROSPECTRA Methodology and System

PROSPECTRA (PROgram development by SPECification and TRAnsformation) も RAIZE と同じくエスプリ計画の中で開発中の形式的開発法である．

3) Ib Holm Sorensen (BP Research) の The B-Method

抽象機械に基づいたもので他の方法と少し趣を異にする．大々的に展示されていたツール B-tool とともに商用的に活用されているらしい．

4) Donald I. Good (Computational Logic, Inc) の Mathematical Methods for Digital Systems Development (The Stack)

2.2 招待講演

招待講演は本会議3日間の各日の冒頭に行われた．

1) Michael Jackson (コンサルタント),
"Description Is Our Business"

2) Robin Milner (Edinbergh Univ.),
"Concurrent Process as Objects"

3) John Guttag (MIT),
"The Larch Approach to Specification"

M. Jackson の話は，対象世界のモデル化の重要性を強調するものである．形式的方法を適用しさえすればうまくゆくということではなくて，ドメインを正確に把握することがいかに大切であるかを，比喩をまじえた，例の人を引きつける調子でしゃべった．ジャクソン法 (JAP/JSD) の版権を会社ごと売却した後でどんなことを話すか，筆者は興味を持って拝聴したのであるが，話しの内容は，"Software Methodology"（日本電子工業振興協会主催，マイコン技術フォーラム，1990年2月）の枠を越えるものではない．

R. Milner の話は，CCS や CSP に代表される従来の通信プロセスを代数モデルとして新しく発展させたものの紹介である．CCS のプロセスはλ算による関数として表現される．このプロセスにデータを包含した形で表現し，その表現方式をπ算と命名する．プロセスはオブジェクトとなり，オブジェクト・パラダイムによる並行通信プロセスの記述を提唱する．ほとんどの聴衆にとって新しい話しであったらしい．説明は，従来の CCS を記述したトランスペアレンシの上に新しい記述のトランスペアレンシを重ねる形で進めたから記述は CCS を拡張したものになっているはずである．(本文執筆中に偶然去年9月東北大学で開催された TACS'91 にπ算の紹介論文があり，既に R. Milner の論文があることを発見した．)

J. Guttag の話しは，Larch の紹介であり，他の形式的方法といかに違うかに力点をおいたものである．筆者には，Larch そのものが理解できていないため他の方法との比較は釈然としなかった．Larch は VDM や Z のように適用実績はないけれども，方法的にはより優れている，という主張を聞いても，招待されていて公然とそんな主張をする神経には度肝を抜かれることはあっても，VDM や Z と比較することに無理があるような気がした．後でも述べるが，Larch は以降の VDM シンポジュームで議論の対象とする方法群に加えられている．

2.3 論文セッション

筆者は論文セッションのかなりの部分に出席していないので，全体の傾向をつかめていない．筆者なりに興味を持ったものをひとつだけ紹介する．

AT&T ベル研究所の Pamela Zave が Michael Jackson と共同で交換システムの仕様を Z で記述した実験を報告している．ジャクソン法といういわば半形式的方法の教宣家たちが形式的方法に接近していることに感心もし，心強くも感じた．発表論文数は 29 件である．

2.4 プロジェクト報告セッション

筆者の関心は，形式的方法が欧州のソフトウェア開発現場でどのように受け入れられているか，その実状を知ることにあったから，期待して聞いた．8 編の報告があったが，個別プロジェクトの報告とは異なり，統合プロジェクトである LaCoS プロジェクトの報告が立派であり，われわれの興味に合致すると思えたので，このプロジェクト報告に焦点を当てて報告する．

LaCoS (Large scale Correct Systems using formal methods) プロジェクトはエスプリ計画の一環をなし，RAISE (Rigorous Approach to Industrial Software Engineering) を基本として，形式的ソフトウェア開発法を産業界に定着させるための実験である．

プロジェクトは消費者 (consumer) グループと生産者 (producer) グループに分かれ，消費者は，列車の運行制御，船のエンジン制御，通信網開発支援ツール，サーバー開発，画像処理，OA(事務計算)，船の運行制御，衛星システムの 8 つの分野からなり，生産者は，デンマークの CRI，イギリスの BNR，デンマークの SYPRO 3 社からなる．消費者は RAISE を適用する部隊であり，生産者は消費者を補佐し，RAISE を発展させることを任務にしている．

報告は，生産者側のコンサルタントグループによってなされた．LaCoS プロジェクトは，1995 年まで継続するプロジェクトであり，まだ初年度の経過にあり，まだ粗い仕様化段階での経験しか持ち合わせていないが，貴重な経験を蓄積している．われわれに参考になりそうな経験を以下に列挙する．

1) 方法を習得する困難さについて

ソフトウェア開発現場にいる実務家や現場に直結している研究者は，数学に基づいた方法は実務家にはとてもむずかしくて現場に適用するのは極めて困難である，という先入観を持っている傾向があるが，そうではなくて実際に習得可能であることを実地に立証している．習得し易い方法と必要な条件をあげている．始めから言語の意味を完全にさせようとすると無理があり，部分的系統的に行うことがよい．言語の意味を参照できる適切なマニュアルが必要である．ちなみに，RAISE の意味論マニュアルはできているが，難解でとても読めそうもない．実務家には公開していないようである．

学習者の背景も大切な要素で，やはり，抽象データ型とか，仕様のスタイルとか計算機科学の用語

に馴染みのある人や形式的方法の基礎を持っているほうが有利である．もちろんそうでない人でも学習可能であって，図式表現はそういう人の理解を助ける．また使いながら理解してゆくようにするとよい．RSL (RAISE Specification Language) で一番むずかしい概念は並行とモジュール化である．言語の意味がむずかしいものは，実務家にとってもやはりむずかしいということであろう．

2) 初期仕様について

顧客の要求を整理する方法としてCOREを使っているが，マニュアルの記述はやや貧弱で，この辺りは課題が残っているようである．面白いと思ったのは，モデルは詳細化の一方向だけでなく，具象モデルから抽象モデルに抽象化することが役に立つことがあるし，方法上も考慮すべきである，と提案していることである．

3) 支援系について

ほとんどの現場でツールを使っているが，シンタックスチェッカとタイプチェッカは仕様の正しさの確信に大変役に立つ．エディタや標準ライブラリを格納しているデータベースが使いづらいなどのクレームもある．マニュアルについていろいろな議論があったが，われわれ外部の人間には入手できる市販の教科書がないことがまず問題である．

プロジェクトは緒についたばかりで，まだ目的コードを生成するまでに至っていないからソフトウェアの生産性とか信頼性を云々できる段階にはないが，それにもかかわらずわれわれに貴重な教訓を与えてくれる．まず，形式的方法は実務家レベルで十分習得可能であること，そのためには，技術的に強力な生産者（指導者）が不可欠であること，さらには現場に普及させるためにツールは強力な武器になることなどである．

2.5 ツールセッション

14個のツールが展示実演していた．単なる実験ツールではなく実際のプロダクトに使われているものが多かったようである．LaCoS プロジェクトで使われているRAISE Toolsetを観察したので簡単に報告する．

タイプチェッカは機能している．単にシンタックスをチェックするだけではなく，スキーム（仕様の単位）の詳細化を行う際，データ型の詳細化の整合性はチェックできる．抽象データ型（仕様）を詳細化する際，公理部 (axiom) の正しさの証明を支援する部分はまだ十分できていないようで，今年中旬頃提供されるらしい．証明規則集ができているので，詳細化された公理がこの規則を満たしているかどうかをチェックするのだろう．RSL から Ada，C++ へのトランスレータはできているという．大概のツールは LATEX を使っていて出力は綺麗である．

3 シンポジュームの今後

VDM シンポジュームは 1985 年に設立された VDM ヨーロッパによって組織され，これまで4回のシンポジュームを開催したことになる．VDM ヨーロッパは形式的ソフトウェア開発法を産業界に移植しようとする研究者や実務家たちから構成され，第2回のシンポジュームまでは VDM を基本にしていたが，第3回で Z をその対象に拡張した．今回の第4回シンポジュームはさらに対象を拡大する過渡の会議として位置づけることができる．

今回の会議中，VDM ヨーロッパは FME (Formal Methods Europe) に改組することが確認された．対象とする方法は VDM，RAISE，Z，Larch，Lotos，ACT/ONE，OBJ など9つになる．さらに候補として4つが挙がっていた．(残念なことに，「FME 設立趣意書」を入手しそこなったので，正確な名称は記憶していない)．第5回の VDM'93 シンポジュームは欧州の形式的方法を網羅した会議になるだろう．

参考文献

[1] Deddo Wiersma, Kazuhito Ohmaki, Kokichi Futatsugi, "Specifications of a General User Interface in LOTOS and OBJ", COMPSAC91 Proceedings pp.90-97

[2] 佐伯元司,「ソフトウェア仕様化・設計の方法論の形式化について」，情報処理学会ソフトウェア工学研究会報告 90-SE-72-8

[3] 松浦佐江子,「 VDM による記述実験に基づく設計プロセスの一考察」, 情報処理学会ソフトウェア工学研究会報告 90-SE-80-8

[4] 秋山義博, 松村一夫,「第 13 回ソフトウェア工学国際会議 (ICSE'13) 報告」, 情報処理学会ソフトウェア工学研究会報告 90-SE-80-11

[5] 松村一夫,「第 13 回ソフトウェア工学国際会議 (ICSE'90) 」, コンピュータソフトウェア Vol.8 No.6 pp.66-72

[6] LNCS 551,"VDM'91 Formal Software Development Methods Vol.:Conference Contribution", Springer-Verlag, 1991

[7] LNCS 552,"VDM'91 Formal Software Development Methods Vo2.:Tutorial", Springer-Verlag, 1991

付録 A RAISE の具体的，モデル指向 (concrete,applicative) 例

```
DATABASE =
class
  type
    Db = Key ~m~> Data,
    Key, Data
  value
    empty : Db = [],
    insert : Key × Data × Db → Db,
    insert(k, d, db) ≡ db † [k ↦ d],
    remove : Key × Db → Db,
    remove(k, db) ≡ db \ {k},
    lookup : Key × Db ~~> Data
    lookup(k, db) ≡ db(k)
    pre k ∈ dom db
end
```

付録 B RAISE の抽象的，性質指向 (abstract,algebraic) 例

```
DATABASE =
class
  type Db, Key, Data
  value
    empty : Db,
    insert : Key × Data × Db → Db,
    remove : Key × Db → Db,
    defined : Key × Db → Bool,
    lookup : Key × Db ~~> Data
  axiom forall k, k1 : Key, d : Data, db : Db •
    [remove_empty] remove(k, empty) ≡ empty,
    [remove_insert]
      remove(k, insert(k1, d, db)) ≡
        if k = k1 then remove(k, db)
        else insert(k1, d, remove(k, db)) end,
    [defined_empty] defined(k, empty) ≡ false,
    [defined_insert]
      defined(k, insert(k1, d, db)) ≡
        k = k1 ∨ defined(k, db)
    [lookup_insert]
      lookup(k, insert(k1, d, db)) ≡
        if k = k1 then d else lookup(k, db) end
      pre k = k1 ∨ defined(k, db)
end
```