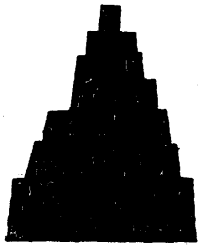

1. プログラミング言語の最近の動向について†

中　田　育　男††

1. はじめに

編集者からいただいた題目は「全体的展望」といったものであったが，本特集号でとり上げているプログラミング言語は大変多岐にわたっており，その全体に対して展望を述べるのは至難の業である．ここでは，その大部分の原稿を読みながらプログラミング言語の最近の動向について考えたことを述べることにしたい．

2. プログラミング言語の分野の広がり

プログラミング言語とは，人間が計算機にやってもらいたいことを正確に表現するための道具である．その意味では，FORTRAN, COBOL, Pascal などの汎用言語，手続き的言語はもちろんであるが，より特殊目的の言語，非手続き的言語，あるいは即実行型の指令言語で，その表現が後に残らないものでもプログラミング言語であるということができる．一般には，計算機にやらせるべき類似の問題が多数あるとき，それらに関して，計算機が処理できる程度に形式的な表現方法を決めれば，それがプログラミング言語であるということができる．プログラミング言語の世界をそのように見たとき，それは現在も将来も，常に膨脹し，発展し続ける世界である．

新しいプログラミング言語は，新しい表現方法に対する要求と，それを実現する技術とのバランスの上に作られる．その言語を使う側からは，問題をいかにして解くかという how よりも，できるだけ what（問題そのもの）の表現ですませること，あるいは，計算機に密着した表現でなく，より抽象化した表現が使えることが望ましい．また，性能に対する要求は使う人の立場によって重要度が異なる．それを実現する（たとえばコンパイラを作成する）立場からは，現在のハードウェア機器とソフトウェア技術（あるいはソフトウェア技術者）を使って，手頃なコストで実現でき，使えるものという要求がある．実際に得られる言語はそれらのバランスをとって作られるものである．そのバランス点は，上記の各条件が変化すれば変化し得る．結局，現状の技術が言語に集約されるので，プログラミング言語，およびそのユーザの動向から，計算機のハード，ソフト，応用のすべての分野の動向をうかがい知ることができる．

以下の事項に関するより具体的な記述は本特集のそれぞれの言語の項を見ていただきたいが，たとえば，関数型言語や述語論理型言語の研究の活発化，LISP のユーザ層の拡大，マイクロコンピュータ言語の動向などがハード／ソフト技術の発達と無関係ではないことを，人工知能用言語の動向から人工知能研究の発展の方向を，抽象データ型言語や並列処理言語や Ada 言語から，ソフトウェア工学が汎用プログラミング言語に及ぼした影響を，それぞれ知ることができる．また，仕様記述言語の動向から，ソフトウェア工学において仕様決定段階での工学が重視されてきていることが分かるであろう．もっとも，仕様記述言語といっても，現状ではプログラムの仕様が完全に記述できるわけではない．仕様のごく一部が記述できるだけであり，その記述内容に対する計算機処理も単純なものが多いが，それでもある程度の効果はある．仕様をより完全に記述しようとする試みもあるが，それは容易ではなく，プログラムそのものを記述／解読するのとどちらが楽か分からなくなることも多い．記述および処理の努力と，記述したことによる効果とのバランスで，実用的な言語が設定されているのである．

本特集号には，なぜか，ジョブ制御言語（JCL）の項が入っていないが，プログラミング言語を前記のように広くとらえてみれば，ジョブ制御言語は計算機システムと利用者とのかかわり方を決める重要なプログラミング言語である．通常のプログラミング言語は本特集号に見られるように進歩を重ねているが，このジ

† On Current Trends of Programming Languages by Ikuo NAKATA (Institute of Information Sciences and Electronics, University of Tsukuba).
†† 筑波大学電子情報工学系

ョブ制御言語，特に大型 OS のそれについては，依然として，アセンブリ言語の記述形式で融通がきかない，ハードウェアを意識しなければならぬ，TSS とバッチで体系が違う，ファイルの使用法が複雑である，プログラミングの能力が弱い，といった悪評が聞かれる．ジョブ制御言語で最近評判のよいのは UNIX のそれである．大型 OS の場合は，様々の利用者層を考慮する必要があること，従来からの OS のしがらみがあることなどがあり，UNIX の場合は利用者層を単純化している，という違いはあるとしても，利用者の立場をより重視した設計理念や設計技術の違いが，ジョブ制御言語に現れているのであろう．

これからは，パーソナルコンピュータと大型機のネットワークの時代になるといわれている．パーソナルコンピュータでは計算機の効率よりもまず使い勝手が重視され，現在の研究の重点もそこにあるようである．UNIX やそれらの経験から，大型 OS とジョブ制御言語も大きく変貌していくことを期待したい．

3. プログラミング言語の普及と規格化

プログラミング言語は言語として人に使われるべきものである．多くの人に使われて初めて一人前の言語になるということもできる．多くの人に使われるためには，いい言語であること，強いサポート（処理系の提供者など）があること，そして，その言語に強く共鳴するユーザ（あるいはユーザのマネージャ）が居ることが必要である．いい言語といえるためには前記のようなバランスのとれたものでなければならない．強いサポートの例としては，PL/I に対する IBM 社，COBOL や Ada に対する米国国防総省などがある．Pascal がまず大学で普及したのは，いい言語であるからだけでなく，大学でもできるような簡単なサポートで使えるものにしたからでもあろう．

ただし，多くの人に使われない言語が意味がないわけではない．使われる場所が局所的であっても，そこで効果があれば十分意味があるわけであるし，その経験が，より広く使われる言語の設計に生かされればよい．Pascal から Ada に到るまでの間には多くの言語が研究開発されたが，それらの経験が Ada に生かされたものも多い．Ada が全体的にバランスのとれた言語であるかには疑問もあるが，やはり一つの時代を画する言語といえるであろう．今後もまた，次の時代に向けて，いくつかの実験的言語の経験をつみ，そこから新しい言語体系を作りあげていくことは行われていくと思われる．

プログラミング言語が多くの人に使われるようになると方言が現れる．ユーザは常に自分にとってよりよい言語を望むものであるし，メーカ（実現者）は他所よりもよいものを作ろうとしたり，与えられた条件の中でバランスをとろうとしたりするからである．それは，言語の進歩をうながす原動力ともなるものではあるが，プログラムの相互利用や財産としての蓄積を考えたら，あまり好ましいことではない．そこで，プログラミング言語の標準化あるいは規格化が必要になる．本特集でも，主要な言語の項では規格について述べられている．

すでに ISO の（国際）規格ができているのは，FORTRAN, COBOL, ALGOL (ALGOL 60), BASIC, PL/I であり，JIS 規格ができているのは FORTRAN, COBOL, ALGOL である．ISO 規格の作成作業中であるのが Pascal と APT であり，APL と Ada についてはその作業が始められようとしている．文書整形用言語やデータベース用言語についてもその動きがある．

規格化の一般的な目的は上記のようなことであるが，具体的な目標はいろいろあり得る．その一つは，世の中ですでに使われているものの共通部分をとって，その規格の範囲内ならばどこでも通用するようにするものである．最初の FORTRAN 規格がそれに該当する．強力な機関でまず言語仕様を定め，それを世の中により広め，かつ方言の発生を抑えるために，その言語（あるいはその部分集合）を規格化する場合もある．COBOL がそうであった．Ada もその流儀でやろうとしているようである．時代の要求にこたえて，あるいはそれを先取りして，既存の言語を改良，拡張していく場合もある．FORTRAN 77 が前者であり，FORTRAN 8 X は後者をねらっているようである．

この例で見るように，規格化がすんだらそこでその言語の成長が止まるわけではない．それが当分，規格として通用するのではあるが，それと並行して，次の規格化に至るサイクルがまた始まるのである．ただし，その周期は一定期間以上とし，言語の拡張にあたっては従来の機能と矛盾しないものとする，など，ソフトウェアの蓄積を無駄にしない配慮は重要である．COBOL の第3次の規格化が遅れている一つの理由は，この互換性の問題にあるようである．また，ALGOL のように，使われなくなって成長の止まるものもある．

COBOL や FORTRAN の規格は，まず ANSI (米国) 規格が作られ，それがほとんどそのまま ISO (国際) 規格となり，次に JIS 規格となってきている．規格を世界的に統一するためには ISO 規格と国内規格が一致するのが望ましいからそのこと自身は悪くない．ただし，日本には漢字やかな文字による特殊なデータがあり，最近は計算機による日本語データ処理も一般的になってきた．COBOL の項でも述べられているように，これに関しては，国内独自の規格化を，国際規格との整合性も考慮しながら，推進する必要があると思われる．

規格の大切な役割りは，言語の仕様がそこに厳密に定義されていることである．しかし，厳密さと分かりやすさを両立させることは難しい．PL/I の規格は，厳密な記述を目的として，ULD 記法を用いて作られたが，記述が複雑で理解困難ということで，今度は，自然言語による記述が議論されている．Pascal の従来の文法書では仕様の細部にあいまいな点があったので，その規格化の主目的は仕様の厳密な定義にあったと思われるが，規格案の英語による記述は大変分かりにくいものである．厳密さと分かりやすさの両立は簡単でないとは思うが，上記の現状はもっと改善されてしかるべきだと思う．

4. 日本製の言語

本特集号でとり上げられた主要な言語の中には，我が国で開発されたものはない．また，前記の規格案に関しても，日本から修正意見は出すが，日本で主たる部分が書かれたものはない．

日本で開発された言語で，本特集号中で触れられているのを列挙すると，PM(マクロ言語)，SDMS/SDL，SSD，HISP (以上は仕様記述言語)，Iota (抽象データ型言語と言語設計プロジェクトの項で)，S-Net (人工知能用言語)，PROGRESS (事務処理用言語)，INTERACT (エンドユーザ用言語)，APAD (の中の OFP が言語) (データベース用言語)，AL (数式処理言語)，LDS，TBM (以上，ハードウェア記述言語)，FAPT，HAPT，HMESH，MINIAPT (以上，数値制御用言語)，TPL/1，FADEBUG-I，SOLDA (以上，テストプログラム記述言語) くらいであるし，それらは必ずしもその分野における日本の代表的言語として選ばれているわけでもなさそうである．また，プロセス制御言語の項のように，国産の言語があってもそれに触れられないものもある．そのほとんどが，実用のものとして (実験的な性格のものもあるが) 使われてはいるが，開発者 (あるいは開発社) 以外の所への広がりはまだないようである．しかし，これらの言語が目差している分野には，今後の発展が期待されるものが多いから，この中から，あるいはこれらの経験をふまえた新しい言語が，世界に通用していくことを期待したい．より汎用のプログラミング言語についてもそれを望みたいが，そのためには，輸入品には共鳴するが国産品には共鳴しないといった通性をあらため，英知を結集する協力体制をとることが必要であろう．

5. おわりに

プログラミング言語が計算機科学の中心的課題であったのは，1950，60 年代であり，70 年代以降は，(依然として重要ではあるが) 中心からは降りているといわれる．確かにそういえるかも知れないが，プログラミング言語を前記のように広くとらえてみれば，それは，常に重要な位置を占め続け，発展していくものである．

以上，本特集の原稿を読んでの感想を述べてみた．個人的には，よく知らなかった分野の動向も要領よく教えていただいたという気がする．分野によって対象や目的が異なるので，言語も独立に開発されてきたものが多いが，それぞれの分野の成果がお互いに刺激となって，次の発展をうながしていくことも期待される．本特集はそのようなことを考えるきっかけにもなるであろう．

(昭和 56 年 5 月 7 日受付)