U.D.C. 800.92:519.682

プログラミング言語の最近の動向

Recent Trends of Programming Languages

渡辺　坦* *Tan Watanabe*
松本恵一** *Keiichi Matsumoto*
林　利弘*** *Toshihiro Hayashi*

プログラミング言語に対しては，プログラムの作りやすさ並びに記述力，移行性及び性能の高いことが要求される。これらの目的に合わせて，プログラムを機械の動作に合わせた形から人間の思考形態に合わせた形へと近づける高級言語化の努力が続けられ，高級言語は，技術計算や事務計算に始まり，機器制御，マイクロコンピュータ，システムプログラムへとその適用範囲が拡大されてきた。

本論文では，この流れを分析してそれらを支える言語上の技法を概観し，FORTRAN，ALGOL，COBOL，PL/I，APL，Pascal，SPL，PL/Hなどの代表的言語でそれらがどのように実現されてきたかについて述べる。また，Adaやその他の最近の言語では，何が主要課題となっているかについて言及する。

1 緒　言

コンピュータが普及し始めた1960年ごろと今日とを比べると，コンピュータの性能とコストの比は100倍以上に向上しているが，それを動かすプログラムの生産性の向上は数倍にとどまっているといわれる。一方，プログラムに対する要求の複雑さは何十倍にもなっている。大形オペレーティングシステムを例にとると，当時は10万ステップに満たなかったが現在では数百万ステップにも達している。

プログラミング言語に対しては，処理内容の複雑化と多様化に対処することが要求される。本論文では，この要求を分析し，それを満たすためにプログラミング言語がどのように発展してきたかについて概観する。言語の個々の概念の発展経過などについては，他の文献[1,2]を参照されたい。また，各種ツールと結合して，設計から製作，保守を一貫して行なうプログラミングシステムについては，本特集の他の論文[3,4]を参照されたい。

2 プログラミング言語に対する要求の変遷

2.1 記述方式に対する要求

プログラミング言語は，**図1**に示すように，プログラムの作成や保守を容易にする生産性向上を主目的として進歩してきた。

記述水準については，日常使われる記法に近い式のレベルで記述する高級言語や，特定分野に特有の表現による簡潔な記述のできる問題向き言語，あるいはコンピュータの専門知識がなくても使えるエンドユーザー言語へと発展するに伴い，コンピュータの機械語対応に記述するアセンブリ言語に比べ，数分の一から数十分の一の記述量で簡潔に書けることが要求されてきた。

記述力については，古くからの技術計算や事務処理のほかに，機器制御やプログラム管理などを行なうシステムプログラムにも適用できるよう，データ形や演算機能の拡張などが図られてきた。

移行性の向上は，プログラムの共用と流通の容易化，及び教育の効果を上げるための要求であり，プログラミング言語に対しても，JIS（日本工業規格）やISO（国際標準規格）が定められてきた。

プログラミング方法論との融合は，ソフトウェア開発の効率化を目指す構造化プログラミングなど[5]の技法を採り入れて，明解な良いプログラムを効率よく作れる言語[6),10),13)]を求める動きであり，どの言語にもこの点の配慮が求められるようになった。

信頼性の向上は，コンピュータの社会的重要性にこたえるプログラム品質の向上と，工数的に最大の比率を占めるデバッグ工程（誤りの検出と除去）の工数削減のために，最近特に

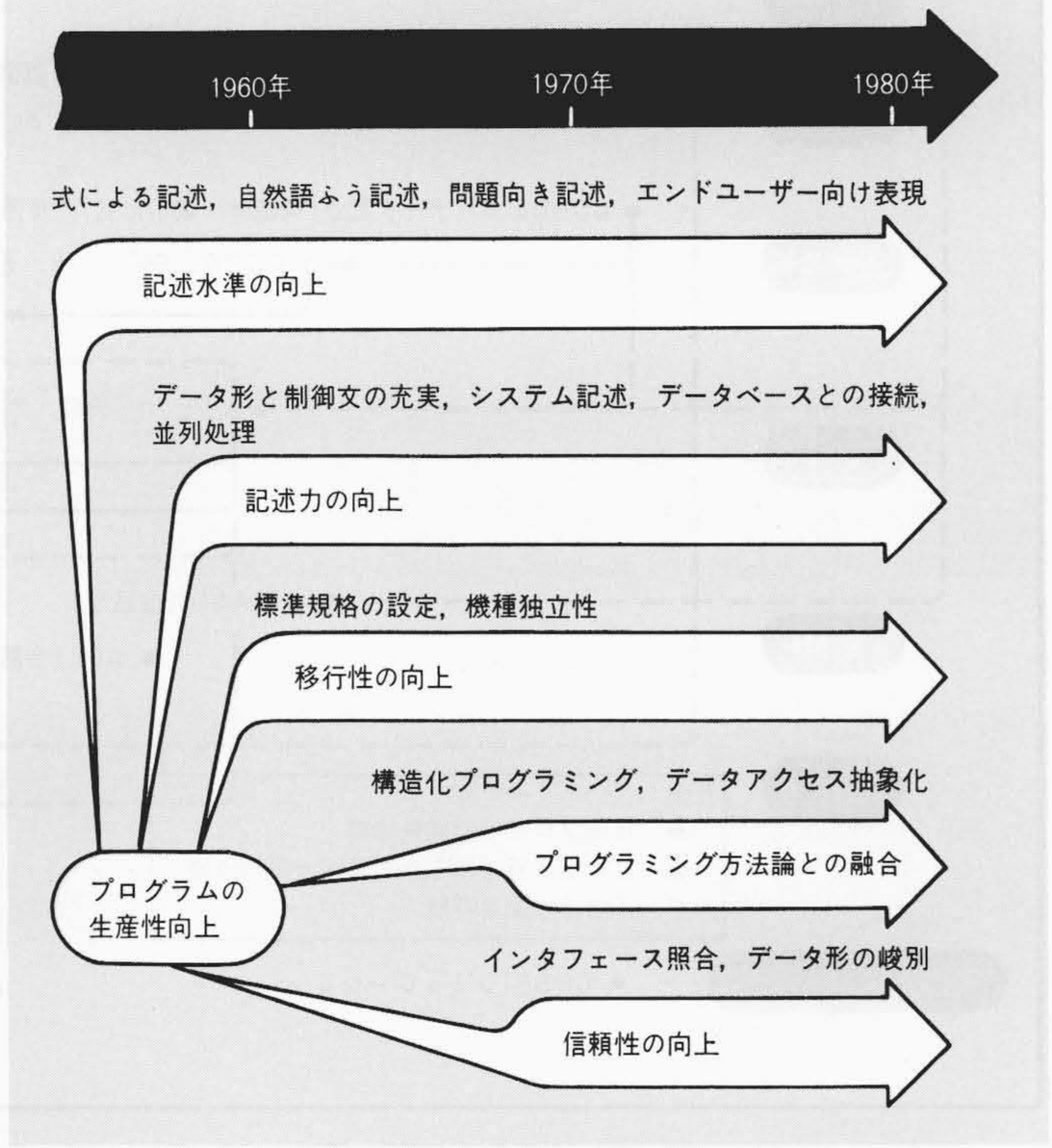


図1　プログラミング言語に対する要求　　プログラムの生産性向上を求めて，プログラミング言語に対しては，記述力だけでなく，品質の良いプログラムが作りやすいことなどの新しい要求が加わってきた。

* 日立製作所システム開発研究所　** 日立製作所ソフトウェア工場　*** 日立製作所大みか工場

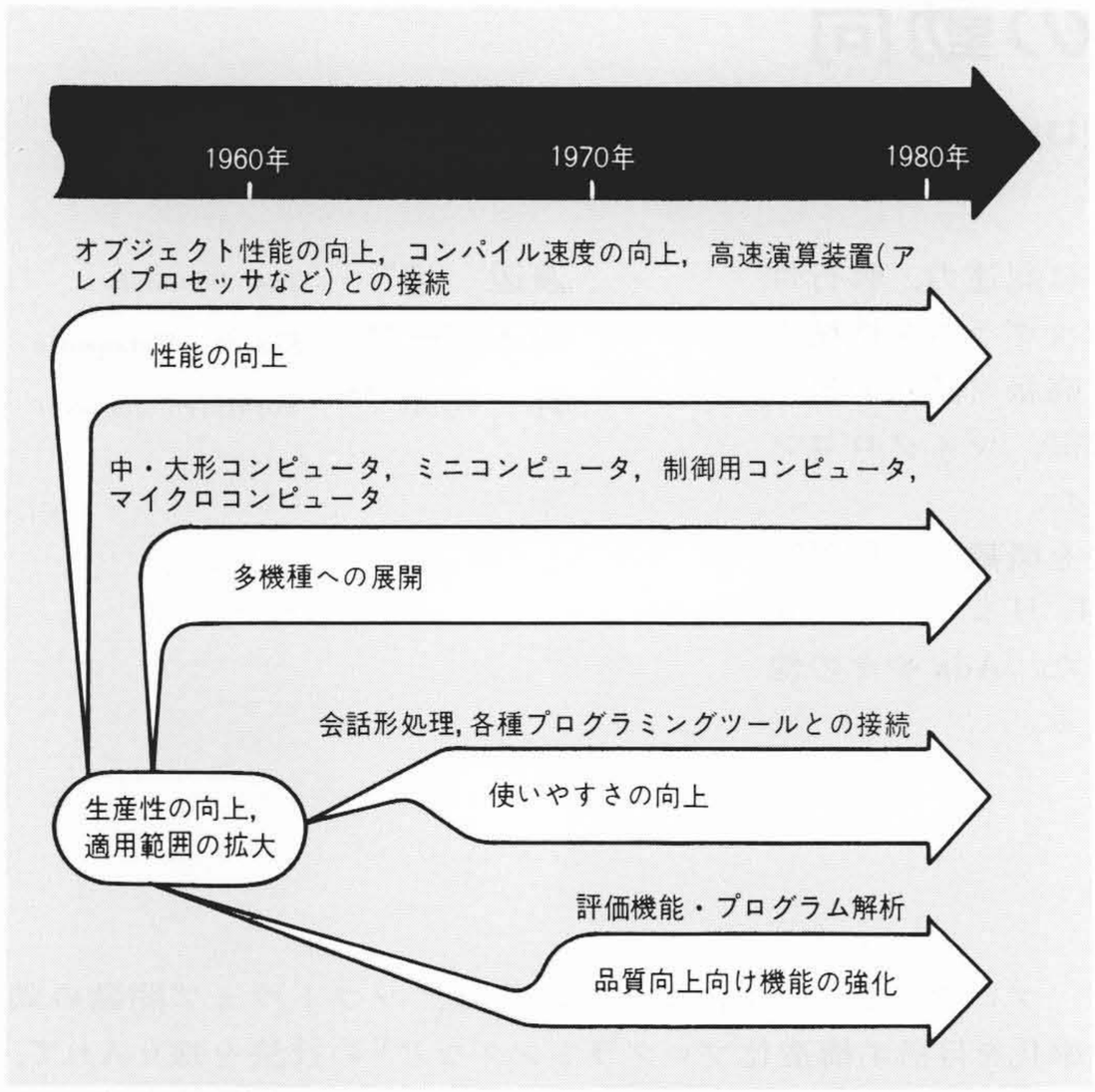


図2 言語プロセッサに対する要求 高級言語をより広い分野でより使いやすくするために，その言語プロセッサの改良，拡充が継続的に進められてきた。

重視されており，プログラムの構成部分間のインタフェースやデータの使い誤りを防ぐ機構の導入などが求められている。

2.2 言語プロセッサに対する要求

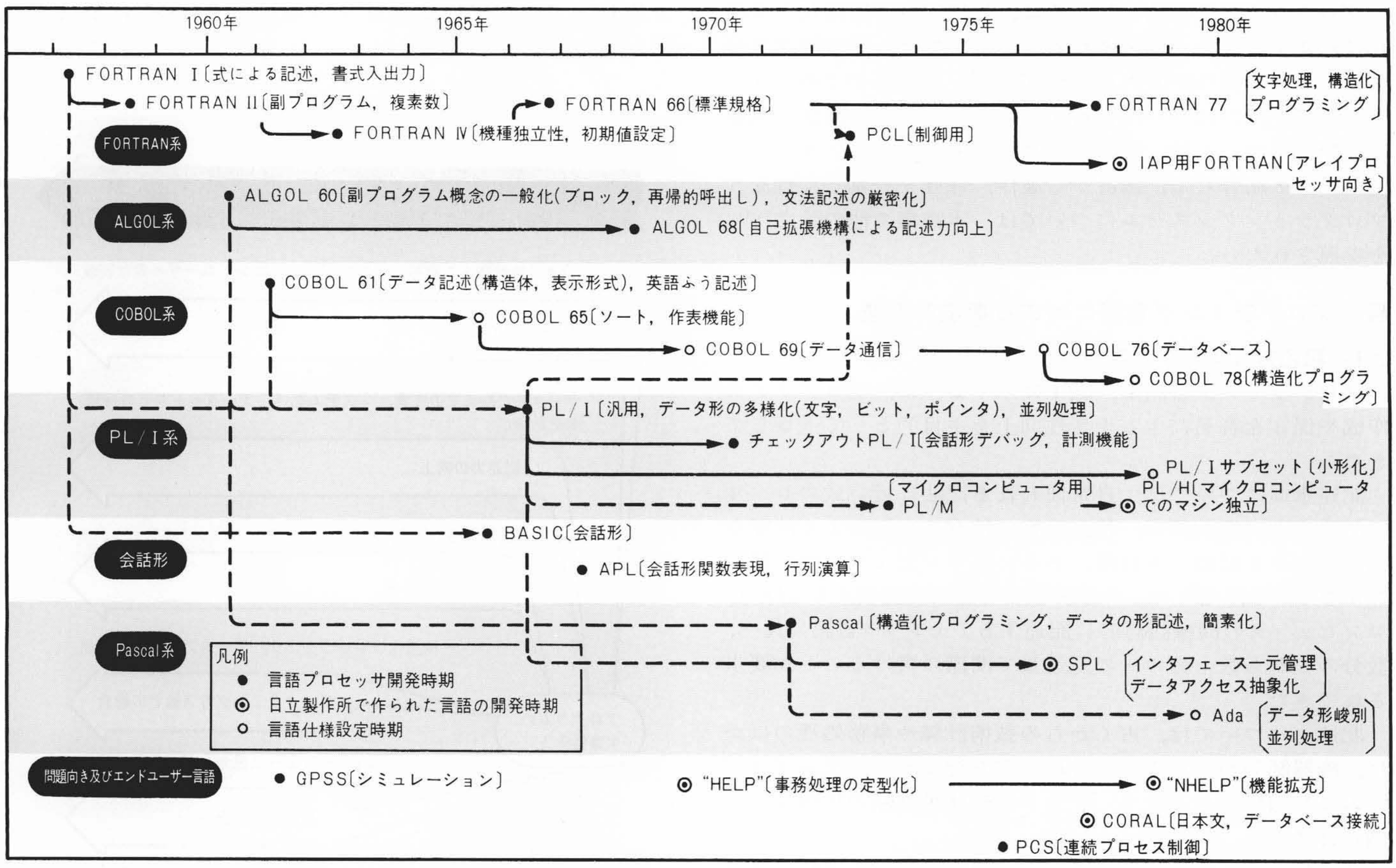
人間の書いたプログラムをコンピュータで実行できる目的プログラムに変換するのが，コンパイラやアセンブラなどの言語プロセッサであり，これに対する要求は**図2**に示すように要約される。

性能の向上は実用上非常に重要であり，実行時間と使用メモリ量の少ない目的プログラムを生成することと，コンパイル時の処理時間と使用メモリ量の少ないことが要求される。

多機種への展開は，高級言語化と移行性向上を，大形コンピュータや制御用コンピュータにとどまらず，マイクロコンピュータにまで要求する形で，強力に進められてきた。

使いやすさの向上は，会話形処理による結果の即時確認，あるいはプログラムの編集や保守用ドキュメント作成を行なう各種プログラミングツール[7]，更には設計，製作，保守を一貫して行なうプログラミングシステム[3,4]などにより，プログラミングを容易に行なおうとする要求である。

品質向上向け機能は，未テスト部分の検出やプログラムの流れを分析して，機械的に分かる矛盾を検出しようとするもので，信頼性向上の要求に従って研究開発が盛んになってきている。



注：略語説明
IAP(Integrated Array Processor)
APL(A Programming Language)
SPL(Software Production Language)
"PL/H"("Programming Language for Hitachi Microcomputer System")
PCL(Process Control Language)
"HELP"("Hitachi Effective Language for Programming")
"NHELP"("New HELP")
CORAL(Customer Oriented Application Program Development System)
PCS(Process Control System)
Ada(米国国防総省開発推進中のシステムプログラミング用言語)

図3 プログラミング言語発展の経緯 新しい要求に応ずるための言語が開発されるとともに，古い言語にもそれらに対応する機能が組み込まれてきた。

表1　各種言語の特性　　各言語は，それぞれ特色がある。その特長を生かした使い方をすることが大切である。

項目 ＼ 機種 ＼ 言語		汎用							制御用		マイクロコンピュータ
		FORTRAN	ALGOL	COBOL	PL/I	BASIC	APL	Pascal	PCL	SPL	PL/H
記述水準	式による記述	○	○	○	◎	○	◎	○	○	○	○
	自然語ふう記述		○	◎	○			○		◎	○
	問題向き記述									○	
記述力	数値データ処理	◎	○	○	○	○	◎	○	○	○	○
	文字データ処理	→○		○	◎		○	○		○	○
	高自由度入出力	○		◎	◎				○	○	
	構造付きデータ処理			○	○			○	○	○	○
	制御データ処理				○				○	○	○
	並列処理				○				○	○	○
移行性	標準規格	○	○	○	○	○	検討中	検討中	—	—	—
	機種独立性	→○	○	○	○	○	○	◎	○	○	○
信頼性・保守性	構造化プログラミング	→○	○	→○	○			○		○	○
	データアクセス抽象化									○	
	インタフェース照合				○			◎		◎	
	データ形照合				○			◎		○	
	インタフェース一元管理									○	
手軽さ	言語仕様の簡潔さ					○		○			
	会話形処理	→○		→○	→○	◎	◎	○			
主用途		技術計算	技術計算	事務計算	汎用	中小規模技術計算	技術計算，事務計算	技術計算	実時間処理，技術計算	実時間処理，技術計算	システムプログラム
日立製作所における主なソフトウェア製品		拡張FORTRAN 最適化FORTRAN 最適化FORTRAN 77 実行形FORTRAN	VOS2/VOS3 ALGOL	拡張COBOL COBOL会話形デバッグ	最適化PL/I チェック型PL/I	VOS2/VOS3 BASIC BASICマスタ	VOS3 APL APL/FILE APL/TOOL	VOS2/VOS3 Pascal	HIDIC 80 PCL	HIDIC 80 SPL	6800 PL/H

注：略語説明など　VOS(Virtual Storage Operating System)，○（該当項目を含む。），◎（該当項目に優れている。），→○（該当項目を含むよう改良された。）

3　プログラミング言語発展の経緯

3.1　プログラミング言語の諸系列

言語に対する要求の多様化，高度化に合わせて多くの言語が開発されたが，それらは**図3**に示すように系列化されるであろう。**表1**は代表的言語の特性を示したものである。

日立製作所でも，**表1**の下段に示すように，大形コンピュータのHITAC　Mシリーズから小形コンピュータのHITAC Lシリーズ，制御用のHIDIC，あるいはマイクロコンピュータのHMCS6800に至るまで，多くの言語が装備されている。言語仕様については，標準規格の定められているものはそれに合わせてあるが，性能や使いやすさの向上のために，多くの努力が払われている。

(1)　FORTRAN系

1957年に開発されたFORTRANは，実用的な最初の高級言語である。これは，プログラムの部品分けを可能とする副プログラム機構や，使用経験に基づく各種機能の追加，改良を経て，技術計算を中心として広く普及するに至った。

実用性向上のためにコンパイラ性能の向上が追求され，最適化FORTRANでは，プログラム中の処理の流れを解析して同じ計算の繰返しを避けるなどの高度の最適化処理を自動的に行なうので，アセンブリ言語を使う場合よりも効率の良いプログラムが容易に作れるというレベルに達している。更にいっそうの高性能化を図ったものとしては，高速演算装置であるアレイプロセッサを自動的に効率良く使うIAP用FORTRANがある。

制御コンピュータ用には，異種データの集合である構造体や，実時間処理の記述機能を付加したPCL[8]（Process Control Language）が利用されている。

この発展過程を通じて蓄積されたソフトウェアを継承し，新しい要求にもこたえられるようにするため，構造化プログラミングや文字処理の機能を追加したFORTRAN77の規格が定められ，その高性能コンパイラは最適化FORTRAN77として実現されている。

(2)　ALGOL系

高級言語によるプログラミングの諸概念の整理と体系化はALGOLで行なわれ，副プログラム中の副プログラムを実現するブロック概念の導入，機種によらない共通言語とするための概念の抽象化と厳密な文法記述の努力がなされた。このことは，以後の言語やコンパイル方式に大きい影響を与えている。

(3)　COBOL系

事務処理向きの言語としては，COBOLが作られた。これは，帳票の構成やその各欄の形式など，データを記述する能力に優れ，表現形式も英語の文章に近い高水準に設定された言語であり，最も多くの人々が使っている言語である。

他の多くの言語は，標準仕様の制定が実用化の後を追っているのに対し，COBOLでは標準化が先行している。すなわち，ソートや作表機能をもつCOBOL65，データ通信機能をもつCOBOL69，データベース機能を含めたCOBOL 76，構造化プログラミング機能を含めたCOBOL78というように，各時代の新しい要求に合わせた標準仕様が次々と制定され，コンパイラ開発がそれに追随している。

(4) PL/I系

PL/Iは，FORTRANとALGOL，COBOLの諸機能を包含し，更にリスト処理や並列制御などの機能をもつ汎用言語であり，FORTRANと肩を並べるまでに普及してきている。PL/Iの功績は，それまでアセンブリ言語の独壇上と思われていたシステムプログラムに高級言語を導入する方向を示した点にもある。また，PL/Iを縮小したシステムプログラム向き言語も多く作られた。マイクロコンピュータ用高級言語“PL/H”[9]（“Programming Language for Hitachi Micro computer System”）はその一つである。最近米国の小形コンピュータでは，PL/Iの機能を縮小したサブセット言語も普及し始めた。

コンパイラは高性能化と高機能化が図られ，最適化PL/Iコンパイラは，プログラムの性能分析と未検査箇所摘出のために，各文の実行回数を出力する機能をもっている。

(5) 会話形言語

会話形処理は，COBOL会話形デバッグやチェック形PL/Iなど，上記言語に対しても実現されているが，言語仕様を即時処理向きに定めた言語としてはBASICが著名であり，手軽さが好評を得て，大形コンピュータの端末からマイクロコンピュータに至るまで広く利用されている。

APL(A Programming Language)は，コンピュータによる操作を，配列を被演算数とする数式として著しく簡潔に表現する言語であり，これまでに述べた言語よりも更に高水準で会話形処理に適している。APL/FILEやAPL/TOOLなどは，APLにファイル処理やプログラム編集の機能を付加し，いっそう使いやすくしたものである。

(6) Pascal系

Pascalは良いプログラミング方法を教えるために開発された言語で，簡素な仕様であるにもかかわらず多様なデータ表現が使え，構造化プログラミングの機能とあいまって，分かりやすく誤りの少ないプログラムを作りやすい言語である。

Pascalの思想を加味して制御用コンピュータ向きに開発されたのがSPL[10,11](Software Production Language)である。これは，ソフトウェアエンジニアリング指向の最新技術に基づく構造化プログラミング言語であり，処理モジュール間で共有される変数を環境モジュールとして独立にコンパイル可能にし，インタフェースの不一致による誤りの大幅削減などを図っている。

(7) 問題向き及びエンドユーザー言語

記述の効率化と利用者層の拡大を求めて，多くの問題向き言語やエンドユーザー言語が作られた。HITACのユーザーと日立製作所とで共同開発された言語としては，事務処理を定型化してパラメータで指定できる簡便な言語“HELP”（“Hitachi Effective Language for Programming”）と，それを拡充整理した“NHELP”（“New HELP”）があり，日本語で書かれた事務処理の仕様をそのままプログラムとするものとしてCORAL(Customer Oriented Application Program Development System)がある。

制御用コンピュータの分野では，連続プロセス処理からマンマシン処理までを簡便な形で指定できるPCS[12](Process Control System)言語などが開発されている。

3.2 新たな展開

最近の重要課題は，誤りの少ないプログラムを書く方法とシステムプログラムを機種依存な形でなく書く方法とである。米国国防総省では，このための言語としてAda[13]の開発を推進している。Adaではデータの使い誤りを防ぐために，演算や操作について，その対象となるデータの形や使用可能範囲を厳しくチェックする。また，複数の処理を並列的に実行する際，一方から他方の呼出しがあり，相手方は呼出し受入れの表明をしているときに初めて呼び出された処理が実行されるという，いわゆるランデブー方式で制御する。そのほか，Adaは，各種例外処理機能や，SPLと同様にモジュール間の共通変数の分離コンパイルなど，多くの機能をもち，**表1**に示したほとんどすべての項目を満たそうとする意欲的な言語である。しかし，性能の良いコンパイラを作る方法など，Adaに対してはまだ研究すべき課題がかなり残されている。

現行の言語は，大部分コンピュータでの処理手順を記述するものであるが，入力と出力の関係は与えるが，出力を導出する手順の組立ては自動化しようとする試みも行なわれている。これを発展させて，処理内容を多くの部分処理に分け，各部分処理は必要なデータがそろうと実行される方式のデータフロー言語[14]も研究されているが，実用化までにはまだ多くの問題が残されている。

4 結　言

プログラミング言語は，プログラムを作りやすくするために，機械の動作に合わせた形から人間の思考形態に合わせた形へと進んできており，最近では信頼性向上の方法が主要課題の一つとなっている。高級言語化に伴う性能上の問題は，コンパイラ技術やハードウェアの改良によって解決を図ってきた。

最後に，執筆に当たり有益な助言をいただいた筑波大学制御情報工学系教授・中田育男理学博士に対して深い謝意を表わす次第である。

参考文献

1) 中田：プログラミング言語の歴史と展望，情報処理，**21**，574～582（1980-5）
2) P. Wegner：Programming Languages—The First 25 Years IEEE C，25，12，1207～1225（1976-12）
3) 松本，外：業務アプリケーション向け構造化プログラミングパッケージ“UDDT”，日立評論，**62**，875～878（昭55-12）
4) 森，外：制御用ソフトウェア一貫プログラミングシステム—SPLとその周辺システム—，日立評論，**62**，889～892（昭55-12）
5) ダイクストラ，外(野下，外共訳)：構造化プログラミング，サイエンス社（昭50）
6) T. Watanabe, et al.：EXTRAN—A Top-Down Programming System, Proc. of IFIP 74, North-Holland（1974）
7) T. Chusho, et al.：Two-Stage Programming：Interactive Optimization after Structured Programming, Proc. of 3rd UJCC（1978）
8) 桑原，外：プロセス制御用言語PCLの開発，日立評論，**54**，765～770（昭47-9）
9) 吉村，外：マイクロコンピュータ用高級言語PL/Hシステム，日立評論，**61**，307～310（昭54-4）
10) 林，外：制御用トップダウン・ストラクチャード・プログラミング言語—SPL—，日立評論，**60**，3，235～240（昭53-3）
11) 野木，外：階層化プログラミング言語SPLの評価，情報処理学会第3回ソフトウェア工学研究会（1973-3）
12) 長谷川，外：化学プラントにおける計算機制御システム，日立評論，**58**，451～456（昭51-6）
13) J. Ichbiah, et al.：Preliminary Ada Reference Manual., Sigplan Notices, **14**，6，Part A（1979-6）
14) D. P. Misunas：Workshop on Data Flow Computer and Program Organization, ACM SIGARCH, **6**，4（1977-10）