プログラム言語に関する国際会議に参加して*

森口繁一**

私は本年3月，ミュンヘンおよびローマで開かれた二つの国際会議に出席した．ミュンヘンの方は小さい技術委員会の会合，ローマのは大きいシンポジウムであったが，両方とも電子計算機のプログラム言語に関するものであった．ここにその概要を報告する．

1. ミュンヘンの会議

1962年3月20日，国際情報処理連合（IFIP）のプログラム言語技術委員会（TC 2 "PL"）の第1回会合第1部が，ミュンヘン工科大学の計算センターで開かれた．

出席者は，ドイツの F.L. Bauer，イタリアの Caracciolo，アメリカの Clippinger，ノルウェイの O.J. Dahl，オランダの E.W. Dijkstra，フランスの F. Genuys，ポーランドの A. Mazurkiewicz，日本の森口，イギリスの M.V. Wilkes およびオーストリアの H. Zemanek（委員長）であった．

議事の第1は，「委員会の仕事の範囲とプログラム」で，はじめ IFIP 会長 Auerbach が作った原案を，いろいろ討議してなおした．しかし，それが後にまた IFIP 評議会での長い討議の結果，いろいろ修正されて，結局次のようにきまった．

プログラム言語に関する IFIP/TC 2 の仕事の範囲

(1) 委員会の仕事の範囲は共通プログラム言語の開発，規定および改良を促進し，改訂，拡張および改善を図ることであるものとする．

(2) 仕事のプログラムは次の事項を含まねばならない．

(a) 形式的言語に関する一般的諸問題，たとえば概念，記述および分類．

(b) 特定のプログラム言語の研究．

(c) 必要があると思われる新しいプログラム言語を研究し，適当ならば，その合成の仕事の調整にあたること．

(3) 技術委員会は，作業グループの設置が適当であるならば，適当な時機にその設置を要請することができる．

(4) 他の適当な国際機関との連絡を確立し，かつ維持すること．

第2に，短期の事業として，各国におけるこの分野での活動状況を知るために調査をすることになった．また，ALGOL に関する作業グループ（IFIP/WG 2.1）の設置を要請することにきまった．

第3に，ALGOL の保守について相談した．アメリカ計算機学会（ACM）は ALGOL の保守を IFIP が引き受けるよう要請してきているので，ALGOL 60 報告の 13 人の著者が同意すれば，IFIP/WG 2.1 "ALGOL" がただちに行動を起こすように取りはからうことにした．

後に IFIP 評議会は IFIP/WG 2.1 "ALGOL" を設置し，その委員長にはオランダ Delft 大学の W.L. van der Poel（ファン・デア・プール）氏を任命した．委員の数はできるだけ少数としようということになった．

なお，保守（maintenance）という言葉についての異論が出て，それは修理（repair）という意味に近すぎて不適当だというので，前記のように改められた．

第4に，他の機関および団体との協力関係については，同じ人が両方に関係するという形で連絡を緊密にしようという方針がきまった．

Remington Rand Univac の E.R. Utman 氏が TC 2 の書記に選ばれた．

2. ローマの会議

情報処理における記号言語に関するシンポジウム（Symposium on Symbolic Languages in Data Processing）が3月26日から31日まで，ローマ南郊 E.U.R. の会議宮殿（Palazzo dei Congressi）で開催された．

講演は全部で 51，パネル討論が六つあった．内容の詳細は Gordon and Breach 社（150 Fifth Avenue, New York 11, N.Y., U.S.A.）から発行される Proceedings に掲載されるはずである．

以下に，いくつかの講演について，メモを頼りに，印象に残っていることを思い出してならべてみる．

---

* Reports on the IFIP/TC 2 meeting in Munich and the ICC Symposium in Rome, by Sigeiti MORIGUTI (University of Tokyo)

** 東京大学

Gorn: 接頭言語（prefix language）という概念を用いて，影響範囲の解析や深さの解析が形式的にうまく記述できるという話(らしい)．後にパネル討論のときに同氏は，この概念を利用すれば言語の処理ルーチンが自動的に作れると述べている．

Naur: ALGOL の考え方，概念および特徴．人と人との間の情報伝達を重視してある．概念の数は少ないほどよい．for statement なんかは procedure として

FOR (j, p, −1, q, w);

のようにしてもよかった．

Woodger: ALGOL 60 では，計算のために real と integer，そして選択のために Boolean を使っている．label とか procedure とかいうものは名前(name)であって，間接の引用（間接の制御飛越）のために使われる．……

v. Wijngaarden: ALGOL の拡張（より一般化すること）の提案．value という specifier をうんと広義に使うこと，new ⟨identifier⟩ という表現があることなどが特徴である．

Schwartz: JOVIAL の話．コンパイラーは5万語で，完成に1年を要した（各機種ごと）．中間言語を適当に選んで努力の重複を避けることが重要．対象とする計算機は高速記憶 16 K，テープ6本というのが最小である．

Katz: GECOM の話．人間は機械のように考えるものではなくて人間らしく考えるものであるという前提で作った．COBOL を基本とし，ALGOL の概念と記号を用いた．コンパイラーの能率を高め，もとの言語で訂正が行なわれるようにした．現存するすべての言語が使用できる．

Brooker: ATLAS 用の Autocode．オペレーションズ・リサーチや市場調査なども含む一般用途を頭において作った．if((1>x≥0) & (1>y≥0)) → 3 などという表現ができる．

Bosche: COMIT の話．記号のつながりを処理するのが主眼．コンパイラーとインタープリターとをどこで分けるかが問題．

Samelson: ドイツとスイスを中心とする ALGOL の実施化のための ALCOR グループの話．

Huskey: Berkeley の Univ. of Calif. でやっている NELIAC コンパイラーの話．簡単で速いのが特徴．予備処理と翻訳とアセンブリーとの3段に分けて処理する．

van der Poel: ZEBRA という機械のための ALGOL 60 の翻訳ルーチン．容量は 8 K．固定小数点の機械なので実行段階で通訳ルーチン(interpreter)を使う．翻訳ルーチンそのものは2,500語．それが「あとのほう」にはいる．実行段階ではその場所に通訳ルーチン（1,500 語），浮動小数点演算ルーチン（250語），および入出力サブルーチンがはいる．

Levine: 種類のちがった記号言語の混用．

Dijkstra: 概念をできるるかぎり統一しようという試み．変数も演算子の一種とみるという徹底ぶりで，したがって，x:=3 は procedure declaration の一種になる．「きわめてエレガントであり，誤解のおそれがあるほど単純である」と自称している．

Nagao: 小さい機械では ALGOL にいろいろ制限を加えるのがよいという意見．

Seegmueller: ALGOL 60 のコンパイラーで，添字付変数の処理を能率化する話．PERM, ERMETH, Siemens 2002, TR 4 などで実施している．for ループの中で，添字 y_k がすべて制御変数 i の1次式 $d_k+i\times c_k$ になっているものだけをとりあげて，その所在番地を 定数$+h(d)+i\times h(c)$ の形に計算するようにコンパイルしようというわけである．手で書いたプログラムに比べて，普通の直接所在番地計算では 3.7 倍ぐらいの時間がかかるのに対し，この方法でやると 1.5 倍ぐらいですむ．

Goodwin: SEAL (Standard Electronic Accounting Language) の話．売上台帳の問題を例にとって説明．

Gardin: SYNTOL の文法．自動的文書化を強調する．syntagmatique と paradigmatique との区別を論じたが，何のことかよくわからなかった．

Moriguti: SIP の話と日本での ALGOL 処理プログラムの話．

Bosset: 小さい機械（PB 250——2,300 語）のための ALGOL——"Le MAGE"——の話．

Pacelli: Olivetti の ELEA 6001 のための言語 PALGO. equivalence 宣言に工夫をこらした由．

de Guenin: SIMPLE (=Simple Intuitive Multi-purpose Language for Everyone) の話．

Mazurkiewicz: **up** と **down** と二つの新しい記号を入れて，処理の水準を変えることにし，たとえば

begin D; D; S; S; S;
up begin……end down; ……

のようにすると，**up** から **down** までは「あとで実行

されるプログラム」に変えられるが，その他は compilation のときに実行されてしまうのである．この up……down は何重にも入れられる．要するに interlude の考え方を一般化して ALGOL に持ち込もうという提案である．

Humby: COBOL の簡略版——Rapidwrite. ICT の 1401 という小さい機械（高速記憶 400 語，ドラム 12,000 語，カード使用）のためのもの．「2 日間の勉強でプログラムが書けるようになりたい」というような人たちに向くやさしい COBOL である．

Gunn: Mercury から Orion へのプログラムの書きかえの問題，能率が良くないので困るとのこと．

Busa: PLANT（＝Program for Linguistic Analysis of Natural Texts）．7090 用．

Nuding: 計算機と計算機との間の情報交換のための言語．

Wegner: 大きい問題では，それを小さい部分問題に分割して，それらをそれぞれ適宜の言語でプログラムしてから全体を一つに組み上げるというやり方が重要であること．そのための FORTRAN での用意．subrontine 宣言に関する提案．

（昭和 37 年 7 月 17 日受付）

情　報　処　理　　May 1962

システム・シミュレーションのためのコムパイラーについて*

片　岡　信　二**

最近，大形電子計算機のプログラミングを簡略化するために，いろいろなコムパイラーがつくられておりますが，本日は，私が滞在しておりました，MIT で使われております，システム・シミュレーションのためのコムパイラー，ダイナモ（DYNAMO）および行列シミュレータ（General Singl-Server Queve-Simulation Compiler）について御紹介したいと思います．

1. ダイナモについて

ダイナモは MIT の Industrial Management の教授，W. Forrestor を中心とした，Industrial Dynamics のグループによって design されましたが，まず，その Industrial Dynamics について簡単にお話しすることにいたします．話を具体的にするために，工場，問屋，小売店からなる生産販売体系を考えてみましょう．

これらの間には，商品，お金，人，情報がたえず流れ，たがいに交互作用を行い，商品，お金のストック量は刻々に変化してゆきます．また，工場，問屋，小売店の三つのレベルで，おのおのの生産，在庫，販売，発注などの政策も，これらのシステムの流れ，ストック量を制御し，またそれらによって変更されます（フィードバック）．このような動的なシステムをシミュレートするに適したコムパイラーがダイナモであります．

このシステムの特性をきめるいくつかの要素の例を上げてみますと，まず第一に決定と行動における時間のおくれ（time lag）があります．大形の商品の場合，お客さんの注文があってから商品が手渡されるまでには，数日，数週間のおくれがあるのが普通ですが，これらを分析しますと，事務手続，郵便の配達，商品の輸送などのための時間があり，これらが三つのレベルで繰返されるときもあるので，おくれの分析だけでもかなり複雑なものになります．

つぎに政策の例を考えてみますと，あるレベル（たとえば小売店）で一つ上のレベルへ発注する場合，(a) 直ちに販売に結びつく発注，(b) そのレベルの在庫量を一定水準以上に保つための発注，などが考えられますが，お客からの注文量，手持ち在庫量などを考慮して発注量を決めます．そのための方法はいろいろありますが，ここでは政策のくわしい議論をするのが目的ではありませんので，このくらいにしておきます．

さて，以上のように時間的に変化してゆくシステムを，全体として把握して，その行動を追求して，さらに，外部，内部の要因の変化（たとえば，お客の注文量が突然 10% 増加したとか，ある機械が故障したとか，政策の変更とか）が，そのシステムにどのような

* On Compilers for System Simulation, by Shinji Kataoka (Hitotsubashi University)

** 一橋大学　情報処理学会第 2 回通常総会特別講演