새로운 軍用電算機言語 Ada의 紹介

工學博士 姜 麟 求
(金星通信 研究所長)

1 머 릿 말

本誌에서 이미 잠깐 紹介된 바와 같이[1], 美國防省은 軍用電算機의 성능향상과 開發費의 節減을 위해서 과거 수년간에 걸쳐 새로운 電算機의 言語를 개발해 왔으며 이 結實이 이 글에서 紹介하고자 하는 Ada[éɪdə]이다.

半導體의 발전에 따른 電算機의 획기적이고 급속한 性能의 발달과 價格의 相對的 低下는 軍用장비의 中樞神經을 電算機가 담당하도록 하고 있다. 이에 따라서 電算機를 유효하게 運用하게 하는 소프트웨어가 더욱 重要視될 뿐아니라 보다 쉽게 쓰고 복잡한 일을 하도록 하자니 電算機 機械값은 상대적으로 싸지고 있는 경향에 反하여 소프트웨어의 값이 더욱 비싸지고 있으며, 電算機開發費의 80%가 소프트웨어 개발에 所要된다고 보고 있다.

그러나 이러한 소프트웨어는 機械가 알아듣는 Binary(2進法)言語로는 작성할 수 없으므로 사람이 쓰기 쉬운 自然言語와 비슷한 言語를 사용해서 作成한후 電算機로 하여금 번역하도록 하고 있다.

이러한 人工言語를 프로그램言語, 또는 高水準電算言語라고 하며 그 數가 수십개가 되어 各各의 특징을 지니고 사용된다. 그림 1은 汎用電算用語의 발달과 복잡성 및 相互系譜를 보인 것이며 FOPTRAN 이상의 복잡한 言語를 高水準이라고 본다.

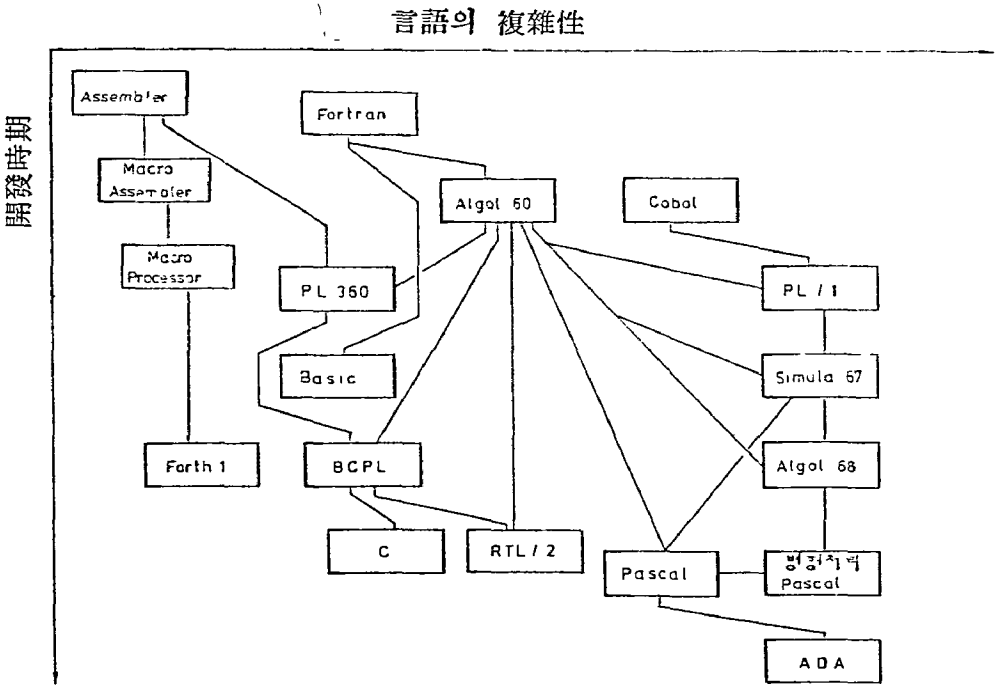
이러한 言語는 電算機種에 따라 약간의 差異, 즉 「사투리」가 있어서 한 機種에서 개발된 소프트웨어가 다른 機種에서 쓰일려며는 상당한 時間을 소비하여 번역해야 하고, 種類가 다른 言語를 사용한 경우의 번역은 더욱 번거러우며 같은 문제를 여러가지 言語로 푸는 일도 非一非再하였던 것이 實情이었다.

이는 소프트웨어 開發費用을 더욱 加重시키는 결과를 가저오므로 이의 改善을 위해 美國防省에서는 ARPA가 主動이 되어 1975年 1月에 作業班(HOLWG)를 발족시켜 三軍에서 共通으로 사용하는 프로그램用 言語를 檢討하기로 결정된다.

Ada는 바로 이 結實인데 數値電算 시스템 프로그램, 實時間處理와 併行處理가 필요한 응용프로그램 作成에 사용된다. Ada라는 이름은 19世紀의 計算機先驅者 Charles Babbage의 熱誠的인 同調者이며 그 自身 2進法의 創始者이기도 하고, 또한 유명한 英國詩人 바이론公의 외딸인 Ada Augusta Lovelace 令夫人의 이름을 따서 쓴 것이다.

2 開發의 발자취

HOLWG에서 제일먼저 着手한 것은 三軍을 비롯하여 産業體, 大學등의 호응을 받아 要求事項을 종합했으며 이를 一連의 要求示方書로 엮었는데 年次的으로 보면, 1975年에 Strawman (짚사람이란 뜻인데 흔히 초기의 要求示方書를 이렇게 부른다)과 Woodenman(木人)이 작성되었고 順次로 改訂되어 1976年 6月에 Tinman (錫人), 1977年 1月에 Ironman(鐵人), 同年 7月에 改訂版鐵人, 그리고 1978年 6月에 Steelman



〈그림 1〉 汎用電算用 言語[4]

(鋼人)이 各各 발표되었다[18].

이와 併行하여 1976年에 錫人을 기준으로 하여 旣存言語 FORTRAN, ALGOL 60, 68 PL/L, PASCAL 등 26種을 總點檢한 바있으며[5], 1977年에 당시 示方(錫人)을 만족시킬 만한 旣存言語는 없으며 새로운 言語을 개발하되 新言語의 기초로는 PASCAL, ALGOL 68 혹은 PL/I 중에서 택일하는 것이 좋겠다는 結論을 얻었다.

1977年 5月에 鐵人示方에 따라 16個의 提案書를 받아 평가한 후 豫備設計段階에서 4個의 業體를 골라 1978年 2月까지 6個月間 개발시켰다[6]. 4個業體란 CII-Honeywell-Bull(佛), Intermetrics, Stanford 硏究所 및 Sof Tech이 었고 評價의 公正을 기하기 위하여 各社가 개발하는 言語를 각각 綠, 赤, 黃, 靑色言語라고 公稱되었다.

네言語가 다 PASCAL을 기초로 한다고 提議되어 있었으나 豫備設計의 결과는 판이한 차이가 있었으며 약 2個月間 80個 이상의 評價機關의 평가를 통하여 長短點이 면밀히 檢討되었다[10, 11, 12, 13, 14]. 완전히 만족할 만한 것은 못되었으나 그 중에서 Honeywell의 綠色言語와 Intermetric社의 赤色言語가 선택되어 1年間 本設計에 들어갔다.

1979年 3月에 예정대로 設計가 완성되었고 50個 이상의 分析팀에 의한 精密分析을 거쳐 그해 5月에 綠色言語가 정식으로 채택되어 Ada라고 命名되었다. 1976年에는 美國電算協會(ACM)의 특집으로 Ada言語의 設計理論[16]과 解說書[15]가 발간되었다.

그후 Ada는 광범한 技術的 試驗評價를 거치는 한편 100個 이상 機關에서 응용프로그램을 作成하여 사용해 보면서 評價한 결과를 1979年 10月에 검토한후 1980年初에 改訂版이 작성되었다. 이 言語를 機械語로 번역하는 콤파이러는 이미 1979年 8月에 첫 試作品이 可用되었으나 Carnegiemellon大學, 뉴욕大學, 부라운大學 등 敎育機關과 各軍에서 보다 능률적인 콤파이러 試作에 熱을 올리고 있어서 금년내에는 완성될 전망이다

Ada를 設計한 CII-Honeywell-Bull의 팀은 프

랑스 파리에 있었고, Jean Ichbiah가 主導하였다.

한편, Ada支援環境의 要求示方을 定義하는 Sandman, Pebblman이 발표되어 있고 1979年末에는 改訂版이 발표되었다.

美國防省에서는 비교적 저렴한 價格으로 Ada를 普及할 예정이고 韓國에서도 Honeywell이 國內業體와 손잡고 營業中이므로 이 言語의 國內普及도 빨리 실현될 가능성이 있다.

다음 節에서 Ada의 性格을 紹介하기 전에 이의 母體인 PASCAL에 대하여 간단히 紹介하고 넘어가기로 하겠다.

PASCAL은 프로그램技法을 어떤 명확한 基本槪念에 입각해서 체계적인 學問으로 敎育시키고저 한 基本目標를 달성하는 道具로서 동시에 이를 實用化하기 위해서 개발된 것이다[3].

그림 1에서 볼수 있듯이 PASCAL의 基盤은 ALGOL 60인데 ALGOL 60은 學問用 言語로서 ACM學術誌의 標準言語로 쓰이기는 했으나 널리 實用化되지는 못하였고 PASCAL은 ALGOL 60의 약점이었든 데이터構成方式을 대폭 보강한 것이다.

PASCAL은 1968年에 첫선을 보였고 1973年에 改訂된 바있다. 지금은 CDC등 大型부터 DEC등 小型까지 널리 普及돼 있으며 특히 그 一種인 PASCAL USCD는 近來 爆發的 需要가 있는 마이크로 콤퓨터의 言語로서 널리 사용되고 있다.

3 Ada의 特徵

프로그램은 크게 데이터의 構成을 定義하는 宣言部分과 데이터의 처리를 지시하는 實行部分으로 나누어지는데 Ada는 PASCAL이나 마찬가지로 이 區分이 명확하며 實行部分은 Begin으로 시작 End(手續이름)으로 끝나는데 手續이름에 있어 區分이 더 明白하다. 例示프로그램에서 SUM-VSUM이 手續이름이다.

宣言部分에서 모든 變數를 定義하는 點은 PASCAL과 같고 그 이름에 따라 變數나 函數의 形(代表하는 數의 性格)이 결정되는 法이 없는 것이 FORTRAN과는 큰 차이점이다.

Ada에서는 콤파일單位를 크게 둘로 나누어 副프로그램과 모듈이 있고 副프로그램은 다시 手續과 函數로 나눈다. 이것이 PASCAL에서 잘 알려진 데이터處理(즉 演算)의 部分이며, 모듈은 다시 Package와 Task로 區分되는데 Package는 副프로그램과 데이터가 합쳐진 것을, Task는 併行처리할 수 있는 構造를 가진 프로그램을 말한다.

Task에서는 Ada가 효과적으로 併行處理를 하게끔 랑데부(Rendezvous—相逢)라는 節次가 있는데 이는 두 Task 사이에 데이터의 處理委任, 交換이 필요할때 呼出하는 쪽과 받는 쪽을 同期化하는 節次이며 이것이 끝나면 각 Task는 獨自로 나머지 實行을 속개한다. 이러한 併行處理에 대한 構造가 Ada의 하나의 특색이다.

프로그램을 作成함에 있어서는 쌓아올리는(Bottom-up) 방법과 풀어내려가는(TOP-down) 방법이 있는데 다른 言語는 主프로그램에 사용할 副프로그램이 미리 確定되어야 하므로 풀어내려가는 방법을 쓰기에 不便한 反面 Ada는 모듈이 獨立的으로 構成되어 큰 프로그램의 일부가 되도록 되어 있으므로 편리하다.

특히 大規模의 프로그램 作成에는 시스템工學的으로 보아서 풀어내려가는 방법이 必須的이므로 그 利點이 더욱 돋보인다.

또한 示方宣言과 그 實現은 分離할 수 있는데 例를 들자면 例示프로그램에서 Function VSUM만 定義하고 그뒤의 End VSUM까지는 뒤에 分離해서 처리할 수 있는 裝置가 있다는 뜻이다.

이와 관련해서 手續名, 變數名, 演算記號 등에 있어서 동일한 이름이 공통된 宣言有效範圍內에 있으면서도 두가지 以上의 다른 뜻을 가질 수 있게 하되 실현시에 一義的으로 확정할 수만 있으면 허가되고 있다. 이 性質을 過負荷(Over Loading)이라고 한다.

데이터의 形은 기본적으로는 整數(INTEGER) 實數(FLOAT), 불수(BOOLEAN) 및 文字이나 프로그램作成者가 定義할 수 있는 方法이 다양해서 그 범위를 定義할 수 있을뿐 아니라 例를 들어 砲의 수와 戰車의 수가 같은 整數形이지만 구분해서 演算하고 이의 混同을 막고 싶으면 따로 定義하면 되고 의도적으로 하지않는 限 混同演算이 不可하게 하는 장치가 있다.

例示 프로그램

(Ada)

```
Procedure SUM-VSUM is···a procedure called SUM
                                        VSUM
type VECTOR is arry(1..50) of INTEGER, ···define
                                        Vector type
V_AMT, V_SLT : VECTOR,
SUMAS_1, SUMAS_2 : INTEGER,
function VSUM(N : INTEGER, A : VECTOR)
  return INTEGER is
  SUM : INTEGER : =0, ···define SUM & Set
                        to zero begin
for I in 1..N loop
  SUM : =SUM+A (I);
end loop,                    ·····end of loop
return SUM,
end VSUM,                    ·····end of function
begin                ·····beginning of statement part
  SUMAS_1 : =VSUM (25, V_AMT)+VSUM(15,
            V_SLT),
  SUMAS_2 : =VSUM (30, V_SLT)+VSUM(40,
            V_AMT),
end SUM_VSUM,                ·····end of procedure
```

(PASCAL)

```
Program sumvsum

type vector : array(1..50) of integer;

var vamt, vslt : array (1..50) of integer,
  sumas 1, sumas 2 : integer;

function vsum (n, integer, a : vector) : integer,

var sum : integer,

begin sum : =0,
for i : =1 to n do          sum : =sum+a (i),



end : {end of function vsum}
begin
  sumas 1 : =vsum (25, vamt)+vsvm(15, vslt),

  sumas 2 : =vsum (30, vslt)+vsum (40, vamt),

end : {end of program sum v sum}
```

또 한편으로는 어떤 프로그램은 變數의 形을 초월한 일반적 프로그램으로 構成하고저 하면 콤파일때 그 形을 정하는 Generic(屬性)의 方式에 의해서 가능하다.

이와 같은 多樣性 때문에 固有用語의 數도 62個나 되며 PASCAL의 35個에 비해 거의 倍나 된다. 이 固有用語는 混同을 피하기 위해서 수속이나 變數등의 이름(Identifier)으로 쓸수 없다.

PASCAL과 比較하기 위하여 두개의 數列 VAMT와 VSLT의 部分 合計를 다시 합쳐서 SUMAS 1과 SUMAS 2를 計算하는 프로그램을 例示했는데 數列의 部分 合計는 VSUM이란 Function을 이용했다.

두 프로그램을 比較해 보면 그 構成에 있어서 닮았으면서도 差異가 있는 것을 알수 있을 것이다. 몇가지 눈에 띠는 것을 說明하면 Ada에서는 固有用語外는 大文字를 쓰는 反面 PASCAL은 全部 小文字이며 End 다음에 무엇이 끝났다는 것을 Ada는 分明히 하고 있다.

또 Ada는 is라는 固有用語를 사용해서 ':=' 의 뜻이 더 狹義로 分明히 했는데 이는 FORTRAN에서 "="가 ":=", "=" 및 "is" 등을 전부 뜻하고 있는 것과 對照가 된다.

또한 說明文은 "— —"로 시작해서 文尾까지 쓰기 때문에 Assembler의 境遇와 비슷하며 PASCAL이 ( )안에 쓰는 방식이나 FORTRAN에서 C로 區分해서 쓰는 방식보다 쓰기와 읽기가 便利하다. 읽기 便利하게 한 點에서는 밑줄 (___)을 이름 사이나 수자사이에 許用한 것도 特色이라고 하겠다.

4. 맺 음 말

以上 간단히 Ada를 紹介하였다. 앞으로의 兵器의 電算化는 필연적인 것으로 보면 소프트웨어의 重要性은 再論할 필요가 없을것이다. Ada가 美國防省의 標準電算言語로 쓰이게 되므로

이에 대한 理解는 電子兵器運用에 도움이 될뿐 아니라 새로운 兵器開發에 可用한 소프트웨어의 빠른 理解로 시간과 노력이 節約될 수 있을뿐 아니라 한걸음 더 나가서 이 새로운 言語가 誘發할 막대한 소프트웨어 開發 市場에 진출해 보는 것도 可能하게 하지 않겠는가 하는 期待도 가져볼 수 있겠다. 多幸히 수소문한 바에 의하면 KAIST서에서 關心을 갖고 공부하고 있다하며 참고문헌中 두가지 〔2, 19〕가 國內에서 사진판으로 入手可能하다.

이 글이 Ada에 대한 關心을 불어 이르키는데 도움이 되었으면 하는 작은 所望에서 시작되었음을 말씀드리며 이 글이 Ada의 전모를 소개하기에는 너무 貧弱함을 自認하면서 펜을 놓는다.

참 고 문 헌

1. 姜麟求 '美國의 80年代 研究開發 方向' 國防과 技術 80/7 p. 9.
2. P Wegner 'Programming Witn Ada : An Introduction by Means of Graduated Examples Prentice-Hall 1980.
3. K. Jensen & N. Writh 'PASCAL User Manual and Report' Springer-Verlag 1974.
4. S. C. Reghizzi 等 諸氏 'A Survey of Microprocessor Languages' IEEE Computer 80/1 p. 48~66.
5. L.C. William & A. Whitaker 'A Defense View of Software Engineering' Int Conf Software Engr 2nd 1976 p. 358~362.
6. W.A. Whitaker 'The US Department of Defense Common High Order Language Effovt' SIGPLAN NOT 13 〔2〕 1978, p. 19~29.
7. D. A. Fisher 'The Common Programming Language Effort of the Department of Defense' AIAA, NASA, IEEE ACM Comp Aerosp Conf, 1977, p. 297~307.
8. T.E. Cheatham 'Programming Language Design Issues' Lact. Notes Computer Science 〔54〕 1977, p. 399~435.
9. 'Department of Defense Requirements for High Order Computer Programming Larguages' SIGPLAN Not 12 〔12〕 1977 p. 39~54.
10. E.W. Dijkstra 'Do D-I : The Summing Up SIGPLAL Not 13 〔7〕 1978 p. 21~26.
11. 同人 'On the BLUE Language Submitted to the DOD' SIGPLAN Not 13 〔10〕 1978, p. 10~15.
12. 同人 'On the GREEN Language Submitted to the DOD' 同誌 p. 16~21.
13. 同人 'On the YELLOW Language Submitted to the DOD' 同誌 p. 22~26.
14. 同人 'On the RED language Submitted to the DOD' 同誌 p. 27~32.
15. 'Preliminary ADA Reference Manual' SIGPLAN Not 14 〔6—4〕 1979.
16. J.D. Ichibiah等 諸氏 'Rationale for the Design of the ADA Programming Language' SIGPLAN Not 14 〔6—B〕 1979.
17. B. Knobe 'Flight Languages ADA vs HAL/S' AIAA Computer Aerosp Conf 2nd 1979 p. 345~351.
18. 德田雄洋 'Ada 實現에 關한 問題點에 關하여 情報處理 80/3 p. 226~232.
19. 'Reference Manal for the ADA Programming Language'
20. J.M. Rosenberg 'The Computer Prophets' Ch. 3, Macmillan, 1969.

◇ ◇ ◇