BERNARD ROBINET

Ingenieur de Recherche
Institut de Programmation
Universite de Paris

Preface de

J. ARSAC
Professeur a la Faculte des Sciences de Paris

le langage APL

1971

ÉDITIONS TECHNIP • 27 RUE GINOUX • PARIS 15e

technip

Ce livre a été écrit à l'occasion d'une étude entreprise pour le compte de la Délégation Générale à la Recherche Scientifique et Technique (DGRST).
Nous tenons à remercier ici cet organisme d'en avoir autorisé la publication.

R. ROBINET

PRÉFACE

J'ai été très touché quand mon collègue Bernard ROBINET m'a demandé une préface pour son ouvrage sur APL. La position que j'ai prise quant à la science informatique est fortement contestée dans l'Université, et cette préface ne sera pas pour tous une invitation à lire l'ouvrage, ce que je déplore car je le trouve remarquable. Aussi, je tiens à lui remercier de la confiance et de la fidélité qu'il m'a ainsi manifestées, sachant fort bien les risques qu'il courait.

Je dois aussi avouer l'embarras dans lequel cette preface, et plus profondément toute l'action de B. ROBINET en faveur d'APL me plonge. J'ai fait connaissance du langage APL (puis-je employer un article défini avec APL, initiales de "A Programming Language" ?) au laboratoire de recherches Watson de la compagnie IBM, dans la banlieue de New-York. Kenneth IVERSON lui-même nous fit une brillante démonstration des possibilités de ce langage. Mais le contraire eut été étonnant : si l'auteur n'est pas passionné par ce qu'il a créé, qui donc le sera ? Qui pouvait être mieux placé que lui pour en connaître tous les raffinements ? Que, dans le même laboratoire, la plupart des chercheurs utilisent APL, cela se concevait aussi.

J'ai rencontré de nouveau APL à l'Université d'Edmonton, à quelques 300 km à l'est des Rocheuses canadiennes, si belles dans les parcs de Banff et de Jasper. J'y ai retrouvé le même enthousiasme, le même prosélytisme. Chacun essayait de me convaincre des beautés du langage. On l'utilisait pour l'enseignement des statistiques (K. SMILLIE et D. Von MAYDELL). Il était à la base des expériences d'aide à l'enseignement de S. PUNKA.

Dans ce livre, B. ROBINET se demande si "l'APLite est une maladie". On a en effet comme l'impression d'une contagion d'APL. Le langage se répand, soulevant les plus vives critiques au passage, mais faisant de plus en plus de fanatiques. Les interpréteurs se multiplient sur les matériels les plus divers, gros ou petits ordinateurs. La réalisation de B. ROBINET sur la machine de faible puissance Elliott 4130 ouvre d'ailleurs la voie d'APL sur une gamme extrêmement étendue d'ordinateurs.

En fait, APL me ferait plutôt penser à une drogue. On s'en méfie, on en dit du mal sans la connaître. Puis un jour on y goûte. On risque fort d'être pris et de ne plus pouvoir s'en passer. Ou encore, je vois APL comme l'idole d'une nouvelle religion. On se fait initier à son langage et à ses rites, et l'on devient un prosélyte. Dire du mal d'APL est un sacrilège, le soupçonner de porter des ambiguïtés un péché très grave (Qui donc absoudra B. ROBINET de l'avoir commis ?). Utiliser d'autres langages est une infidélité !

Il m'étonnerait beaucoup que le lecteur de ce livre ne soit pas pris sous le charme de l'idole. La présentation de B. ROBINET est remarquable à tous points de vue, ses dons pédagogiques subsistent dans ces pages à une lecture facile et reposante.

Et pourtant il me reste une dernière hésitation à vous encourager à la lecture. Si l'emploi d'APL doit se généraliser, il faudra modifier beaucoup de choses en informatique. Les structures que manipule ce langage sont facilement complexes, et n'incitent pas du tout à l'économie des moyens. Un emploi extensif amènera à des modifications dans le câblage même des machines. C'est ce que Pecqueur vient d'exposer au congrès de l'AFIF à Ljubljana. L'enseignement devra lui aussi se réformer : finies les petites boucles et les schémas récursifs, et vive les programmes denses : "Oui, le langage APL est comme cela, il dit beaucoup en peu de paroles". L'exposé algorithmique recule au profit de la notation algébrique. Y gagnera-t-on ?

C'est une question que je me pose très sérieusement, et dont je ne connais malheureusement pas la réponse. Alors que faire ? Garder APL sous le boisseau, en parler le moins possible ? C'est peut-être ouvrir un outil essentiel à de nombreux utilisateurs de l'informatique, et en tout premier lieu les chercheurs et ingénieurs amenés à faire de nombreux calculs de moyenne importance et non répétitifs, qui ne justifient pas un gros effort de programmation; pour eux, APL est certainement la meilleure solution.

On me dira au demeurant que je prends bien au sérieux l'ouvrage de B. ROBINET que s'il ne le publiait pas, un autre ferait quelque chose de comparable (mais peut-être pas aussi facile à lire, c'est l'un des ouvrages en anglais sur le sujet, je ne les crois pas comparables pour la valeur pédagogique), ou il ne faut pas attacher une telle importance à nos actes personnels, et que le devenir scientifique est tout tracé, déterminé, insensible à nos pauvres options individuelles.

Je ne saurais admettre un tel déterminisme. Si je ne croyais pas à la valeur des actes de chacun, j'aurais depuis longtemps cessé d'agir et de me battre au sein de l'Université pour le développement et la promotion de l'informatique. Quel pourrait être le sens d'un combat dont l'issue est certaine, indépendante de mes actes ? J'ai besoin de croire à leur valeur pour pouvoir les passer encore en 1977, après les années noires que nous venons de vivre. Et j'ai l'outrecuidance de penser que l'histoire du développement scientifique eut été changée si chacun de ceux qui y ont participé ne s'étaient pas trouvé au bon moment à la bonne place. Non pas seulement les génies comme Galilée, Newton, Pascal... mais aussi les obscurs dont l'histoire n'a pas toujours retenu le nom, qui ont patiemment collecté les faits sur lesquels la science élabore ses théories, qui ont fourni à ces grands noms la contre-réaction dont ils avaient besoin pour l'élaboration de leurs idées. "Si le nez de Cléopâtre..." ne s'est pas vrai que pour l'histoire des civilisations.

C'est en pensant à tout cela que je recommande vivement la lecture de cet ouvrage. Le lecteur succombera-t-il à l'idole ? Sans que cela en devienne une maladie, je lui souhaite d'apprécier APL, car c'est certainement quelque chose de très joli. Ce livre contribuera-t-il au développement de l'"APLite", permettra-t-il au contraire de démystifier APL pour en faire ce que c'est réellement, un beau langage avec ses limites ? L'avenir le dira. Ce que je souhaite, ce dont je suis sûr, c'est que ce livre ne passera pas inaperçu.

J. ARSAC
Professeur à la Faculté des Sciences de Paris.

TABLE DES MATIÈRES

INTRODUCTION

" Intelligente pauca "

Sans échanges d'informations, il ne peut y avoir de progrès, d'avancement dans la connaissance et plus particulièrement dans les domaines scientifiques et techniques. Ces échanges ne peuvent se faire qu'en employant des langages de précision et d'exactitude maximales et, si les discours tenus sont parfois trop longs, on use d'abréviations possédant les mêmes qualités.

Ces échanges d'informations se font généralement d'homme à homme - c'est ce que nous faisons en ce moment - mais aussi d'homme à machine ; cette situation, vieille comme la machine, a été particulièrement révélée par l'introduction de l'ordinateur dans notre monde. Dans pareil contexte, l'emploi de langages de programmation suppose qu'ils obéissent eux aussi aux nécessités évoquées plus haut.

C'est à ces nécessités, certains les nommeront contraintes, qu'est due la naissance du premier langage de programmation digne de ce nom :

le FORTRAN ; une prise de conscience plus nette des diverses propriétés d'un langage non ambigu conduisit à la conception puis au perfectionnement incessant du langage ALGOL qui est devenu, ainsi, un langage fort élaboré, d'Algol 60 à Algol 68 en passant, sans toujours s'arrêter, par Algol W, Algol X et autres.

La prise de conscience de ces caractéristiques et le développement de nouveaux outils de conception ont facilité la création de nombreux langages de programmation : langages plus ou moins spécialisés, de puissance plus ou moins grande, le problème de leur implémentation en a souvent abrégé les jours. Néanmoins, malgré ou à cause de cette croissance anarchique, on trouve peu de langages aptes, à la fois, à la formulation mathématique usuelle, susceptibles de décrire des systèmes ainsi que des traitements.

Il faut créditer le Professeur AIKEN d'avoir, l'un des premiers, perçu cet état de chose et d'avoir mis sur pied, il y a une quinzaine d'années, une équipe remarquable animée par K.E. IVERSON, laquelle a développé un langage connu sous le nom de l'animateur.

Mais, pour beaucoup de langages puissants, il y a souvent conflit entre les intentions de l'auteur et les possibilités d'implémentation : le conflit n'est pas toujours d'ordre conceptuel, mais plutôt d'ordre technologique. Il faut, en effet, trouver un compromis entre la puissance du langage et les contraintes apportées par les ordinateurs - place en mémoire, temps d'exécution -. C'est ainsi que K.E. Iverson et A.D. Falkoff

ont développé un langage de programmation nommé APL pour " A programming Language ", titre de l'ouvrage qui présenta pour la première fois, et d'une manière complète, le langage d'Iverson [IV1] ; APL est en fait un sous-ensemble simplifié et compilable du langage d'Iverson. Beaucoup d'expériences d'implémentation du langage APL ont et ont eu lieu ; en particulier, la première version expérimentale fut pour l'ordinateur IBM 7090 [IV2] ; actuellement, la version d'APL la plus utilisée est celle du " APL 360 Terminal System " qui est une implémentation du langage sous le système IBM/360 [FALIV] .

Afin que cet ouvrage réalise, et nous le souhaitons, sa vocation de cours de langage, notre choix s'est donc porté sur APL\360, le langage que l'on rencontre, maintenant usuellement, sur l'ordinateur IBM/360.

Dans cette version, le langage APL est utilisable sous deux modes ; l'un est nommé "mode exécution" et l'autre est nommé "mode définition" : de toute façon, quel que soit le mode d'utilisation du système APL, on définit des traitements à partie d'opérateurs de base appliqués à des objets convenables. C'est sur ce point que nous avons insisté, considérant l'aspect système comme trop contingent à un équipement.

Cet ouvrage est donc composé principalement de deux parties.

Dans la première, intitulée "APL PAR L'EXEMPLE", l'intention qui y préside est de fournir une introduction au langage APL et à son emploi dans la création ou l'emploi d'algorithmes pour des problèmes d'origines fort diverses. Aussi plusieurs opérateurs ne sont pas du tout mentionnés

et quelques autres ne sont pas explicités dans leur entière généralité ; il est néanmoins souhaité que cette présentation suffise pour faire prendre le virus APL au lecteur.

Dans la seconde partie, intitulée "SYSTEMATIQUE D'APL", l'effort a été porté sur la syntaxe et la sémantique du langage : le lecteur rencontrera là une description systématique des opérateurs usuellement implémentés.

C'est dans les ANNEXES que le lecteur trouvera une description sommaire du " APL\360 Terminal System " et toutes références nécessaires à de futurs travaux pratiques.

Le langage APL est encore méconnu en Europe et nous souhaitons que cet ouvrage contribue à faire reculer cette ignorance . Sans les Professeurs ARSAC et GIRAULT de l'Institut de Programmation, ardents promoteurs d'APL, cet ouvrage n'aurait pu naître ; sans l'aide et les suggestions de Mrs ARLETTAZ, GIRARD et MICHEL, il n'aurait pu être mené à bien.