16-11-67

Programming Language Design

Principles and Practice

Contents

probably never submitted

Total length: about 60,000 words.
(could be reduced to 40,000 by omission of the appendix, or expanded by inclusion of other material).

Note: The Record Handling chapter is quite different from the Grenoble lectures.

Programming Language Design

~~Principles and Practice.~~

C.A.R. Hoare

Introduction

A programming language may validly be regarded as nothing more nor less than tool for describing programs; and the design of good programming languages must therefore be firmly based on standards of craftsmanship and excellence in the same way as the design of tools intended for other purposes or trades. The main requirements of a designer of tools is:

1. A clear recognition of the purpose for which he is designing.
2. A deep understanding of the criteria by which the design is to be judged.
3. A painstaking search for techniques and expedients which will satisfy the criteria to the highest degree, by reconciling any conflicts between the criteria, or by choosing a successful compromise where this is not possible.

Each ~~paper~~ chapter in this ~~collection~~ book illustrates the connection between purpose, criteria, and techniques in the design of features and facilities in general-purpose high-level programming languages.

This ~~collection of papers~~ book does not put forward just another proposal for yet another programming language. Instead, it concentrates on selected problems of programming language design, ~~and~~ shows how a consistent and rational approach to these problems can lead to solutions which are technically sound. The proposed solutions themselves are usually described in terms of ALGOL 60, but they are equally applicable to any programming language, whether special -purpose or general-purpose, high-level or low-level, which intends to cover the relevant problem area.

discusses the solutions proposed in various programming languages, such as FORTRAN, COBOL, PL/I, and ALGOL. Finally, it

Thus the ~~papers~~ book will be of interest to all specialists engaged on programming language design in any field; it will also be useful to all programmers looking for clear and efficient methods of expressing themselves, and will serve as a guide in the rational assessment and choice of high-quality programming languages, when such a choice is available. The primary objective of the collection will have been achieved if it leads to a wider appreciation of the factors which contribute to quality in their programminglanguages, both on the part of their designers and their users.

The papers of this collection have been reprinted from various sources, mainly the ALGOL Bulletin. They represent the development of the author's thinking on the subject over a period of several years. Over that period, certain of the ideas expressed have changed, or have been superseded by ones which are believed to be better. However, the earlier papers have deliverately been left as they stand, and the incompatibilities have been pointed out in an accompanying note. The reason for avoiding changes to the text is that the papers as they stand will better convey the background and philosophy which underly the search for solutions if they also contain the various blind alleys which were explored and later abandoned. The resulting inconsistencies of detail in no way obscure the basic unity of approach displayed in all the papers. An understanding of the principles of the approach will be more valuable to a language designer and user than any slavish acceptance of the details of the solutions proposed.

The papers in the collection are printed in chronological order, but they may be read in any sequence, except that the papers on Record Handling (numbers 5 to 8) are best studied in the sequence in which they are printed. Some familiarity with ALGOL 60 will assist in the understanding of the examples quoted.

The ~~first~~ second chapter presents a solution to problems posed by anomalous position of switches in ALGOL 60, and theassigned and computed go to in FORTRAN. The solution avoids the pitfalls and inefficiencies of the PL/I assigned labels. The second paper deals with the inefficiencies involved in the ALGOL for statement, by proposing a definition very similar to that which assists optimisation in FORTRAN. The third paper clears up the difficulties associated with the ALGOL name parameter and the FORTRAN and PL/I "address" parameter mechanism, by suggesting a slight extension of the highly successful technique of the ALGOL value parameter.

The fifth paper introduces the concepts of Record Handling, which underly most applications dealing with structured data. ~~Further remarks on the same topic are contained in the next paper.~~ The ~~seventh paper~~ shows how record handling concepts can be applied to business-oriented applications involving the processing of high-volume files.

The eighth paper makes a recommendation for the use of backing stores for record handling in cases where the serial access provided by files is inadequate.

The ninth paper presents the correct method of dealing inside a computer with finite sets of objects. It is intended for applications in which the operations of set algebra are used heavily inside loops, and in which high volumes of data are to be processed. The tenth paper attempts a solution to the vexed twin problems of Subscript Checking and Subscript optimisation. It shows that by good language design one can achieve the security of checking together the efficiency of optimisation, without excessive complexity in a compiler. The last paper deals with the simple problem of using a variable before assigning to it, and gives a simple and obvious solution which has long been unrecognised.

The appendix illustrates the application of the principles of good programming language design to the design of a complete language, rather than just to the selected areas treated in the previous papers. It incorporates many of the ideas expressed in the earlier papers, but omits all proposals whose correctness was subject to doubt from either of the authors.

Acknowledgments

The following acknowledgments are given with pleasure;

to the Editor of the ALGOL Bulletin

for permission to publish papers 1,2,3,4,5,6 and 7.

to the Editors of the Communications of the ACM and to Dr. N. Wirth

for permission to publish the Appendix

to the Editors of the Proceedings of IFIP '68

for permission to publish paper 8

to Elliott Automation Computers Limited

for support of the entire work and for permission

to publish it.