FINAL REPORT

on SRC Research Project Into Program Proving

As mentioned in the first annual report, activity on the project during the second year has been proceeding at a lower intensity than in the first. Nevertheless, a useful amount of work has been done, and all the major objectives of the research project have been met.

Firstly, a clearer understanding has been reached on the relationship between proof-oriented methods of formal definition of programming languages, and other methods currently proposed. It has been possible [1] to use several different definition techniques for the same simple language, and to prove that they are consistent. This has thrown light on the relationship between definition methods; and since the more constructive definition methods approximate to an (abstract) implementation of the language, the proofs may perhaps serve as a guide for the proof of the "correctness" of more concrete implementations of the language in question.

During the year Dr. Clint has concentrated mainly on proof techniques appropriate for dealing with"coroutines". The SIMULA 67 language was taken as a basis for this work, but it was necessary to make considerable simplification in order to find reasonably simple proof techniques [2].

Proof techniques for programs involving parallelism have been given in [3] ; their discovery was greatly assisted by Dr. Clint's interest and participation. These have been found highly useful in [4]. Unfortunately, the "conditional critical region" concept, thoughit appears to be quite appropriate to a high-level language for "user" programs, does not appear to be so suitable for formulating scheduling policies for overloaded resources, which is the main problem encountered in operating system design. For this purpose, the proposal for monitors made in [5] appears more appropriate; though the associated proof techniques would appear to be much more intricate.

The final goal of the Research Project was the elucidation of proof techniques for a complete programming language suitable for the construction of software programs. This has now been accomplished. The programming language PASCAL has been clearly proved to be suitable, at least for writing compilers; and it has now received an axiomatic definiton [6]. It is hoped now to find out how far the language (possibly an extended version) is suitable for the construction of operating systems.

The ultimate objective of constructing a complete software system completely proven still seems a long way off. Apart from the development of more powerful proof techniques (possibly involving machine assistance) to remove some of the tedious handling of detail, there is the need to train a new generation of computer scientists with the skill, the interest, and the professional pride to contribute

to such a project. Nevertheless, there can be no doubt that the elucidation of program proving techniques will have a longterm influence on the ways in which programs are designed and developed, the high-level languages in which they are coded, and the general reliability of the end product.

My sincere thanks are due to the S.R.C. and to my colleagues on this project (as well as critics outside it) for supporting a project with such ambitious objectives, and bringing it to such a successful conclusion.

REFERENCES

1 P.E. Lauer. Consistent Formal Theories of the Semantics of Programming Languages. IBM Research Laboratories, Vienna.

2 M. Clint. Program Proving: Coroutines. (privately circulated).

3 C.A.R. Hoare. Towards a Theory of Parallel Programming, submitted to Belfast Symposium on Operating System Techniques.

4 P. Brinch Hansen. A Comparison of two Synchronising Concepts. Submitted to Acta Informatica.

5 C.A.R. Hoare. Towards a Theory of Parallel Programming. (preliminary draft). (privately circulated).

6 C.A.R. Hoare. An Axiomatic Definition of the Programming Language PASCAL. (available in draft).