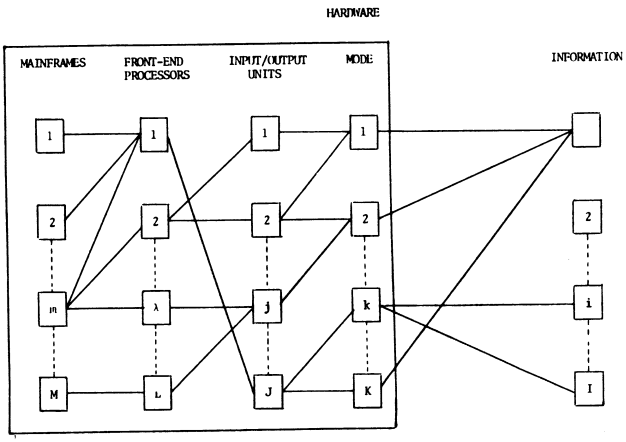

Fig. 2 General multi-unit system with sample connections

B_λ —maximum number of characters per second through processor type λ.

$c_{ijk\lambda m}$—average total time for a query of type i to be processed along the route j—k—λ—m through the system.

$n_{k\lambda}$ —the number of lines available into front end processor λ in mode k.

$D_{i\lambda}$ —the required minimum number of processed queries per hour of type i through input/output device j.

u —utilisation of lines.

The problem statement is then:

$$\text{Max } Z = \sum_{i,j,k,\lambda,m} x_{ijk\lambda m}$$

$$\text{Subject to: } \sum_{i,j,k,\lambda} a_{im}\, x_{ijk\lambda m} \leqslant 3{,}600 \quad \text{all } m$$

$$\sum_{i,j,k,m} b_i\, x_{ijk\lambda m} \leqslant 3{,}600\, B_\lambda \quad \text{all } \lambda$$

$$\sum_{i,j,m} c_{ijk\lambda m}\, x_{ijk\lambda m} \leqslant 3{,}600.u.n_{k\lambda} \quad \text{all } k, \lambda$$

$$\sum_{k,\lambda,m} x_{ijk\lambda m} \leqslant D_{ij} \quad \text{all } i, i$$

$$x_{ijk\lambda m} \geqslant 0 \quad \text{all } i, j, k, \lambda, m$$

References

GORDON, W. J. and NEWELL, G. F. (1967). Closed Queuing Systems with Exponential Servers, *Operational Research*, Vol. 15, pp. 254-265.

LANGEFORS, B. (1970). *Theoretical Analysis of Information Systems*, Student Litteratur, Lund, Sweden.

MOORE, S. C. (1975). Approximating the Behaviour of Non-Stationary Queues, *Operational Research*, Vol. 23, pp. 1011-1032.

PACK, S. D. (1975). The Output of an M/D/1 Queue, *Operational Research*, Vol. 23, pp. 750-760.

Book reviews

The Architecture of Concurrent Programs, by Per Brinch Hansen, 1977; 317 pages. (*Prentice Hall*, £13·55)

The programming language Concurrent Pascal is by now fairly well known, and has generated a lot of interest, as perhaps indicated by the official distribution of its implementation to 252 institutes. This book is concerned with the use of Concurrent Pascal in developing reliable concurrent programs and is divided into nine main sections. The first section discusses design principles, with Brinch Hansen presenting his requirements of the qualities a program should exhibit. Sections 2 and 4 provide a formal introduction of the main concepts and language notation of Concurrent Pascal (namely, processes, monitors and classes). Section 3 gives a short overview of Sequential Pascal, which is in the main identical to Wirth's Pascal, and can be used to program other system utilities (e.g. the compilers).

Sections 5 to 7 present three complete examples of working model operating systems written in Concurrent Pascal. As the text of these programs is included, these sections are rather long and form approximately half of the book.

Section 8 provides the definition of Concurrent Pascal, together with the additional restrictions and extensions of the implementation that was distributed for the PDP11/45. As the compiler in this implementation generates code for a virtual machine, Section 9 describes the interpretive implementation of this virtual machine on the PDP11/45.

The book is written in the usual self congratulatory fashion of Brinch Hansen. The informed reader will have seen already several sections of this book published elsewhere. While the book claims to present a systematic way of developing reliable concurrent programs, this is illustrated solely through the use of Concurrent Pascal. It is clear that some of the methodologies advocated could be applied in other languages, for instance in the assembly languages that are still widely used (unfortunately). Indeed, the kernel which supports the Concurrent Pascal virtual machine has itself been programmed in assembler in the distributed implementation. This kernel has been extremely well engineered; however, Brinch Hansen only devotes 10 pages of the book to the discussion of its implementation. In contrast, the inclusion of three complete examples programmed in Concurrent Pascal seems somewhat excessive, and I feel that a reader would have benefited more from a more detailed description of what Brinch Hansen calls 'reliable machine programming'. However, the book is trying to sell Concurrent Pascal to replace assembly languages, and there perhaps a conflict arises: perhaps the book should have been titled *The Architecture of Concurrent Pascal Programs?*

P. A. LEE (Newcastle upon Tyne)

Computer Programming Methodology, by W. M. Turski, 1977; 208 pages. (*Heyden and Son*, £14·00)

Programming has reached the stage where there is a measure of agreement about the features of a 'good' program. Sometimes there is sound theoretical support for a concept, sometimes overwhelming evidence from experience, and sometimes nothing more than intuition. *Computer Programming Methodology* presents a thorough survey of current thought on both the design and construction of good programs. Apart from a preface and a page on 'Using this book', there are only four chapters, and the first of these is a four page introduction. The second and third chapters, each of over sixty pages, discuss basic operations and data structures, the way these basic constructs can be incorporated into modules and schemes for module interaction with due emphasis on parallelism and quasi-parallelism. The justification of the chosen constructs is theoretical where the theory exists and intuitive where it does not. Axiomatic methods are well illustrated in Chapter 2, which includes an overview of Guttag's work on axiomatic specification of data types.

The final chapter (72 pages) draws on the preceding material and tackles the problem of actually creating programs. For the individual programmer ways of achieving a predesigned structure are discussed, whilst for the project manager organisation of teams and documentation principles are dealt with too. Notes on reliability (as distinct from correctness), program improvement and maintenance complete the chapter, which, with its firm base in experience, complements the earlier more theoretical material.

In a book of this length there cannot be more than a brief mention of many topics, and the copious references are one of its best features. The book is nicely printed on good paper and my only major complaint is the index, which at 105 entries can at best be described as skimped.

A. P. BLACK (Oxford)