

is also some brief mention of attribute grammars and two-level grammars, together with applications.

Dick Grune is a well known and respected Dutch computer scientist. Both he and his co-author, Cereil Jacobs, work at Vrije University in Amsterdam. They write clear, unambiguous prose which is a pleasure to read. This text, like many others on the same subject, is fairly densely printed and is packed with illustrations and tables. The examples have been carefully chosen to illustrate the techniques being presented. There is a useful index and, as stated above, an excellent bibliography. There are very few misprints.

I am really most impressed with this book. I think it is an exceptionally good modern text on the subject and most certainly will be recommending it to my students.

V. J. RAYWARD-SMITH
Norwich

C. R. SNOW

Concurrent Programming. Cambridge University Press. 0-521-32796-2, 0-521-33993-6. £30 hardback, £13.95 p/b

This is an introductory textbook on concurrent programming, aimed at undergraduates. The study of concurrent programming has often been hidden away inside courses on operating systems, which is unfortunate since students are more likely to encounter a need for concurrent programming techniques within applications programs; this is increasingly true as parallel computers become more commonplace. This new book is therefore a welcome addition to the relatively small number of books that deal with concurrent programming in its own right. Other recent titles that come to mind are Ben-Ari's *Principles of Concurrent and Distributed Programming* (1990) and *Concurrent Programming: Principles and Practice* by Andrews (1991).

The order of material in the first five chapters is fairly standard. The first two chapters introduce the basic concepts of concurrency and process invocation. Chapter 3 discusses interprocess communication by shared data, first using busy waiting and then semaphores; this chapter includes the ubiquitous mutual exclusion algorithms by Dekker and Peterson, and the bounded buffer example. Chapter 4 introduces higher-level shared-data facilities such as monitors, illustrated by a few classic examples: the bounded buffer, readers and writers, and disk head scheduling. The relatively short Chapter 5 deals with communication by message passing. A minor complaint here is that the description of some facilities is perhaps too closely identified with their use in a particular language, e.g. it is hard to distinguish the general properties of a channel from those of an Occam channel.

Chapter 6 is, in my view, the least satisfactory part of the book, which does not justify the 60 pages that it occupies (just over a quarter of the book's length). It describes, in some detail, the concurrent programming features of several languages, namely Concurrent Pascal, Concurrent Euclid, Mesa, Path Pascal, Ada, Pascal-M and Occam. The author claims that this is "a representative selection of languages available at the present time", but many of them are neither modern nor widely available. In particular, the description seems somewhat repetitive, partly because of the similarity between several of these languages and partly because all of them have already been introduced in earlier chapters.

The final chapter describes the implementation of a concurrency kernel in Pascal. This is reminiscent of the "Implementation Kit" in the appendix of Ben-Ari's 1982 book *Principles of Concurrent Programming*, except that it seems to be intended for pedagogical purposes rather than for actual implementation by the reader.

In summary, this is a readable and well written book, though rather dry in places. The only error that I noticed was a misspelling of Eratosthenes, a common fault! It is pitched at a more elementary level than the books mentioned above, especially the one by Andrews, which is far more detailed. In comparison with these other books, there is more detail of specific languages and less emphasis on concurrent algorithms or applications: relatively few examples are presented. This, I feel, is the book's major weakness: the student is likely to end up with a good knowledge of some of the various concurrent programming paradigms and languages but less idea of how to apply them to actual problems.

S. GREGORY
Bristol

D. DIAPER and N. HAMMOND (Eds)

People and Computers VI (Proceedings of the HCI '91 Conference). Cambridge University Press. 0-521-41694-9. £40.00

"People and Computers VI" is a collection of the 30 papers presented in the British Computer Society HCI '91 Conference. Consequently, the selected and refereed texts are of a consistently high standard and are drawn from across the field of HCI, ranging from abstract theoretical discussions to detailed descriptions of particular systems. Paper contributors are multinational, though predominantly British. The Editor's introduction gives an overview of the volume, as well as noting substantial changes in emphasis since the HCI '87 conference as this field develops and matures. The papers are divided into sub topics, such as HCI Frameworks, Graphical Interaction and Task Analysis, with two or three papers in each section.

The first of the two invited conference papers, by Brad Myers of Carnegie Mellon University, describes demon-

strational interfaces and proposes these as the next step in the development of direct manipulation graphical interfaces. Such a system would monitor the user's actions and guess generalizations from his actions to increase the apparent power of the interface while not making his task any more complex. In the second invited paper, James Alty of Loughborough University examines multimedia interfaces, noting that previous studies in this developing field have tended to examine the possibilities of multimedia, rather than the users' requirements and the opportunities presented by the increased communication bandwidth. He concludes with a series of research questions for multimedia use.

Two papers from the HCI Group at York University examine multi-user interfaces and how an individual user's expectations can be supported in such an environment. Co-ordination of groupware work is examined by Pendergast and Beranek, with a description of both the problem areas and a prototype system developed to overcome them. Iconographer, an interactive tool for investigating alternative iconic object representations, is described in two papers from GIST in Glasgow, and a third paper describes the novel "Wet and Sticky" system intended to reintroduce the qualities of real painting materials into artwork produced on computers.

The first of two further papers from GIST in the section on "Evaluation" proposes a better method of HCI evaluation using focus groups rather than individual assessments of usability. The second proposes usability assessment parameters of guessability, learnability and experienced user performance, and demonstrates these by way of two simple experiments. The third paper in this section reports a large study to evaluate two multimedia training systems.

The Speech and Language Technology Group from Hatfield Polytechnic demonstrate an improved recognition performance in a speech-driven word processor by using linguistic prediction to focus the recognition domain on the user's current action. The Police HOLMES computer system was the subject of another group, with a user-centred approach to produce an improved demonstration interface for the system, with some success. Conversational turn-taking was investigated by a group from Dundee and Edinburgh, and this knowledge applied to developing a prototype text telephone for the speech and hearing impaired. An interface for the blind is proposed by a group from York University, suggesting that use of stereo audio cues may be a viable method of navigation for such individuals.

The volume concludes with two papers examining spatial concepts; the first describes research seeking to improve the usability of graphical information systems and the final paper describes a hypertext prototype based on the spatial metaphor.

I. R. MURRAY
Dundee

JOHN J. DARRAGH and IAN H. WITTEN

The Reactive Keyboard. Cambridge University Press. 1992. 0-521-40375-8. £25.00 hardback

There are a number of people who have physical conditions which, in addition to a general physical disability, render them permanently unable to communicate using their voice. It would seem that computer technology has a great deal of potential help to offer such people: with a computer-based system the user can control a powerful and versatile system with only a small amount of movement. The reality is that computer-based communication systems thus far developed are a certain amount of help to non-speakers, but they as yet only allow them to achieve communication rates far below that of unimpaired speakers. (Typically 2-10 words per minute, compared with a normal rate of 120-200 words per minute.) Physically impaired people who are able to make use of their voice face this same rate barrier when they wish to access computers.

There is clearly the need for research into methods for improving this communication rate. A general point: devising ways of helping severely physically impaired people use computer systems is an endeavour which is obviously of benefit in itself, but it also has the potential to give us insights and information which can be more generally applied in interface design.

The task here is, given a ceiling on the input rate which is imposed by the user's impairment, to maximize the result of each input action. One technique is abbreviation expansion. Another is a coding system for storing and retrieving words or sentences. A third method is prediction: attempting to predict what the user is trying to say and completing the task for them.

This book traces the development of a predictive typing system, originally designed to help physically impaired users of a computer system save keystrokes (or the equivalent) in entering UNIX commands. Darragh and Witten begin with an introduction to the problem of communication disability and technical systems which have been developed to help. They devote a separate section to an analysis of predictive text generation systems, both commercially available systems and research prototypes. An earlier prototype of their Reactive Keyboard is described in this section.

The main part of the book is a detailed description of the Reactive Keyboard program. The authors describe the system in terms of its user interface, the predictive mechanism, and its implementation in code. A complete "C" code listing is given in an appendix, since the authors have made the program freely available and invite others to use and modify it as they wish.

The predictive method used is a stochastic model at the character level. While this method can produce very good results, particularly with strings of characters which are used often, as with command sequences to an operating system, one problem is that it can produce plausible but nonsensical strings. The authors' solution