specification for someone with minimal mathematical skill and an appreciation of software engineering problems.

Dave Grosvenor  
Hewlett Packard Labs  
Bristol, UK


The title is in danger of raising both the expectations of the uninhibited and the blood pressure of the cognicenti simultaneously, which is no mean feat. It does, of course, depend upon their views on the task of systems analysis in general, and structured systems analysis in particular! The title is probably the worst thing about the book.

If one goes beyond the title, the preface softens the blow by introducing the book as a description of a “methodology that links structured systems analysis and Prolog to form a logical model of the problem domain.” However, even here the cited advantages of the methodology, and indeed the wide audience for whom the methodology is “likely to be of interest,” are in danger of implying that this link has almost magical powers to solve the besetting problems of the design process.

The enthusiasm of the author for the methodology described has, perhaps, done a dis-service to what is a creditable book. He brings together two subjects which are commonly believed to be at opposite ends of the computer science spectrum and shows the utility of integrating them for solving system-oriented problems. For this reason in particular, it is a very useful text for undergraduates who often see, and are often taught the subject as a plethora of disjoint specializations. The exercises, discussion and background reading at the end of each chapter are very useful aids. The latter is particularly valuable, given the selective nature of the text.

The book is relatively short (approximately 200 pages), but is written so as to be accessible by those with little knowledge of either systems analysis or of Prolog. Indeed it is claimed that no knowledge of either area is required to appreciate the ideas. The author states that he is aware of the dangers of either “scantily treating” or even “mis-representing” one of the specialist areas. However, the selectivity required to address the combination of the two in such a short space and in that style, and perhaps too the tendency to make the method do too much, inevitably leads to some distortion.

The major problem with the systems analysis aspect is that the techniques discussed are not presented in a wider sociological and technical setting. The consequence is that the reader is led to believe that the techniques presented are both necessary and sufficient for the design process. In practice, the data flow diagrams and decision
tables are examples of process modelling and process logic respectively, and arise from just one of the several different perspectives for engineering information systems. For example, although the data dictionary is also well described in the text, explicit data modelling, such as entity/relationship modelling is neglected. The references to other approaches and techniques are not adequate.

This treatment is compatible with a course based upon the excellent, but now dated, Gane and Sarson text, which promotes the combined data dictionary and data flow diagram techniques as the prescribed way for structured systems analysis. The Prolog model of implementing these techniques is presented clearly; the advantages of consistency checking and the simple mapping between the Prolog clauses and the data flow concepts are generally conveyed very well. The functional decomposition reflected by the hierarchical organization of the diagrams is equated to stepwise refinement but is the least satisfactory part of the Prolog approach.

The Prolog aspects of the text, whet the appetite. It does not, however, prepare the reader for the complexity of large programs and the dirty "cuts" which are present in many "real" programs. It also implies that Prolog code is a good means of communicating between analysts and users. I suspect that it is the use of the associated comments, rather than the code which is the more informative, particularly when the modules become larger and require the more esoteric syntactical constructs.

The interface design with "standard" Prolog is restricted and is usually enhanced by proprietary code. This requires familiarity with flavours of Prolog or the decision to opt for simple menu schemes, like the one in the text. The text also provides good examples of Prolog as a tool for implementing the described analysis techniques, but it is not convincing when Prolog is portrayed as a tool for implementing the system under consideration. The Prolog templates themselves are well thought out, but add to the confusion; the reader is left wondering whether the author's intention is to substitute a Prolog representation for the traditional techniques, or to complement them as part of the logical model.

I do not believe that many practising professionals will use this book; it has come too late with relatively sophisticated tools such as analyst workbenches now available, obviating the necessity of becoming familiar with yet another programming language. Undergraduates, however, will benefit from this text because of its "integrating" approach. Furthermore, graduates with these skills may well use such a method to good affect for the exploratory modelling of requirements and for prototyping. Interestingly, an analogous situation exists with formal methods, the advent of CASE tools and the software engineering community.

The author succeeds in showing the flexibility of Prolog by applying it to problems in the commercial domain. One is tempted to speculate on new books in this "integrating" vein; object-oriented SSADM for example might be next, reflecting the popular debates and offering "new ways" of tackling old problems. The cynic, or perhaps the traditional analyst, will still believe that the proposed method, and its means of implementation, will only embellish rather than rewrite the book in what will remain a difficult and challenging area.
Structured Systems Analysis through Prolog is a book with one such method but it is no worse for that. I recommend it.

S. Kay
Department of Computer Science
University of Manchester
Manchester, UK


The programming language Pascal was invented nearly twenty years ago and tutorial textbooks have been available for nearly fifteen years. Despite the more recent advent of other languages serving the same purpose, such as Modula II; despite a greater stress in student teaching on problem analysis and specification; despite the invention of totally new techniques of expressing solutions, such as 4GLs and functional programming... Pascal as a medium for teaching the basic pragmatic skills of programming is still sufficiently popular to encourage publishers and authors.

Judy Bishop's second edition is physically well produced and attractively bound. Desktop publication has been used very effectively to produce camera ready copy. She has used good, but not overdone, variation of font sizes and styles with line divisions and cue graphic symbols highlighting the structured presentation. The syntax diagrams defining the language are an unfortunate exception, with keywords in a different but very similar font to the syntactic references, non-alphabetic symbols in a ludicrously larger font and the syntactic references confusingly in a different font from the corresponding heading on its definition.

The style of presentation of the content is excellent! Clear listed subheadings for each chapter are followed by a stated aim. Numerous miniproblems are used to illustrate language features, including how to choose your socks and how to make up your party invites. Each chapter closes with a summary, quiz and exercises, whose answers are provided in an appendix. However, the explanatory introduction is not up to the generally high standard of content. Here, the viewpoint of computer functionality is not developed properly to show that it is information which is processed. Thus by default, the reader may be left with the impression that it is a calculating engine. Equally, the role of application packages is totally omitted, though these do admittedly have to be programmed initially and tailored by an intellectual act similar to programming. Hardware descriptions are based on an 8086 PC and must rapidly become outdated. The publisher may look to sales of a third edition, but the purchaser may not be so happy.

The philosophy of the book is based on a course to undergraduates. Indeed, an editorial slip in the introduction refers to the book as a course! The aim of the book