designed with economy and consistent structure. The economy arises in this methodology from the table-like description of the interactions, which could be implemented more quickly than by brute-force programming. The structural consistency arises from the notation for prompt, input, action, flow control, escape, help, and input editing for each interaction. The principal human factors benefit would be this consistency, which is only one of many issues that need to be considered in designing a good interface. (By J. M. Hammer, Atlanta, GA) [Excerpted from COMPUTING REVIEWS, July, 1985].

Croft, W. B. (Univ. of Massachusetts, Amherst)
The role of context and adaptation in user interfaces. INT. J. MAN-MACH. STUD. _21_, 4(Oct. 1984), 283-292.

This paper clearly argues the case for _context_ and _adaptation_ in building user interfaces. These factors are in contrast to the issues of presentation and the mechanics of interaction, such as windows, icons, and pop-up menus. A user interface is viewed as a means by which a user maps his task into the available set of tools. Context defines the features of the environment which are important in determining the flow of the interaction. Adaptation is the ability of the system to act appropriately in a given context. An office system and a document retrieval system are used to illustrate the use of context and adaptation.

In the office system, the context is used to adapt the user interface and the system to the user in several ways. For example, planning can be used to automate certain tasks which can be anticipated because of the context. The system can propose possible actions to the user based on what has already transpired. The context can be used to recognize and correct local and global errors of the user. For example, in a database management system environment, most users would have little difficulty in forming a _syntactically_ correct query. But the query may not be semantically accurate and the naive user may not realize this since the magnitude of the data involved will usually be large enough to preclude hand checking. Many people also view any results which come from a computer as being unchallenged gospel.

The paper includes a formalism for task definition, which may be helpful in defining adaptive user interfaces. In the document retrieval example, the types of documents retrieved are adjusted based upon the user satisfaction with previous retrievals. (By A. L. Tharp, Raleigh, NC) [Excerpted from COMPUTING REVIEWS, July, 1985].

Smith, Michael J. (Univ. of Wisconsin - Madison, Madison)
Human factors issues in VDT use: environmental and workstation design consideration. IEEE COMPUT. GRAPH. APPL. _4_, 11(Nov. 1984), 56-63.

Smith's paper provides a summary of human factors issues which relate to the design and use of VDTs. Smith summarizes the results from a number of field studies on the health problems associated with the use of VDTs. Topics such as screen glare, improper illumination, screen flicker, and temperature and humidity problems associated with workstation design are discussed. The paper is recommended reading as an introductory source on the topic of human factors in VDT use. (By W. Barfield, West Lafayette, IN) [Excerpted from COMPUTING REVIEWS, July, 1985].

Sheppard, Sylvia B., Bailey, John W., and Bailey, Elizabeth K. "An empirical evaluation of software documentation formats." In _Human factors in computer systems._ J. Thomas and M. Schneider (Eds.), ABLEX Publ. Corp., Norwood, NJ, 1984, 135-164.

This paper presents a summary of tests discussing various symbolic and spatial formats on sfotware program specification documents. Programmers of varying ability (years and exposure to multiple programming languages) are presented with design specifications ranging from four types of symbolic design languages to three spatial types of format arrangements. The symbolic language consists of ideograms, a program design language, and normal and abbreviated English. The spatial arrangements consist of sequential, hierarchical, and branching flow charts. Tests were conducted in the areas of writing, debugging, and modifying of software programs. The graphical representation of the documentation formats at the end of the paper and the tables throughout the paper are indispensable.

Conclusions are drawn on the effect of the programmer's experience and software documentation formats as they relate to errors in the programs and time expenditures. The combined use of a succinct language with a branching spatial arrangement in the documentation specifications and broad programming experience is found to be optimal for effective programming. (By C. Baddorf, Golenta, CA) [Excerpted from COMPUTING REVIEWS, July, 1985].