**Listed are abstracts from recent papers by IBM authors. Inquiries should be directed to the publications cited.**

# Abstracts

**An approach to computer-aided translation,** Erhard O. Lippman, *IEEE Transactions on Engineering, Writing and Speech*, February 1971. As part of a preliminary investigation of computer-aided translation under time-shared data processing, various techniques of computerized dictionary lookup and text file management have been explored under the time-sharing system TSS/360. Computer-aided translation (CAT) is a storage and retrieval operation carried out on line with a computer during the time in which a translation is produced. A system of dictionary access and updating routines, text-processing facilities, and on-line utilities is designed to telescope the delay between the initiation of a translation and its finished printout. The system does not attempt to simulate the human translator by producing an autonomous translation via programmed algorithms; rather, it serves as an extension of the capabilities of the user, who is able to call on the resources of the computer as needed in the process of translation and get an immediate response. Under the system described, users communicate over ordinary telephone lines with the computer by means of remote terminals. In employing the system, the user can switch back and forth as many times as required among human translation, direct dictionary lookup, editing, printing, and system interrogation, and thereby achieve rapid iteration toward the desired goal, i.e., a finished translation.

**A computer model for the financial analysis of urban housing projects,** Denos C. Gazis, *Socio-Economic Planning Sciences*, **5**, 1971. A financial analysis model has been implemented within the framework of the APL time-sharing system. The model allows different levels of input and output commands which can be used by persons with different degrees of familiarity with computing, in general, and the APL system in particular. It is the result of a joint study between IBM and the Urban Coalition aiming at providing a computer model of the urban housing process which would allow both government planners and private developers to analyze the financial prospects of individual housing projects under a variety of subsidy provisions and other constraints such as FHA regulations concerning profits. The model includes provisions for different types of sponsors of urban housing projects allowing for different tax treatment of profits and losses, and different formulas for estimating the rent charged. The basic output is either in the form of a summary of financial figures or complete financial tables such as income statements, balance sheets, and cash flows.

**Data processing – now frontier for scientific management,** William J. Schroeder, Jr., *Management Advisor*, July-August 1971. The advent of the computer during the past twenty years has brought with it the evolution of a new company department – data processing. This growth has not been without far-reaching problems. Cost control within a service department of this magnitude is a formidable task, and planning takes a back seat. This article examines several of the many techniques of scientific management that appear to offer significant benefits to data processing such as cost control and budgeting, project management, and long-range planning. Each technique is discussed with specific examples to its use in data processing department management and control. The author concludes with direct suggestions on how to get these tasks accomplished with the tools and resources available.

**Ecap II—An electronic circuit analysis program,** F. H. Branin, G. R. Hogsett, R. L. Lunde, and L. E. Kugel, *IEEE Spectrum*, June 1971. Ecap II, a nonlinear dc and transient circuit analysis program, uses a free-format, problem-oriented language for describing circuit parameters and topology. Its computation speed is high enough to make the transient analysis of kilobranch nonlinear circuits feasible. The program includes a nested model feature that permits the user to store circuit descriptions of devices and subcircuits for subsequent recall and insertion into larger circuits. Nonlinearities can be specified by means of tables, built-in functions such as the diode equation, and FORTRAN subroutines. A diagnostic feature pinpoints errors in the input format or inconsistencies in the circuit description.

**Human factors in the design of an interactive library system,** Caryl McAllister and John M. Bell, *Journal of the American Society for Information Science*, March-April 1971. ELMS (Experimental Library Management System) is an experimental system for total library management, operating on-line with an IBM S/360 through IBM 2260 and 2741 terminals. The system is designed to handle large amounts of highly variable information which it processes on command, giving on-line computer service for all library operations. At the same time, it must accommodate the different needs and skills of a broad range of library users, from new patrons to well-trained librarians.

Such a system presents programming problems that will be typical of large, interactive computer systems in the seventies. This paper discusses ELMS features that facilitate user interaction, and may prove useful in similar systems: techniques for tutoring the user (display format, one-question, one-answer displays, and KWIC indexing); adaptability for the experienced user (command chains and a standard set of four-letter mnemonic codes for higher-level control); minimization of keying (line numbers, one-character mnemonic codes used with procedures, and use of default conditions); performance of clerical tasks by exception notification; and collection of operational statistics to help improve the system.

**A run-time mechanism for referencing variables,** W. Henhapl and C. B. Jones, *Information Processing Letters*, February, 1971. This paper describes a mechanism, which the authors believe to be new, for the run-time referencing of variables in Algol-like languages. The ability, in such languages, to invoke procedures recursively and to pass procedures as parameters makes the discovery of such mechanisms an interesting problem.

The published mechanisms are generally based on the extremely elegant approach of Dijkstra, with which the reader is assumed to be familiar. It has been established that simple blocks can be statically related at compile time to ther containing procedure, thus leaving only procedure entry/exit to be handled during execution.

The proposed mechanism retains the concept of a display which need be updated only during procedure entry and exit. However, the display, which is used to link some statically assignable index of a procedure to the storage for the variables of that procedure, is based on a different index. Whereas the published methods assign an index to each procedure which is the number of statically enclosing procedures, the proposed mechanism assigns a unique name and display position to each procedure and specifies an updating mechanism considered to be optimal. This increases the length of the display vector from the depth of the most deeply nested procedure to having one entry for each procedure. As mentioned, such a display must be maintained on procedure entry and exit. The advantage of the proposed, longer display is that, by identification of special cases, its maintenance will require fewer operations.