Syntactic definition and parsing of molecular formulae, Part 2: Graphical synthesis of molecular formulae for data base queries

P. G. Barker and P. S. Jones

Department of Computing, University of Durham, Science Laboratories, South Road, Durham DH1 3LE

This paper describes a series of graphic interfaces to a chemical information system. The data base and query language permit several different types of interactive data base queries to be formulated.

(Received January 1977)

Interactive computer graphics, a facility that enables human/computer communication by the use of imagery, permits a new dimension in man/machine interaction. Techniques for computer manipulation of graphical information have been proliferating quite rapidly over the last few years. These techniques are now well established in such areas as engineering, science and education. Within the area of computer aided design (Prince, 1971) a light pen or other suitable graphic device can be used to help the engineer in his iterative design process. Similarly, within education the sophisticated types of graphic effects and man/machine dialogue involved within systems such as PLATO (Bitzer, 1976; Smith and Sherwood, 1976) and CALCHEM (Ayscough, 1976) help to place a more natural interface between the student and the computer. Developments within the general area of graphic interaction have led to a three tier approach to the implementation of 'graphic languages' (Nake and Rosenfeld, 1972) as follows:

(*a*) formal languages which generate or parse non-string like objects such as arrays, labelled graphs, etc. (Narasimhan, 1966)

(*b*) programming languages for interactive graphics (Kulsrud, 1968)

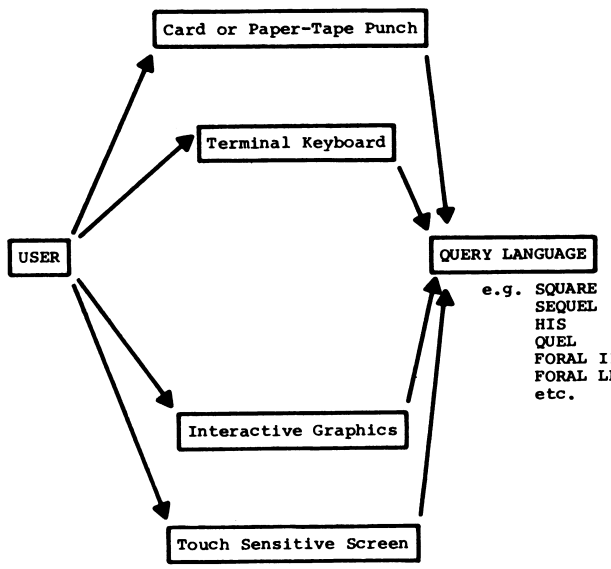
(*c*) software packages for both graphics and image processing (Koenigsberg and Thanhouser, 1974).

Over the last year or so there has been much interest in the application of the principles of computer graphics within the data base and information system areas. Here graphic equipment has been used for both the formulation of search queries and for the facile and ergonomic display of retrieved information (Benz, 1974; Bork, 1974). Some of the major reasons for using graphic equipment in this area are:

1. It can permit a more natural mode of communication with a computer system; a human is able to simulate 'pointing' instead of interacting via error prone typing techniques. Minimising the amount of keyboard interaction can help to eliminate both typographical and spelling errors.
2. Graphic languages are able to increase the quantity of information flow across the interface that exists between an information system and its users. This is the prime objective of human engineering within man/machine systems—optimising the exchange of information and adapting the machine to the needs of man.
3. It permits the facile implementation of the technique of information retrieval by the method of stepwise refinement (Barker, 1976a).

In this paper we describe some application of computer graphics to the design of a query language that permits the retrieval of information from a chemical data base. For this, we assume a simple and commonly accepted picture of the data base utility as shown in **Fig. 1**.

It is through the query language that the user of the system specifies to the access functions the nature of the information that is to be retrieved from or added to the data base. Frequently, the need for a query language may be obviated by means of a suitable data manipulation language (CODASYL, 1976) which may be used to describe the subschema in which the user is interested. We are not concerned with this type of data base philosophy in this paper. The role of a graphics interface is depicted in the following illustration:



It is via a keyboard terminal, card or paper tape punch, touch sensitive screen or interactive graphics device that a user's information requirement is ultimately mapped onto a data base query language such as SQUARE (Boyce *et al.*, 1973), SEQUEL (Chamberlain and Boyce, 1974), HIS (Barker, 1976b), QUEL (Held, Stonebraker and Wong, 1975), or 'Query by Example' (Zloof, 1975). The two dimensional capability of graphics terminals provides a closer approximation to the real three dimensional world than do one dimensional punched cards. It is natural to assume that graphic input and output can simplify any type of problem solving—particularly if the linguistic interaction is in end-user terms. In view of this, interactive graphics equipment, such as a Light Pen (LP) and Cathode Ray Tube (CRT) combination, are, increasingly, forming an important part of the hardware necessary to formulate data base queries that are more easily 'understood' both by the linguistic analysers associated with the information/data base system, and by the users of the system. The work of Senko (1976), Stonebraker (Held, Stonebraker and Wong, 1975; McDonald and Stonebraker, 1975), and Weller and Williams (1976) are typical of recent activity within this area.

Stonebraker uses a graphical human/machine 'front-end'

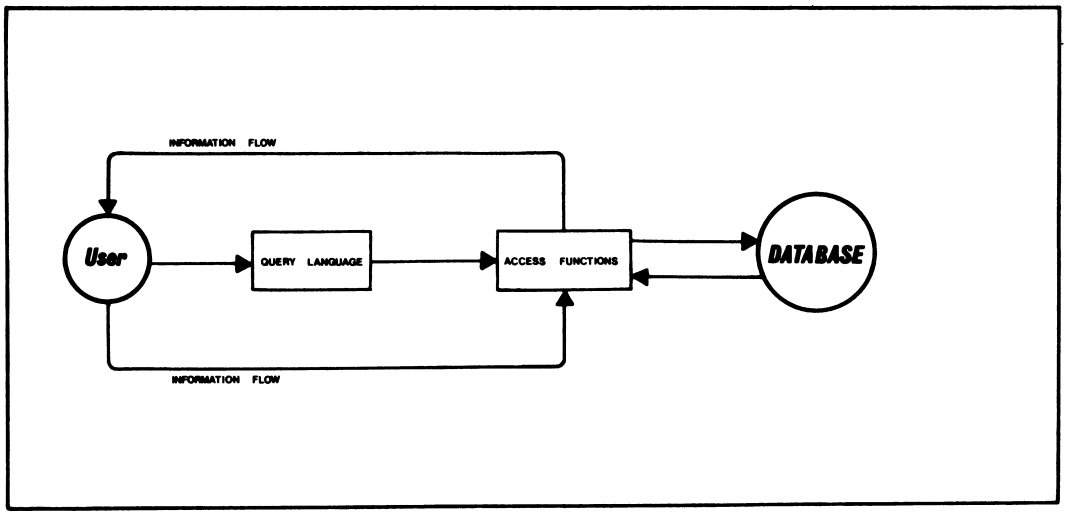


Fig. 1 Conceptual model of the data base system

interface system called CUPID (Casual User Pictorial Interface Design) to support nonprogrammer interaction with a relational data base management system called INGRES (Held, Stonebraker and Wong, 1975). In CUPID, a query is diagrammed by the user by means of 'menu-move' operations on a CRT. The picture processing for CUPID is handled by a graphics modelling system called PICASSO (Austin and Holmes, 1972) which compiles pictures into the query language QUEL. Similarly, in Senko's system which uses an information structure based upon a semantic network of binary associations, FORAL II is used as an end-user query language which has a simple statement structure designed to be easily understood by nonprogrammers. FORAL II provides the basis for the CRT display/light pen language, FORAL LP. This graphic language achieves a high level of interactive dialogue by allowing the user to work with a light pen on screen representations of the real world associations between the entities in his problem. As a final example, the recent work of Weller and Williams on graphic and relational data base support for problem solving is worth mentioning. This work illustrates how a picture building system may be used to provide high level commands for easy use of a data base through the use of both graphics and nongraphics terminals. In this work a relational data base called XRM—an eXtended (*n*-ary) Relational Memory—is used. Commands are included to facilitate graphical input to and output from the data base. The symbolic interface to the relational data base is GXRAM which allows relations and domains to be referenced by symbolic name. An important component of this, picture building system is a 'relations editor' which allows the user interactively to create, edit and display relations. The tuples from which a relation is composed may be typed in or added graphically so that, to add a geometrical line to a relation, the user could type in the tuple or just point to the desired end points on the display screen.

This paper presents a description of some work being done by the Interactive Systems Group at the University of Durham. It involves the utilisation of a Digital Equipment Corporation (DEC) Graphics facility (CRT + LP) for the implementation of graphic interfaces that permit a chemical data base to be interrogated by means of a light pen and terminal keyboard combination. The interfaces that we have constructed consist of a series of '*frames*' that are displayed upon the screen of the CRT. The user synthesises his data base queries by pointing, with the light pen, at various syntactic and semantic units that are displayed within his viewing area. The software that has

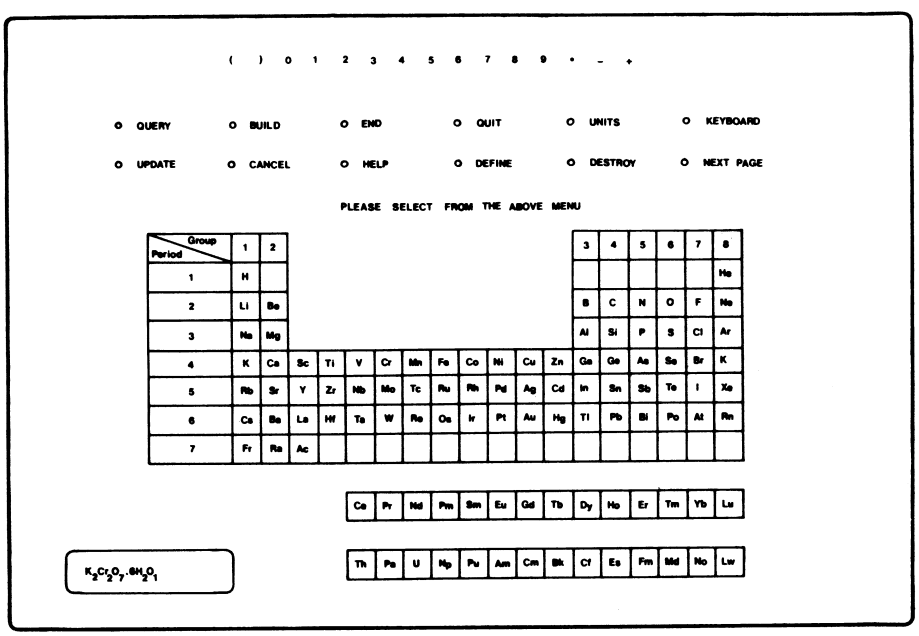


Fig. 2 Graphic interface to a Chemical Information System (opening frame)

been constructed maps light pen hits into either data base queries or frame manipulation commands. Frames are related to one another according to a hierarchic tree structure. The '*opening frame*', that is the first one presented to the user when the system is activated, is shown in **Fig. 2**.

The graphic interface shown in this diagram enables a user of the system to build up a data base query by means of strategies that are constructed through the use of a light pen. Essentially, through the *Command Directives* that are available the user is able to retrieve/update stored information on either

(*a*) chemical elements

(*b*) selected chemical compounds.

Alternatively, using other menus the user is able to specify various selection criteria that can be used to form a 'search profile' for sequential or step searching of a file of stored information. These applications will be described later.

The CRT screen, as shown in Fig. 2, comprises four general areas. Of these three are light sensitive, that is register a 'hit' (or CPU interrupt) when the light pen is brought into their vicinity. The fourth area is a display area that acts as a status register to depict the current state of string vectors that are being synthesised by the user via his graphic dialogue. The three light sensitive areas consist of:

1. The upper row of decimal digits, parentheses and other special symbols that can be seen in the topmost part of the diagram.
2. A set of command directives that control the nature of the interaction that takes place between the user and the system, and
3. a graphical representation of the '*Periodic Table*' containing all the atomic elements known to man. In this diagram the elements are represented by their atomic symbols, all of which are light pen sensitive.

Because each of the above areas of the screen are light pen sensitive it is an easy matter to recognise command directives, chemical symbols and any of the other symbols shown at the top of the screen. The bottom left hand corner of the viewing area contains a display section which enables character strings, built up under either program or light pen control, to be displayed. The string shown on the bottom left corner of Fig. 2 is that which would have been constructed after the sequence of interactions, whose map is shown in **Fig. 3**, had been completed. The concept of a 'hit map' is based upon the similar diagrams used by Senko (1976) to indicate temporal order in a graphic dialogue. A map similar to that shown here is used to indicate the sequence in which light pen hits are made and how they correlate with the screen symbols that produce them. To interpret the map, the viewer commences at the node labelled start and proceeds to the node at which the edge/arrow leaving that node points. The number, enclosed within a diamond shaped box, gives the ordinal sequence of the light pen hit produced by the unit whose node is currently being entered. Fig. 3 depicts how a user would 'build' up a molecular formula as a result of a series of light pen hits. The display shown in the bottom left hand corner of this figure represents the state of the molecular formula vector during a synthesis, after five light pen hits had been registered according to the following scheme.

The user

(*a*) points at the BUILD unit (a Command Directive)

(*b*) points at the *K* symbol (Group 1, Period 4)

(*c*) points at the 2 symbol (the topmost display line)

(*d*) points at the Cr symbol (Group VIa, Period 4)

(*e*) points at the 2 symbol (topmost display line).

He then carries on registering hits with the light pen until he has completely synthesised a molecular formula at which point he could request a data base search or update based upon the formula he has constructed. Termination of the context created by the BUILD command directive is achieved automatically when the next command directive (QUERY or UPDATE) is issued. Each of these directives creates its own context and frame sequences. Before either of these contexts is set up the system has to check, in the case of a previous BUILD context, that the molecular formula synthesised by the user and displayed in the display area of the screen is correct. This is achieved by means of a syntax checker based upon an algorithm described in Part 1 (Barker, 1975). An invalid formula produces an appropriate diagnostic which is displayed upon the CRT screen and then returns the system to its initial state. A correct formula permits the new context (required by the QUERY or UPDATE directives) to be generated with subsequent retrieval from or updating of the data base. Whether the system generates the frames and context to process

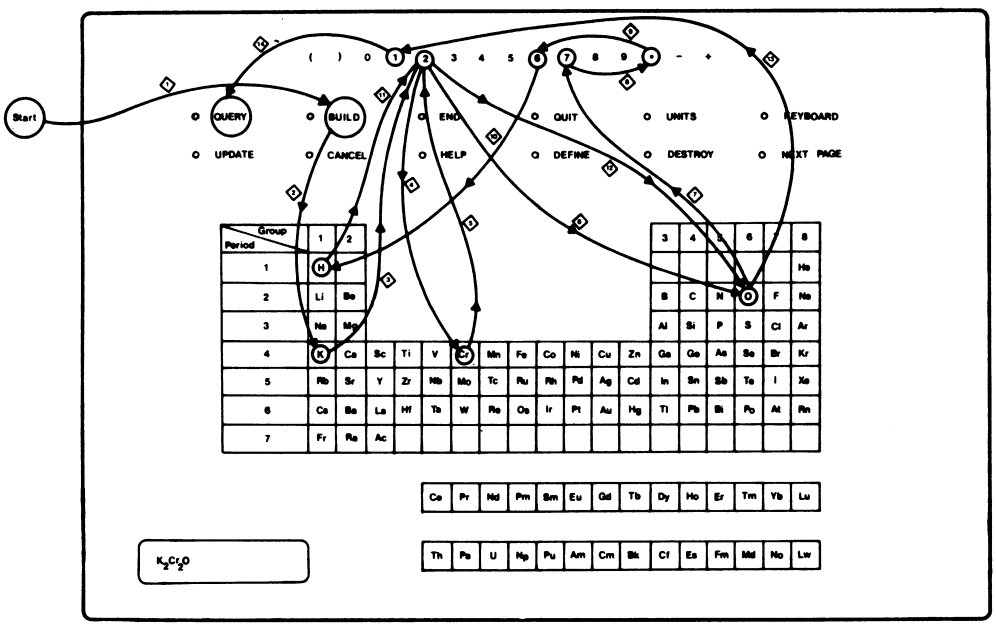


Fig. 3 Synthesis of a molecular formula via graphic interaction (hit map)

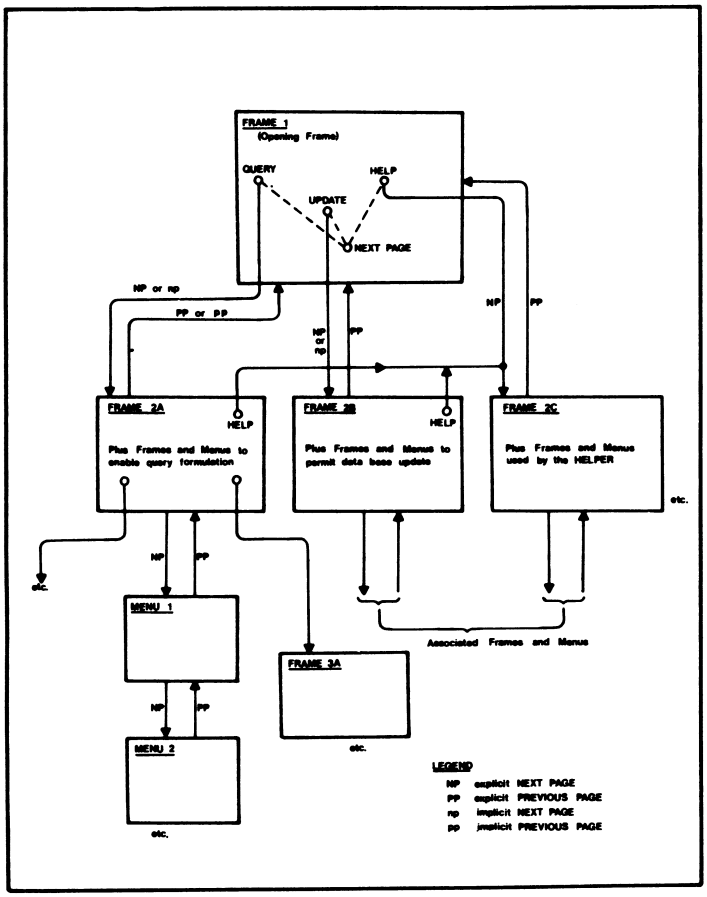


Fig. 4 Relationship between frames and menus

a query or update is subsequently determined by the setting of a predetermined switch variable within the communication vector through which the overlayed processing modules communicate one to another. Details of this will be presented later.

Description of command directives

An outline of the various command directives available through frame 1 (the opening frame) and subsequent frames is represented below. Each of these is available to the user through light pen selection.

QUERY

This initiates a data base query with respect to an item that has been selected or constructed as a result of a dialogue with the opening frame. Hitting the QUERY unit on the CRT screen causes the opening frame to disappear and the second frame to be displayed. Whether an element or compound has been selected for the query is indicated to the new frame by means of a switch variable within the communication vector. Deletion of the opening frame, and activation of a subsequent one, only takes place if an element has been selected or if a syntactically correct formula has been constructed. Any violation of this procedure causes a suitable diagnostic message to be issued and the invitation

PLEASE SELECT FROM THE ABOVE MENU

to reappear on the screen. This permits the user to return to 'square one' and start again.

BUILD

This command directive indicates to the system that the user wishes to construct a molecular formula by selecting symbols from within the Periodic Table of the elements and from the topmost display line. As symbols are pointed to they are reproduced in the display area at the bottom left hand side of the screen.

END

This directive is usually not used within the context defined by the opening frame. Essentially, this directive is used to terminate a build sequence (molecular formula or query) when there is no means of implicit termination available. Typical of such situations is that which exists when the HELP environment is operative and the system is in tutor mode and has invited its user to present a molecular formula or query for checking during a teaching sequence.

CANCEL

May be used to cancel the effect of the previous light pen hit within the context of a BUILD directive. *n* successive CANCEL operations would delete the *n* rightmost symbols contained in the display area. Given outside the context of a build directive, this command cancels the effect of the command within whose context it is issued.

UPDATE

Allows the user of the system to alter the values of stored information or else delete from or insert into the data base. Where the system is shared over many users the necessary security locks must be opened before updating will be allowed. Like the QUERY directive, the UPDATE directive initiates a new frame.

QUIT

Causes the system user interaction to be terminated; this is accompanied by the display, optionally, of statistics that have been collected during the dialogue.

NEXT PAGE

Causes a new frame to be displayed. By means of this directive and its associated PREVIOUS PAGE directive, the user is able to scan through the frames available to the system. These commands are the inverse of one another and so the user can skip backwards and forwards at will. The relationship between the frame/page structure of the system and these commands is shown in **Fig. 4.** These frame/page manipulation directives will be discussed in more detail in the next section.

HELP

This is used by the user when he wishes to obtain some help or instructional material on how to use the system. This feature is also used to tell users of the system about new or enhanced features added to the system or changes made to old features. When the HELP directive is given the computer enters tutor mode.

UNITS

All information values stored within the system have associated with them their appropriate scientific units. Frequently, a user will need to have retrieved information reported in terms of units that are different from the default values used by the system. The UNITS command enables a user to interrogate existing units and specify the nature of any new units in which information values are to be reported.

DEFINE

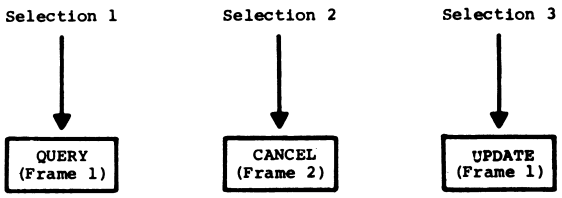
Is a command directive that permits a user to create and store names queries/relations. A definition capability is a mechanism which allows the system to 'learn' new concepts. For example, 'INERT GASES' could be the name of that subset of elements that exist within Group O of the Periodic Table. Similarly, 'LANTHANIDES' could be the name of the

subset of entities such that the elemental atomic number of a member entity lies in the range 57 through 71.

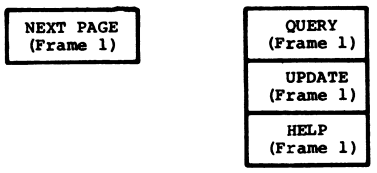
DESTROY
Is used in conjunction with DEFINE and UNITS to erase the definitions of stored relations or conversion factors used to transform scientific units.

AND, OR, NOT
These are not really command directives but are the standard logical connectives used to construct data base search queries. They have their usual meaning and operator hierarchies.

Combining rules for the command directives
Each command directive has a context associated with it. Normally, two contexts cannot co-exist simultaneously—the exception, however, is the HELP context. Some contexts necessarily precede others since one may set up the environment required by a subsequent one. Consequently, the system only permits command directives to be issued in a well defined fashion. This is achieved, in part, by means of the dynamic way in which the system operates, so preventing many illegal combinations of command directives. Thus, the selection of the QUERY directive by a user automatically precludes his subsequent selection of an update directive. QUERY, given as a leading context, would lead to a frame change, so preventing subsequent selection of UPDATE. Note that the following combination would, however, be valid.

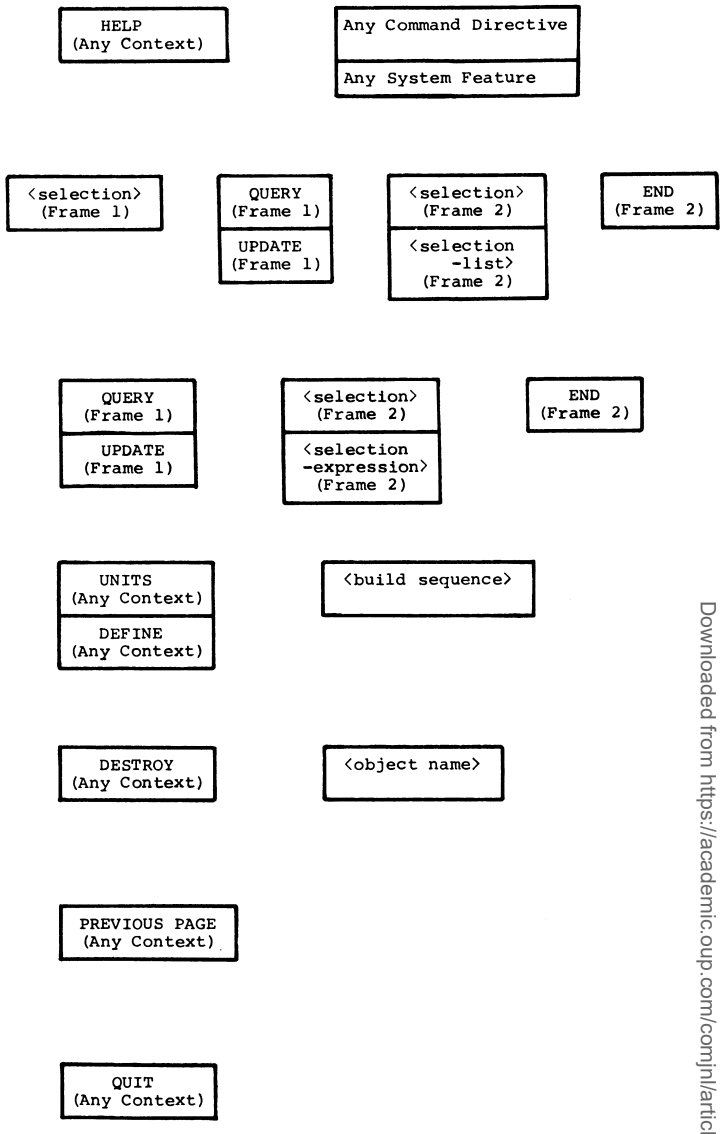


In the above example, the command directive chosen by light pen selection is enclosed within a rectangular box. The order in which hits are made is as indicated—from left to right. Each of the boxes contains the name of the command directive selected and also the context in which that selection is made. Further examples of how the command directives combine together and how the simple syntax may be used to describe a graphic dialogue are shown in the following examples.



The last example shows how the NEXT PAGE directive is followed by one of QUERY, UPDATE or HELP. The vertical arrangement of boxes, shown on the right of the above 'graphic sentence', is used to imply selection from the possibilities listed. The NEXT PAGE command used in the above fashion, and in conjunction with the PREVIOUS PAGE command, can be used to permit casual frame browsing. Further examples of the simple syntax are given below.

Further details of the techniques for representing a graphic dialogue and a more explicit definition of the syntax of the graphic language will form the basis of a future communication. Note that the system does not permit, at present, embedded QUERY commands and in this context should be compared, and contrasted, with languages such as SEQUEL and FORAL LP.

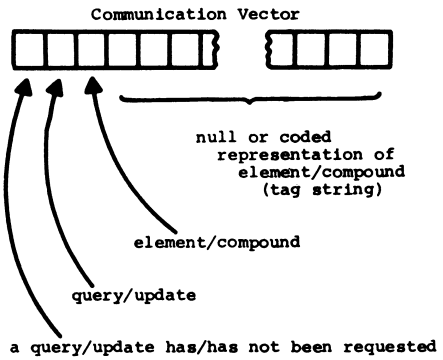


Implementation details
The system described in this paper was implemented on a DEC PDP 11/10 minicomputer system supporting a VT11 graphic display processor, VT17 CRT Display with light pen attachment, LA35 Decwriter console and two RKO5J exchangeable disc units. The memory size of the configuration was 16K. The hardware that was utilised was an integral part of the DERMAN system (1976) being implemented at the University of Durham. The software used was the RT11 real time system provided by DEC (1974a).

The BASIC/GT facility used enabled the frames displayed on the CRT screen to be constructed as a series of subpictures (DEC, 1974b) each of which could be activated as a display subroutine by the display processor. Each subpicture had a tag by which it could be referenced/identified. This allowed light pen hits to be identified and subpicture status to be changed (for example, to enable 'blink' mode, to turn off/on light pen sensitivity of subpictures and to alter display intensities, etc.).

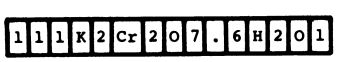
The frame depicted in Fig. 2 was originally implemented using a subpicture for each command, each symbol in the topmost display line and each atomic symbol contained in the Periodic Table. This necessitated a total of about 130 subpictures. Using this technique for correlating light pen hits with atomic symbols was not effective because the structure of the display file, produced as a result of using so many subpictures, caused considerable flicker of the displayed frame. To overcome this problem the set of chemical elements was displayed as one subpicture and the individual element

responsible for a hit identified using a table look-up and hashing procedure based upon the (x, y) coordinates provided by the light pen software. As light pen hits occurred the subpicture tags were stacked, allowing repetitive CANCEL operations to be performed, when necessary, simply by altering the value of the stack pointer. On completion of a BUILD operation the string of subpicture tags could be passed to a parser for syntax checking and then (provided no errors were detected) via a communication vector to the other software modules responsible for implementing subsequent frames and the data base system. The format of the communication vector is as follows.

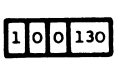


The communication vector corresponds to a reserved collection of storage locations within a communications area used by the graphics supervisor that controls the overall activity of the system. Each display frame was constructed by means of a BASIC/GT program and 'saved' as a disc file. The individual components of the processing system were established as separate BASIC routines. By using this approach, both the display files and program files could be activated as modules under the control of a resident supervisor. This supervisor controlled the loading of the saved display files and the overlaying of the processing segments in the program overlay area. This is shown diagrammatically in **Fig. 5.** When the system is loaded, Frame 1 is always the opening frame.

When the opening frame (or any other frame) deletes, it sets the switch settings within the communication vector so that ensuing frames can pick up the 'threads' left by the previous one. Thus, when the user hits the QUERY unit, after constructing a screen display similar to that shown in Fig. 2, the communication vector would be set up as follows.



This indicates that a QUERY/UPDATE hit has been made (location 1 of the vector is set to 1), more specifically that it is a QUERY (location 2 is set to 1) and that a molecular formula has been synthesised (location 3 is set to 1) and that this formula follows, in coded form, in subsequent locations of the vector. The atomic symbols that comprise a molecular formula are represented within the communication vector by integers whose value corresponds to 100 plus the elemental atomic number. Thus, a communication vector having the following contents:



would indicate that an update is to be made of some stored information associated with the element whose atomic number is 30, namely, *Zn*. Each of the above examples illustrates what is called a Type 1 QUERY/UPDATE. As a final example, consider the following state of the vector:

TYPE 1 ACCESS—ENTITY KNOWN, ATTRIBUTES REQUIRED

Case 1

CONTEXT	*Zn*
ATOMIC NUMBER	?
MELTING POINT	?
NEXT PAGE	
ANALYTICAL METHODS	?
NEXT PAGE	
LITERATURE SOURCES	?
END	

Case 2

CONTEXT	NalCll
DENSITY	?
CRYSTAL STRUCTURE	?
NEXT PAGE	
NEXT PAGE	
PREVIOUS PAGE	
ANALYTICAL METHODS	?
END	

TYPE 2 ACCESS—ATTRIBUTES KNOWN, ENTITY/ENTITIES REQUIRED

Case 3

CONTEXT	?
ATOMIC WEIGHT	>300
AND	
DENSITY	>4·63
AND	
DENSITY	<10·61
OR	
ATOMIC WEIGHT	>350
AND	
DENSITY	<11·64
END	

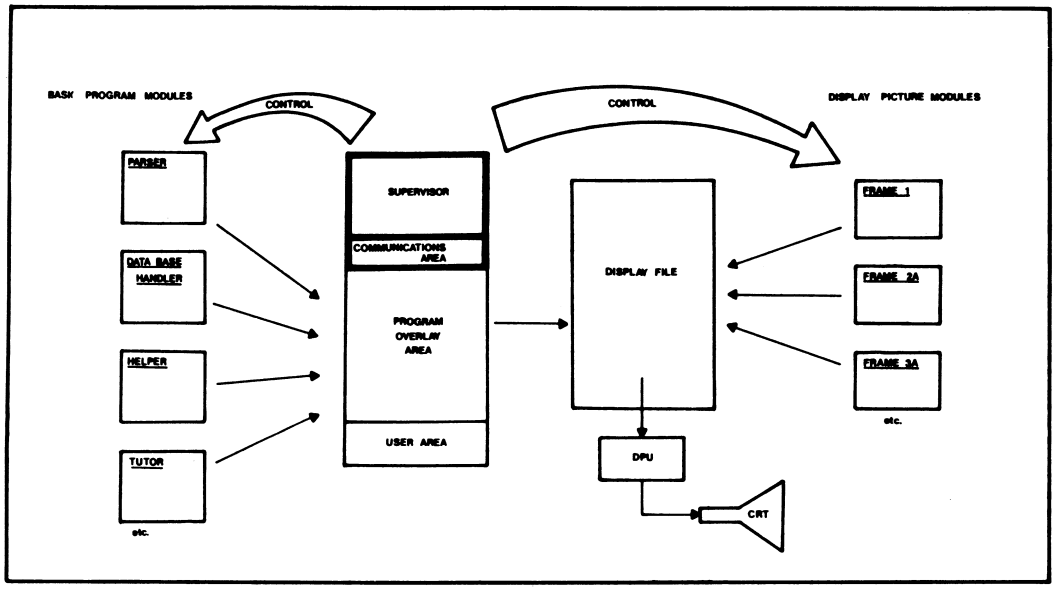


Fig. 5 Module organisation and overlay diagram

This latter example shows that a QUERY has been specified but no element symbol has been selected or molecular formula built. This form of the communication vector indicates that elements or compounds are to be retrieved rather than their attributes. This is called a Type 2 QUERY/UPDATE. These will be described in more detail in the section that describes the graphical dialogue and query construction.

The supporting data base

The data base necessary to support the system described in this paper was part of a Chemical Information System (CIS) which has been outlined earlier (Barker, 1976b). Several types of chemical information system have been described in the literature. Recently Munn (1965) has described a hierarchical tree structured computer based information system which could be used to create, modify and search selectively a data base both in interactive and non-interactive mode. Feldmann *et al.*, (1972a; 1972b; 1970) have described interactive systems based upon both conventional keyboards and graphic interaction. The latter systems have been used to retrieve chemical structure information (typically mass spectra, carbon-13 nuclear magnetic resonance spectra and X-ray diffraction data) and bibliographic information relevant to these areas. Feldmann's system consists of a series of numerical and bibliographic data bases together with a battery of interactive conversational computer programs that can be used to search for and retrieve information from them. Our system, which is also interactive, uses both keyboard and graphic interaction, is based upon the relational model of data, is not specifically orientated towards the handling of structural data, attempts to minimise keyboard interaction and uses the technique of stepwise refinement to improve the precision of retrieved information. Further details of the implementation of this system are given elsewhere (Barker, 1978).

Examples of graphical dialogue and query construction

As we have mentioned earlier, the system permits two basic modes of access; these are referred to as Type 1 and Type 2 access modes. In many ways these two modes are the inverse of each other. Type 1 access is used to manipulate the values of the attributes of a specified entity, while Type 2 is used to manipulate the symbolic representation of entities specified by means of suitable predicate relations between attribute values. These differences are brought out in the 'hit history' diagrams shown below.

Case 4

CONTEXT	?
MOLECULAR WEIGHT	>460
AND	
DENSITY	<4·7
AND	
PRESENT	Na
AND	
ABSENT	Si
END	

The above hit history tables indicate how Frame 1 (Fig. 2), Frame 2A (**Fig. 6**) and the menus used in conjunction with this latter frame (**Table 1**) can be used to formulate four different selection strategies. The menus presented in Table 1 are used to extend the scope of Frame 2A.

To illustrate the procedure by which a user constructs an access strategy we will describe, in detail, the formulation of a Type 2 query. When the user first interacts with the system, he is confronted with Frame 1. He touches the QUERY unit with the light pen thus causing an immediate page change. He is now confronted with Frame 2A. On this frame he selects the option MELTING POINT and qualifies this by building the string '>1000' using suitable light pen selections. He then selects AND logic and specifies a second attribute, namely, DENSITY. This latter attribute is qualified by the string '>4·56' again synthesised via light pen selection. Finally, to narrow down the scope of the search, the user indicates that the attribute ELEMENT is applicable. This interaction has now produced the following search strategy:

(MELTING POINT > 1000) AND (DENSITY > 4·56) AND (ELEMENT)

He terminates his building operations by touching the END

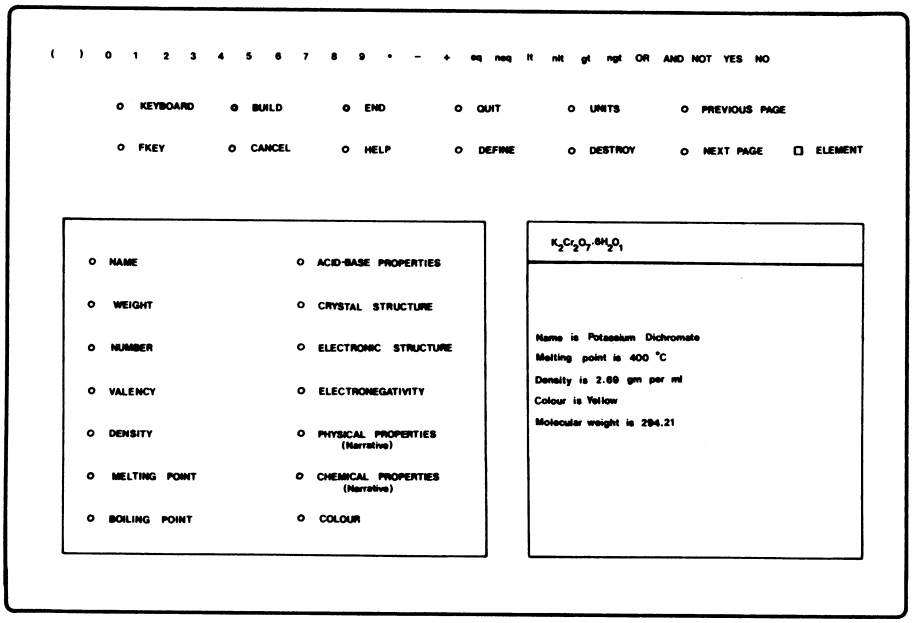


Fig. 6 Details of CIS Frame 2A

unit. After checking the syntax of the search query that has been synthesised the system interrogates the data dictionary to deduce the nature, meaning and locations of the attribute values involved in that query. Provided that the semantic checking is successful a data base search is initiated. The hit history corresponding to the above query is shown below:

CONTEXT	?
MELTING POINT	>1000
AND	
DENSITY	>4·56
AND	
ELEMENT	
END	

The hit map corresponding to the graphic interaction described above is shown in **Fig. 7.**

Conclusions and future work

The relational implementation used in this work was a test bed to investigate the feasibility of the relational approach within this area. We have found it feasible to use this model of data, in conjunction with the technique of stepwise refinement, to provide a user-oriented workable information system for use by chemists. In agreement with the conclusions of Williams (1974), we believe that the combination of low cost interactive computer graphics equipment with a relational data base architecture can offer many worthwhile benefits to users of such systems.

Further work will involve the implementation of a relational model within the host environment provided by the UNIX Time Sharing System (Richie and Thompson, 1974). We shall pay particular attention to the use of graphic interfaces and make user-orientation our prime design objective.

Table 1 Menus used in conjunction with Frame 2A

1. Menu 1

☐ COMPOUNDS	
☐ ANALYTICAL METHODS	
☐ ISOTOPES	
☐ ATOMIC VOLUME	
☐ IONIC RADIUS	
☐ ATOMIC RADIUS	
☐ HEAT OF FUSION	
☐ THERMAL CONDUCTANCE	
☐ ELECTRICAL CONDUCTANCE	
☐ HEAT OF VAPORISATION	
☐ SPECIFIC HEAT	
☐ COVALENT RADIUS	

2. Menu 2

☐ FIRST IONISATION ENERGY	
☐ STRUCTURE	
☐ LITERATURE SOURCES	
☐ SPECIALISTS	
☐ MANUFACTURERS	
☐ PRESENT	☐ ABSENT

We are also looking at ways in which the notation that we have used to describe a graphic dialogue can be utilised to specify and teach the interactive sequences required to construct both static and motion (Baeker, 1969) pictures on a CRT. It is believed that such pursuits are of worthwhile pedagogic value as they might enable us to understand, in greater depth, how we might be able to utilise the computer graphics technology of the future to teach young children how to create meaningful graphic images on a CRT screen, thereby improving their ability within the domain of artistic creativity.

Acknowledgements

The authors wish to acknowledge the co-operation and stimulating ideas of Mr. K. Brunt, who wrote the programs to set up the data within the chemical elements data base; Mr. J. S.

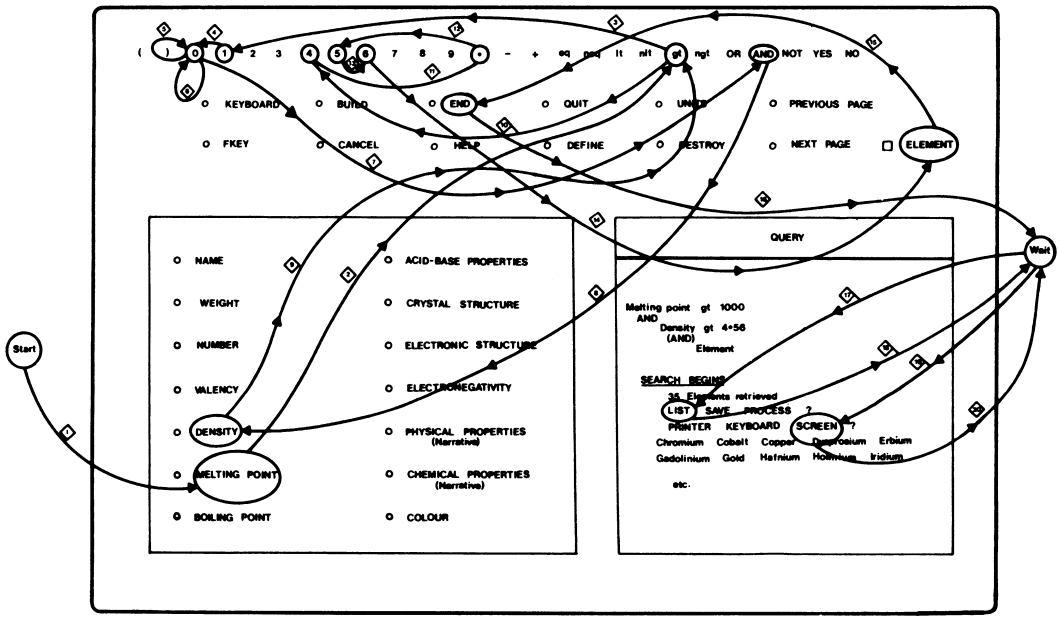


Fig. 7 Graphic formulation of a Type 2 query by menu selection

Roper for much helpful criticism and many suggestions, particularly with respect to the relational model of data and the computer system which was used to implement the data base; Mr. S. J. Tunstall, who implemented the parsing algorithm for syntax checking molecular formulae and who also interfaced it to the supervisor/data base system; and Bell Telephone Laboratories for providing us with a copy of the UNIX system and associated documentation. The 'ed' and 'roff' features of the UNIX system were extensively used in preparing all but the final draft of this paper.

References

AUSTIN, D. M. and HOLMES, H. H. (1972). PICASSO: A General Interactive Graphics Modelling Program, AEC Contract No. W-7405-eng-48, Lawrence Berkely Labs, Berkely, California, January 1972.

AYSCOUGH, P. B. (1976). CAL—boon or burden?, *Chemistry in Britain*, Volume 12, Number 11, pp. 348-352.

BAEKER, R. M. (1969). Picture Driven Animation, *Proc. of the SJCC*, AFIPS, Volume 34, pp. 273-288.

BARKER, P. G. (1975). Syntactic Definition and Parsing of Molecular Formula, Part 1: Initial Syntax Definition and Parser Implementation, *The Computer Journal*, Volume 18, Number 4, pp. 355-359.

BARKER, P. G. (1976a). Information Retrieval by the Method of Stepwise Refinement, Interactive Systems Group Working Paper.

BARKER, P. G. (1976b). A Historical Information System, *Computer Bulletin* Series 2, pp. 24-27.

BARKER *et al.* (1978). Implementing a Chemical Data Base using a Relational Approach, forthcoming.

BENZ, C. (1974). *Human Engineering in Process Automation*, Siemens Forsch, -u. Entwickl. -Ber. Bd 3 Nr. 5, Springer-Verlag.

BITZER, D. (1976). The Wide World of Computer Based Education, in *Advances in Computers*, Volume 15, Eds: M. Rubinoff and M. C. Yovits, Academic Press, New York, pp. 239-283.

BORK, A. (1974). 'Learning, Computers and Pictures', *THE Journal*, Volume 1, Number 5, pp. 6-13.

BOYCE, R. F., CHAMBERLAIN, D. D., KING, W. F. and HAMMER, M. M. (1973). Specifying Queries as Relational Expressions: SQUARE, *Proceedings of the ACM SIGPLAN-SIGIR Interface Meeting*, Gaithersburg, MD, November 1973. See also: Research Report RJ 1291, IBM Research Laboratory, San Jose, California, October 1973.

CHAMBERLAIN, D. and BOYCE, R. (1974). SEQUEL: A Structured English Query Language, Proc. 1974 ACM-SIGFIDET Workshop on *Data Description Access and Control*, Ann Arbor, Michigan.

CODASYL (1976). Selection and Acquisition of Data Base Management Systems, A Report of the CODASYL Systems Committee, page 88, March 1976.

DIGITAL EQUIPMENT CORPORATION (1974a). RT-11 System Reference Manual, Document DEC-11-ORUGA-C-P, DN1, DN2.

DIGITAL EQUIPMENT CORPORATION (1974b). BASIC RT/11 Language Reference Manual, Document DEC-11-LBACA-D-D, page J-17.

FELDMANN, R. J., HELLER, S. R., SHAPIRO, K. P. and HELLER, R. S. (1972a). An Application of Interactive Computing: A Chemical Information System, *J. Chem. Doc.*, Vol. 12, pp. 41-47.

FELDMANN, R. J. and HELLER, S. R. (1972b). An Application of Interactive Graphics—The Nested Retrieval of Chemical Structures, *J. Chem. Doc.*, Vol. 12, pp. 48-54.

HELD, G. D., STONEBRAKER, M. R. and WONG, E. (1975). INGRES—A Relational Data Base System, *AFIPS Conference Proceedings*, National Computer Conference, pp. 409-416, May 19-22.

HELLER, S. R., MILNE, G. W. A. and FELDMANN, R. J. (1978). A Computer Based Chemical Information System, *SCIENCE*, forthcoming.

KOENIGSBERG, L. and THANHOUSER, N. (1974). A Graphics System for APL Users—APL/GRAPH II, *Proceedings of the Sixth International APL Users Conference*, pp. 528-543.

KULSRUD, H. E. (1968). A General Purpose Graphic Language, *CACM.*, Volume 11, Number 4, pp. 247-354.

MCDONALD, N. and STONEBRAKER, M. R. (1975). CUPID—The Friendly Query Language, Paper presented at the 16th IFIP/IAG Data Base Workshop, Brussels, 14-17 December 1975.

MUNN, R. J. (1965). A Computer Based Information Retrieval System, *J. Chem. Ed.*, Volume 62, pp. 662-663.

NAKE, F. and ROSENFELD, A. (Eds.). (1972). *GRAPHIC LANGUAGES*, Proceedings of the IFIP Conference on Graphic Languages, Vancouver, Canada, 22-26 May 1972, North Holland Publishing Company.

NARASIMHAN, R. (1966). Syntax-directed Interpretation of Classes of Pictures, *CACM*, Volume 9, Number 3, pp. 166-173.

PRINCE, P. M. (1971). *Interactive Graphics and Computer Aided Design*, Adison-Wesley Publishing Inc.

RITCHIE, D. M. and THOMPSON, K. (1974). The UNIX Time Sharing System, *CACM*, Volume 17, Number 7, pp. 365-375.

SENKO, M. E. (1976). DIAM II with FORAL LP: Making Structured Queries with a Light Pen, Research Report RC 6034 (No. 26141), IBM Thomas J. Watson Research Centre, Yorktown Heights, New York 10598.
SMITH, S. G. and SHERWOOD, B. A. (1976). Educational Uses of the PLATO System, *SCIENCE*, Volume 192, pp. 344-352.
UNIVERSITY OF DURHAM. (1976). Report of the Vice-Chancellor and Warden for the Year 1975-76, page 24.
WELLER, D. and WILLIAMS, R. (1976). Graphic and Relational Data Base Support for Problem Solving, Proc. 3rd Annual Conference on Computer Graphics, Interactive Techniques and Image Processing—SIGGRAPH '76 July 14-16, University of Pennsylvania, Ed: U. W. Pooch, *ACM Siggraph Publication*, Volume 10, No. 2, pp. 183-190.
WILLIAMS, R. (1974). On the Application of Relational Data Structures in Computer Graphics, *Information Processing* '74, Proc. IFIP Congress 74, Stockholm, Sweden, August 5-10, Ed: J. L. Rosenfeld, pp. 721-726.
ZLOOF, M. M. (1975). Query by Example, *AFIPS Conference Proceedings*, Volume 42, pp. 431-438.

Book reviews

Computational Methods of Multivariate Analysis in Physical Geography, by P. M. Mather, 1976; 532 pages. (*John Wiley*, £13·75)

It is a little unfortunate that this worthwhile book, which is written by a geographer, will only be read by geographers! The restriction imposed by the title is unjustified when so many of the multivariate techniques covered have wide application. There are four parts dealing with the essentials of matrix algebra, the analysis of dependence, the analysis of interdependence and classification. In each part there is some attempt at elementary level instruction but this has to make way for the advanced references to recent research papers. These constitute conflicting aims which the author does not resolve and the research orientation predominates.

A third aspect of the book is the use of computer programs including the data and results for illustrative examples. Here the author is more successful and these will prove useful to the practitioner and will certainly give confidence to the user to develop new programs. Particularly valuable are the warnings and general wisdom contained in the papers and the excellent up-to-date bibliography which covers regression methods, trend surface analysis, factor analysis, classification and discriminant methods. The nonlinear least squares section is well done but the binary discriminant section is regrettably thin.

ROBERT W. HIORNS (Oxford)

Software Engineering and Education: Needs and Objectives, edited by A. I. Wasserman and P. Freeman, 1976; 159 pages.) *Springer-Verlag*, $9·80)

This volume contains the proceedings of a one day Interface Workshop held at the University of California, Irvine early in 1976. Like Hermann Goering, who is credited with the remark 'when I hear the word culture I reach for my revolver', my heart sinks when I see the word 'interface' used in a nontechnical context. However, despite its pretentious title, this volume makes interesting reading, since it records the views of industrialists and educators on the requirements of education in software engineering. Some seventeen people took part, and since the proceedings run to almost one hundred and sixty pages, it is clear that much of what it presented must be 'position papers' prepared in advance of the meeting and not presented as such at the workshop. Nonetheless, they are still interesting.

The first section presents the industry view of what is required of education in the field of software engineering. The overall impression one gains is that there is no consensus of opinion as to what software engineering is, never mind how it should be taught. As Jim Horning says '. . . many people have confused the existence of the term with the existence of a discipline', though Bob Graham is perhaps too cynical when he says that '. . . when industry gets a new graduate, the biggest task is to housebreak the student, that is to kill his spirit so that he can fill out the forms every week . . .'.

The second part of the book covers the approach of the academics to education in software engineering. They evidently have a clear idea of what the subject is (more precisely, several different clear ideas), and the overall impression is of a group of people who perceive a problem and are vigorously attempting to solve it. There is much of interest here for anyone involved in computer education. The third section of the book is a record of the discussions, in which the participants 'exchanged many valuable views in their respective languages'. Verbatim records of discussion, even when edited, do not make good reading, but it is worth persevering with the prejudices and preconceptions for the occasional nuggets of insight and wisdom.

There is still a great confusion between computer science and software engineering. One contributor has as a title 'Software Engineering as a Central Computer Science Discipline', and in the 'Annotated Software Engineering Bibliography', Knuth's *magnum opus The Art of Computer Programming* is described as being 'a necessary part of the library of every serious computer scientist'. Despite these confusions, this volume is an important step in the definition of software engineering, and should be on the bookshelf of anyone concerned with the development of computer education.

D. W. BARRON (Southampton)

Business Data Processing Systems, second edition by L. S. Orilia, Nancy and R. Stern, 1977; 404 pages, workbook 88 pages. (*Wiley*, £10·70)

This is a readable book written for American computer science courses and would be suitable for the more advanced data processing syllabus of computers and business studies courses in the UK. The monetary examples and problems are in dollars and the text and illustrations orientated towards IBM. The structure and sequence of the chapters appears to be illogical, e.g. the chapters on fundamental concepts (3) and problem definition (14) would be appropriate as 1 and 2, preceding systems studies and the systems analyst. In the systems studies chapter it was noted that examples of a payroll listing and exception report were illustrated and discussed, and the systems analyst chapter contained a section on data collection which in reality was some techniques of fact finding.

It was disappointing to find no chapter devoted to the existence or need for standards. The definition of a feasibility study on page 4 relating to 'whether a company should acquire basic data processing equipment' is too restrictive. The systems flowcharts on page 154 *et seq.* fail to depict an 'edit' (data-vet) program and the input cards are shown as being mechanically sorted to the Master File sequence. There are other criticisms of detail, some of which are misleading unless pointed out. However, the attempt to be 'all things to all people' can be said to be somewhat successful. The price would probably be unacceptable to students, but the book would be a useful addition to the library of books on systems analysis and design. The general layout, presentation and illustrations are clear and to a high standard. The end of chapter questions, glossary of terms used, two case studies and a Student's Workbook (extended end of chapter questions) would help to reinforce the learning.

C. POTTER (Brentwood)