Int. J. Man-Machine Studies (1981) **15**, 87–134

Consistency and compatibility in human–computer dialogue

P. J. BARNARD, N. V. HAMMOND, J. MORTON AND J. B. LONG

Medical Research Council Applied Psychology Unit, 15 Chaucer Road, Cambridge CB2 2EF, U.K.

AND

I. A. CLARK

IBM UK Research Laboratories, Hursley Park, Winchester SO21 2JN, U.K.

(*Received 21 March 1980, and in revised form 18 January 1981*)

To tackle problems of human–computer interaction the traditional scope of human–machine studies needs extending to include the complex cognitive skills of understanding, communication and problem solving. This extension requires a fusion of the conceptual and empirical tools of human factors with those of cognitive psychology. A methodological approach to this fusion is outlined as a background for three studies of structured human–computer dialogue. The studies involved a task in which secret messages were decoded in a number of discrete steps corresponding to computer commands. Each "command" required two numeric arguments. The study investigated underlying variables using questionnaire techniques in addition to user performance in an interactive version of the task. Three factors concerning the order of arguments in a command string were investigated: the consistent positioning of a recurrent argument, the relationship between argument entry order and their order in natural language, and the relationship between argument entry order and the position of argument values on a VDU. In Study I software specialists were asked to design command structures for the task and to give reasons for their choices. In Study II naive subjects were asked to choose between telegrams in which alternative argument orders were expressed in terms of alternative word orders. In the interactive version of the task, used in Study III, positionally consistent systems were most readily learned, but this depended on having the recurrent argument in the first position. With positionally inconsistent systems there were reliable effects due to the position of the direct object of individual command verbs.

1. Introduction

1.1. BACKGROUND

The range of new applications for interactive information and decision support systems is broadening with extraordinary rapidity. Few would dispute the prediction (e.g. Alloway, 1979) that the rate of development will continue to increase throughout the 1980s. Although a sizeable proportion of the end-user population will be skilled personnel (e.g. programmers or transaction clerks), an increasing proportion will be individuals who will have little or no specialist skills in computing. For these individuals, interactive systems will simply be one type of resource which they can call on while pursuing the wider objectives of their occupation or profession (cf. Codd, 1974; Fitter,

0020-7373/81/050087+48$02.00/0

1979). This class of user has variously been described as *general* (Miller & Thomas, 1977), *naive* (Kennedy, 1975), *casual* (Codd, 1974), *discretionary* (*Bennett*, 1979) *or occasional* (Hammond, Long, Clark, Barnard & Morton, 1980). In spite of growing concern for the "usability" of systems intended for this type of end-user population (e.g. Engel & Granda, 1975; Fitter, 1979; Gaines & Facey, 1976; Martin, 1973; Miller & Thomas, 1977), past and present systems frequently fail to incorporate an interface which can be mastered easily (e.g. Bennett, 1972, 1979). They also often fail to meet the end-user's wider job needs and expectations (e.g. Eason, Damodaran & Stewart, 1974; Hammond, Long & Clark, 1978).

The design and development of user-friendly interactive systems which meet the end user's job needs pose both methodological and conceptual problems for research on human–computer systems. Historically, research on human–machine systems has been conducted within three approaches: "classic" ergonomics, "systems" ergonomics and "error" ergonomics (Singleton, 1976). Classic ergonomics stresses systematic evaluation of individual variables typified by the "knobs and dials" approach, and its accumulated literature has obvious extensions to computer peripherals such as keyboards and displays (for example, see Miller & Thomas, 1977). In contrast, the systems approach stresses the integration of individual, organizational and societal factors (see Chapanis, 1969), and as such offers a wider perspective on computerization (see Chapanis, 1979). The more limited aim of error ergonomics, to eliminate successively the problematic features of interfaces, also offers short-term procedures for enhancing the usability of new interfaces. However, these approaches were developed primarily in response to the automation of mechanical control and information display. With interactive computing systems there are important differences in the nature of the tasks being automated. The control and display of physical systems is being superseded by the manipulation and display of conceptual ones. While, for example, the attributes of keyboards and VDUs (Visual Display Units) will continue to be of major ergonomic concern, they simply mediate the exchange of information between user and system. At present, user–system *dialogue* is poorly understood (Gaines, 1978; Baecker, 1979) as are the wider issues associated with more general system characteristics (Miller & Thomas, 1977). Essentially, ergonomics must be extended to analyse both the nature of the user's dialogue with a system and his understanding of the information which it can manipulate and make available to him. To be truly "usable", a system must be compatible not only with the characteristics of human perception and action but, and more critically, also with users' cognitive skills in communication, understanding, memory and problem solving.

It must be acknowledged at the outset that the analysis of these cognitive activities is highly complex even in relation to the kind of simple tasks characteristically employed in behavioural research. Where information must be understood and used in more practical contexts, such as application forms, government leaflets and instructions for the use of machinery, behavioural studies suggest a general pattern in which the "cognitive compatibility" of information is multiply determined by a range of factors (e.g. Barnard & Marcel, in press; Wright, 1978). These include the structure and content of the information itself as well as the characteristics of the individual, his knowledge and the task he is required to perform. Similar implications can be drawn from research specifically concerned with the understanding and use of programming languages (for example, see Green, 1980).

Against this background, any ergonomic approach which attempts to formulate guidelines or design principles on the basis of highly selective parametric studies is likely to have inherent limitations. For example, even simple "cognitive" guidelines for presenting written information are subject to significant qualifications (e.g. Wright & Barnard, 1975; Wright, 1980). The situation is unlikely to be different in the dynamic and complex environment of interactive information and decision support systems. Indeed, the complexity of these environments and the limitations of ergonomic support have undoubtedly contributed to a viewpoint in computing science that ergonomics can at best serve the function of eliminating misfits (e.g. Brooks, 1977). However, this viewpoint merely serves to underline the inadequacy of the current state of affairs in both computer science and ergonomics.

A more productive approach would involve analysing the requirements for ergonomics with a view to developing new strategies for research and its relationship to system design. These research strategies must be sufficiently rich to encompass users' cognitive skills without losing sight of the objective of providing a practical basis for ergonomic support in the process of design and development.

1.2. A RESEARCH APPROACH

One strategy developed with these aims in mind is represented in Fig. 1 (Morton, Barnard, Hammond & Long, 1979). The main objective is to analyse cognitive incompatibilities between man and computer in a manner that permits designers' efforts to be directed productively but not proscriptively. The strategy itself requires development of both the conceptual and the empirical tools of behavioural research. For example, the role of the conceptual tools is much the same as the analytic support provided by physics, metallurgy and aerodynamics in the development of aircraft. The role of the empirical tools is to provide valid evaluative procedures both for the introduction of new systems and for proving the conceptual tools. As such they fulfil a function not unlike systematic observation of the performance of existing aircraft,

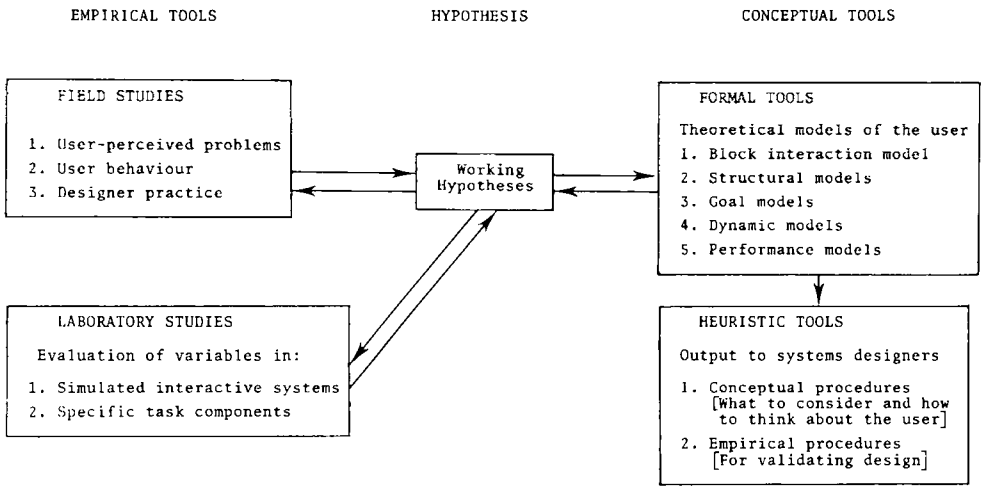


FIG. 1. Research strategy: the relationship between empirical and conceptual tools. (Adapted from Morton *et al.*, 1979.)

the evaluation of alternative configurations simulated in a wind tunnel and proving flights with prototypes.

The strategy outlined in Fig. 1 stresses the integration of different types of empirical evidence and different analytic methods. The empirical component of the strategy draws on the tools of classic, system and error ergonomics as well as on those of experimental psychology. As with system ergonomics, field studies (e.g. Hammond *et al.*, 1979, 1980) attempt to identify and integrate user-perceived problems, user behaviour and designer practice. As in classic ergonomics and experimental psychology, laboratory studies play a complementary role, with the aim of providing systematic manipulation of variables which may influence user behaviour. Working hypotheses serve to integrate evidence and to direct empirical evaluation. In view of the complexity of cognitive behaviour, individual studies of particular applications may not furnish simple implications for design of new systems involving different applications. In consequence, the research strategy stresses an interpretive route to the application of evidence. In this respect, the working hypotheses have the second function of integrating empirical and analytic tools, where the analytic tools are required for the interpretation of evidence.

Just as a range of empirical tools is invoked, no single cognitive theory is viewed as being sufficiently rich to capture all the salient features of users' understanding, memory, knowledge and task performance. Accordingly, a range of conceptual models are brought to bear on the interpretation of different aspects of user behaviour. The pragmatic aim of this interpretive process is to provide heuristic tools for the designer (e.g. Card, Moran & Newell, 1980; Moran, 1978). Heuristic tools of an analytic nature should enable the designer to conceptualize users' cognitive behaviour, with a view to identifying an appropriate and narrow range of solution paths for a particular application. Once determined, a narrow range of design options can be tested empirically and evaluated at relatively low cost in the process of system design and development.

1.3. SPECIFIC AIMS OF THE INVESTIGATION

Within this framework, the investigation aims to address both methodological and substantive issues. One methodological aim is to try out an interactive software environment for modelling different system characteristics while keeping the task conditions unchanged. The actual task environment selected will be outlined in section 2. A second methodological aim is to assess users' interactive performance in relation to questionnaire measures of their natural language predispositions and of the design predispositions of a sample of software specialists. The general objective of the empirical techniques is to explore and develop working hypotheses concerning the underlying factors which govern an occasional user's dialogue with an interactive system. This dialogue is almost invariably constrained by current restrictions on computer "understanding". In the case of command languages, there has always been pressure to base their characteristics on natural language. However, the problems in this area are substantial (for example, see Hill, 1972) and, as Miller & Thomas (1977) point out, command languages are likely to continue for some time as pidgin dialects.

Under these circumstances various aspects of a user's knowledge could potentially be brought into play in the context of a specific application (e.g. medical diagnosis, town planning or accountancy). Figure 2 shows one simple way of representing

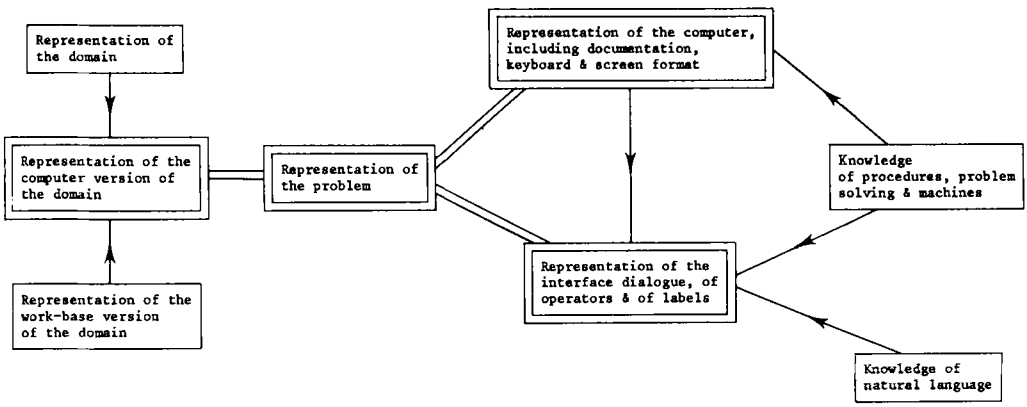


FIG. 2. The Block Interaction Model. The blocks with double boundaries, connected by double lines indicate the blocks of information used by the ideal user; other blocks and lines indicate potential sources of interference or facilitation.

interrelationships between the types of knowledge a user may recruit in the course of human–system dialogue. In this representation, a Block Interaction Model (Morton *et al.*, 1979), an attempt is made to characterize potential kinds of interference between types of knowledge which could give rise to cognitive incompatibility at an interface. This gives a way of classifying the errors which occur in system use (e.g. Hammond *et al.*, 1980). It is not intended to serve any more than the most superordinate of taxonomic functions. There are a number of ways of expanding this model, but the appropriateness of any expansion would be a function both of the system in question and for the particular questions under consideration.

In Fig. 2, the blocks with double boundaries represent the kinds of knowledge an "ideal" user would recruit to solve a problem. This user would have perfect knowledge of the way in which the domain of application was represented in the computer. For the less-than-ideal user, knowledge of some other specific means of solving the problem, such as a paper and pencil work base, and other more general means of representing aspects of the domain of application, are potential sources of cognitive incompatibility in relation to the actual representation of the domain in the computer. Thus, a user found to be indulging in complex checking procedures after using the command FILE turned out to be perplexed that the material filed was still present on the screen. With pieces of paper, things which are filed are physically transferred rather than being copied. Specific attributes of the system environment may also conflict with users' general knowledge of systems, machines and procedures. Likewise, day-to-day use of natural language or a current display of information may be incompatible with the operations, command terms and syntax incorporated into a specific applications language. The present investigation will focus on the development of working hypotheses involving the last group of factors.

In this context initial working hypotheses can be formulated in terms of the notions of consistency and compatibility. These terms are potentially confusable, so the distinction requires clarification. In the present context, they can be defined in relation to the blocks of Fig. 2. Consistency will be used to refer to relationships *within* a type of knowledge represented by a single block. Hence, if, for example, components of

a user's representation of operators and labels were to conflict in some way, then the conflict would be termed as inconsistency. In contrast, compatibility will be used to refer to relationships *between* blocks in Fig. 2. Thus, natural language representations might be incompatible with the representation of operators and labels. However, unsystematic labelling of operations, represented in a single block, would be described as inconsistent rather than incompatible. An example of this is the inconsistent actions required to perform the same operation of removing an unwanted character. When operating the editor on a certain system, the user must press a key labelled DELETE, while when using an interactive language, the user must press a BACKSPACE key. Either label alone might be compatible with a natural language representation of the action, but the labels are inconsistent when both occur in the same system.

Consistency and compatibility are generally acknowledged to be basic and useful ergonomic principles. A system which provides the user with a consistent representation within a particular type of knowledge is likely to be easier to learn and less prone to error than an inconsistent one. Similarly, incompatibilities between system characteristics required for ideal operation or between system characteristics and other related types of knowledge in the possession of the user are also likely to have a price in time and accuracy. The Block Interaction Model specifies four potential influences on a user's representation of the "ideal" classes of knowledge. Several empirical questions can be asked concerning their interactions.

First, what are the important issues in relation to the *internal* consistency of a set of commands?

Second, to what extent can general knowledge be used to induce the principles which underlie command structure?

Third, what are the important issues in relation to the compatibility of commands with natural language?

Fourth, what are the important issues concerning the user's knowledge of the machine, where sources of information may include documentation or specific information displayed on a VDU?

The potential importance of these questions can be illustrated by reference to command languages in which a reserved command word is followed by a number of arguments (e.g. DELETE ⟨LINE NUMBER⟩ ⟨ENTITY⟩). The problem of internal consistency can most readily be discussed in relation to *positional* formatting where the arguments must be entered in a fixed order. For example, in some text editors the line number must be specified as an argument for each of a number of the available operations. One form of consistency would involve always specifying this argument in the same position, that is the LINE NUMBER would always be entered as the first argument (or the second) of any command for which it is required. This *positional* consistency would mean that a user would have to remember the identity of each command's arguments but could make use of positional consistency to determine their order. In contrast, if the position of the LINE NUMBER varied across commands (DELETE ⟨LINE NUMBER⟩ ⟨ENTITY⟩ and CHANGE ⟨ENTITY⟩ ⟨LINE NUMBER⟩) the user would have to remember argument identity and position for each command.

Adopting a positional principle for the internal consistency of a set of related commands may, however, be incompatible with other factors. In particular, positional

consistency may be incompatible with natural language representations of the command string. In the example DELETE ⟨LINE NUMBER⟩ ⟨ENTITY⟩, the natural language expansion would be "Delete from line X, entity Y". In this case positional consistency in which the LINE NUMBER is entered as the first argument would conflict with the natural language tendency to express the direct object of a verb before the indirect object (Delete entity Y from line X). Obviously, the flexibility of natural language enables these orders to be varied, and ambiguities can still occur with double object verbs, for example transactional verbs such as take or show ("Show the man the picture" and "Show the picture to the man"). However, the "direct object first" scheme would appear to be a useful rule of thumb for the present investigation.

Irrespective of the particular conceptual principle used to organize a set of commands, the ordering itself may be incompatible with other forms of information dependent on the user's representation of the machine. In the task environment, for example, a user's knowledge may include information concerning the structure of a display of information. This might contain argument values appropriate for the particular point reached in the dialogue. Items might be presented in such a way that natural search or reading strategies conflict with the way in which the information must actually be entered into the system. Parameter information could occur in some constant screen format where the relative left–right position of two arguments is the opposite of their ordering for command entry. There is, for example, evidence that alternative layouts for information on a VDU affect performance in search tasks (Barnard, Morton, Long & Ottley, 1977) and data entry (Long, Dennett, Barnard & Morton, 1977).

An observational study of a positionally formatted applications language identified both item and order specification as sources of difficulty for occasional users (Hammond *et al.*, 1980). Obviously, in some systems problems could be avoided by the use of *keyword* formatting in which arguments can be entered in any order and then identified and interpreted by the system according to their semantic class. However, keyword formatting is impractical in some situations and can bring with it additional problems for the user. We are currently investigating some of these. For the research purposes of exploring general components of consistency, positional formatting has the advantage of enabling both item and order errors to be taken into account in the assessment of competing influences on user performance. Accordingly, the present study focussed on positional formatting and made provision for an empirical evaluation of the relative influences of positional consistency, natural language compatibility and compatibility between a displayed order of arguments and their order of entry.

The above analysis makes some practical assumptions concerning the relevance of specific positional and natural language variables. In the case of the internal consistency of a set of commands, system designers and experienced programmers may well consider positional consistency to be of only minor importance and use other organizing principles which could prove important in the assessment of user performance. Accordingly, one part of the investigation (Study I) will examine the principles adopted by designers in organizing argument sequences. Similarly, the users themselves may not always agree upon a "natural" ordering for a sequence of arguments. The assumption concerning the direct object–indirect object ordering, for example, may not extend from natural language to the kind of "telegraphese" notation of command strings. In addition it seems likely that the potency of any "natural" order will vary within a set

of commands. Thus, the command DELETE may involve stronger ordering constraints than SEARCH ("search X for Y", "search for Y in X"). In view of these possibilities, a further investigation (Study II) will attempt to assess the ordering constraints which govern users' own judgements as to the "naturalness" of argument orders. The final and major part of the investigation (Study III) will examine users' behaviour on-line in an interactive task where the three factors, positional consistency, natural language compatibility and display compatibility, are systematically varied. Since each of the three parts of the investigation involves features of the task used in the interactive study, the basic task will be outlined in the next section to avoid later repetition.

2. The basic task

2.1. OUTLINE

For the purposes of interactive experimentation, the task and the software system used to implement it must incorporate a number of characteristics. First, the interactive task must reflect critical features of actual systems, including structured human–system dialogue in a non-trivial task environment. Second, the system and the task environment must be sufficiently flexible to allow a range of system, command, object and user variables to be manipulated under broadly comparable circumstances. Third, the interactive implementation must allow measurement (times, errors, help calls) at a level appropriate to the class of variable being manipulated. This last requirement is confined to the interactive version of the task to be used in Study III (see section 5). However, the other two requirements concern the conceptual operations and commands of the basic task which will also be central to the non-interactive tasks of Studies I and II.

The domain of application, which was chosen to form the basis of several investigations of the present type, involved decyphering coded messages. This domain was selected primarily for its flexibility in modelling various features of interactive systems such as data base enquiry, string manipulation, text editing and file handling. By their very nature cyphered messages can also be encoded in relation to a wide range of conceptual objects (e.g. texts, arrays, visuospatial entities such as maps or pictures, and even on waveforms). In addition, the task of decoding can be broken down into a sequence of clearly definable steps or conceptual operations which can be equated with individual commands. Finally, unlike many applications such as accountancy or computer-aided design, decyphering can be understood by a population of naive users with minimal experience of both interactive systems and the domain of application.

Each of the three studies reported in subsequent sections involved individual messages encoded in a simple substitution cypher. For any given message the same ordered sequence of six steps was always involved. These steps included obtaining, decoding and storing the message. Each step defined a conceptual operation realized by a command structure in which an imperative verb was followed by two numeric arguments. One of the two arguments was common to all six conceptual operations (a message identifier) although its numeric value varied as the task progressed. The identity of the other argument varied according to the specific operation. In order to avoid results being attributed to individual command names, Studies II and III made use of two different sets of six verbs to refer to the operations. The actual sequence of operations required to complete the task for each message is described below. The

description includes (a) the two alternative command names, (b) the context, (c) the operation performed, (d) the variable argument required and (e) the result of the operation. Appendix 1 shows the sequence of screen displays from one of the training problems of Study III, in which the operation of the commands are illustrated.

2.2. THE SIX TASK STEPS

STEP 1

Command Names: *SEARCH or SELECT*

Context: To start the sequence, a cyphered message is required. The message identifier (the common argument) is known.

Operation: The command fetches a cyphered message from a file and displays it.

Variable Argument: The number of the file from which the message has to be fetched. This is displayed explicitly.

Result: The message is available on the display as a segmented sequence of characters, for example:

HAYQ 62;* 6J2J ALI; 6KWJ IJE; AOWJ 2;QH QJI6 E:;D AJAE AJ9K ; JAF

In the actual interactive task of Study III, the message identifier is automatically incremented by one after *each* task step.

STEP 2

Command Names: *TRIM or LINK*

Context: The message is displayed in equal sized segments of characters. The size may be 3, 4 (as above), 5 or 6 characters in each segment. These must be normalized into a single, unified string.

Operation: The command joins together the segments by closing up the spaces in the message.

Variable Argument: The size of the segment, in characters. This must be obtained by observing the segments actually displayed.

Result: Spaces disappear to give the unified string, e.g.:

HAYQ62;*6J2JALI;6KWJIJE;AOWJ2;QHQJI6E:;DAJAEAJ9K;JAF

STEP 3

Command Names: REPLACE or CONVERT

Context: A unified string of cyphered characters encoded in a known substitution code.

Operation: The command substitutes each cyphered character with the appropriate character according to the specified code (identified by a code number).

Variable Argument: Code number to be used (displayed explicitly).

Result: A new string of characters, which does not yet resemble normal English, is displayed, e.g.:

GEPUSA :S5A5EHT SNO5T5R EMO5A UGU5TSR* FE5ERE5CN 5E4

STEP 4

Command Names: INVERT or REVERSE

Context: The string represents a cypher in which groups of characters are in backward order in relation to normal English. The size of the backward grouping varies (3, 4 or 5).

Operation: The command takes each backward group and re-orders the characters into the appropriate left to right sequence.

Variable Argument: The size of the backward groups (displayed explicitly).
Result: A re-ordered sequence of characters which resembles normal English but which contains random occurrences of a single digit:
PEGASUS: 5A5THENS T5O R5OME A5UGUSTS *R5EFERENC5E5 4

STEP 5
Command Names: DELETE or ERASE
Context: The decoded message contains repeated occurrences of an unwanted digit.
Operation: The command removes each occurrence of the unwanted digit and closes up the spaces which would otherwise have been created.
Variable Argument: The digit to be removed.
Result: The message is now fully decoded, e.g.:
PEGASUS: ATHENS TO ROME AUGUST *REFERENCE 4

STEP 6
Command Names: SAVE or STORE
Context: To complete the sequence the decoded message must be filed.
Operation: The message is removed from the screen and stored in a new file for decoded messages.
Variable Argument: The position in the file, given by the reference number in the decoded message.
Result: The sequence is successfully completed. (In the interactive task of Study III, confirmation appears on the VDU and the sequence starts again for a new message.)

3. Study I: designer practice

3.1. AIMS AND METHODS

The objective of this first study was to establish the extent to which software specialists consider the consistency issues raised in the introduction (section 1.3). System designers and systems programmers may not only vary in their choice of command structure but they may also be constrained by considerations other than those of positional or natural language consistency. They may, for example, give priority to logical or system structures as primary determinants of an appropriate form of argument ordering in commands.

A questionnaire technique was used to assess the predispositions of a sample of software specialists. They were invited to "help design a system to be used for decoding secret messages using a visual display unit". The questionnaire was divided into four sections. The first part consisted of a brief description of the system to be designed and instructions for filling in the questionnaire. The second part described in detail each of the tasks steps given above (section 2.2) and the designer was asked to supply command names and argument orders. The third part asked the designers to give reasons for their choices, and the final part assessed their experience of interactive systems.

3.2. SUBJECTS

Questionnaires were sent to about 80 computer professionals (systems designers, systems and applications programmers and computer managers; referred to collectively

as "designers") in seven organizations. Replies were received from 49 of the sample (61%). Of the designers who completed the section of the questionnaire dealing with their computer experience (46), all had used interactive systems, 45 (98%) had written programs, 42 (91%) were familiar with more than one interactive system, 40 (87%) had themselves written or modified an interactive system, 34 (74%) had designed or helped design an interactive system and 21 (46%) had taught about or published papers on interactive systems.

3.3. QUESTIONNAIRE DESIGN

In the first part of the questionnaire, the outline of the system stated that the syntax for the commands had already been decided, and was to take the form of "COMMAND ARG1 ARG2". However command names were yet to be decided as was the order of arguments for each command.

The second part of the questionnaire was divided into six sub-sections, one for each command (see section 2.2). The sub-section consisted of a description of the operation itself, an example of a message before the operation of the command and the same message after the operation of the command, the labels of the arguments used with the command (the order of the two labels alternated on successive commands) and a space for the designer to enter his solution. For example:

COMMAND FIVE

All instances of the specified digit are removed from the specified message.
Example: Before—SEC9R9ET I9RA 9CO9DE
After—SECRET IRA CODE

Arguments: Digit to be removed, Message identifier

)))Specify COMMAND ARG1, ARG2:

Thus if the designer thought the best word for this operation was OMIT and that the first argument should be the digit, he would write after the last line "Omit Digit, Message id". It was stressed in the instructions that both the command name and the order of the arguments was to be specified. The order of the six operations in the questionnaire was the same as that in the interactive version of the task (i.e. steps 1–6 in section 2.2).

In order to control for any biases caused by wording, four versions of the questionnaire were prepared. The four differed only in the descriptions of each operation and the order in which the two argument labels were typed on the page. Since all operation descriptions referred to both arguments, it was possible that the order in which they occurred in the description ("digit" before "message" in the example above) might influence the order that the designer chose. For half the questionnaires, the descriptions were therefore re-worded so that the arguments occurred in the opposite order. For example, the alternative wording for command five was "The specified message has all instances of the specified digit removed". Likewise, the order in which the argument labels were given might influence the designers' choices. Therefore for half of the questionnaires with each description the argument labels were presented in the opposite order. For example the alternative ordering for command five was:

Arguments: Message identifier, Digit to be removed

Equal numbers of the four versions of the questionnaire were prepared and circulated to the designers.

3.4. RESULTS AND DISCUSSION

The designers' choices concerned both the naming of the command operations and the choice of order of insertion of the arguments. Only the latter will be reported here; results concerning command naming are to be published elsewhere.

For all six commands, one of the two argument labels was the message identifier. Thus designers could choose message identifier as either the first argument or as the second argument for each operation (details of the argument labels for each command are given in Table 3 of section 5). There was no evidence for reliable differences between the four versions of the questionnaire, and so the results were combined over the versions. In particular, there was no evidence of a correlation between the order of the two argument labels in the questionnaire and the arguments chosen by the designers. The numbers of designers choosing the argument order of message identifier first for each command is shown in Fig. 3. This order was favoured with

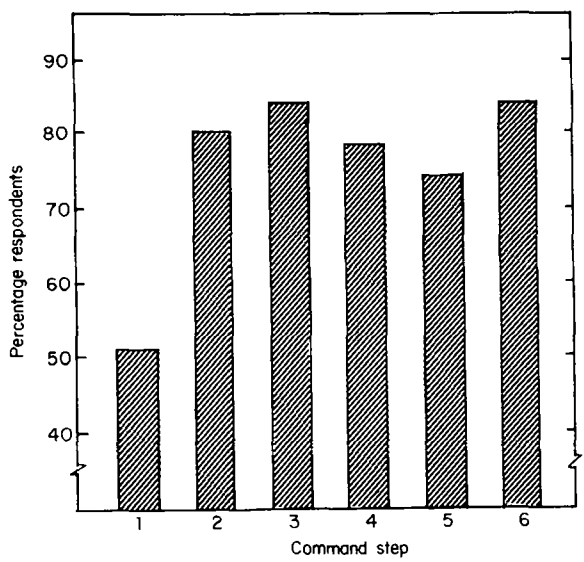


FIG. 3. Study I: Numbers of designers choosing argument order of "message identifier first" for each of the six commands.

every command, although the tendency was mush less pronounced with the first command than with any of the others. The bias in favour of message identifier as first argument was statistically reliable for commands 2–6, $p < 0{\cdot}01$ for each command (Binomial test), but not for command one. The pattern differed significantly over all six commands ($\chi^2 = 19{\cdot}65$, $df = 5$, $p < 0{\cdot}01$). There was, however, no reliable difference between commands 2–6 ($\chi^2 = 2{\cdot}26$, $df = 4$, $p > 0{\cdot}2$). Thus, the bias for command one differed from that of the other five commands, which did not vary among themselves. The reason for this difference in argument ordering for command one might be due

to a tendency for designers to conceptualize the file from which the message was to be taken as the most significant operand, rather than the message itself. This view is supported by the justifications given by the designers for their choices (see below).

A major point of concern was the extent to which designers made use of positional consistency, as this was a variable to be investigated in Study III. However, although designers tended to favour message identifier as first argument, they did not necessarily choose this order for all six of the commands. The majority of designers did design a consistent system (that is, message identifier always as the first argument or always as the second argument), but many did not. Figure 4 shows the numbers designing

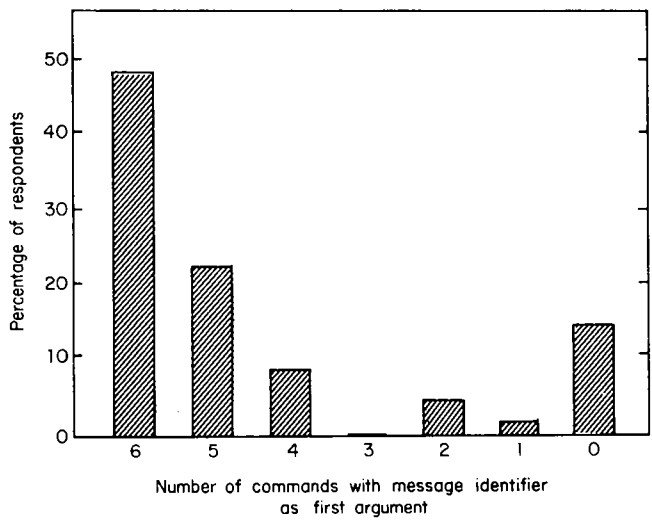


FIG. 4. Study I: Numbers of designers producing systems with the argument order "message identifier first" for six, five, four, three, two, one or none of the six commands.

systems with message identifier as the first argument for all commands, or for five, four, three, two, one or none of the commands. Forty-nine per cent of the designers chose message identifier as the first argument for all commands, and 14% placed it consistently second. In Fig. 4 this is represented as systems with no commands with the message identifier as the first argument. The remaining 37% designed mixed systems in which the message identifier was the first argument for some commands and the second argument for others. Therefore, while consistency of order appears to be an important principle for system designers, it is by no means the only one.

The reasons that the designers gave for their choices of argument orders shed further light on the organizing principles that they used. Justifications were categorized into seven classes. The numbers of designers using each class of justification are presented in Table 1. The table also breaks down the numbers in terms of the argument orders chosen: message identifier consistently as the first argument (Cons 1st), message identifier consistently as the second argument (Cons 2nd) or as the first argument for some commands and the second for others (mixed). Since some designers gave more than one justification, some gave none, and a few justifications could not be classified (e.g. "By instinct"), the percentages in the table do not sum to 100.

TABLE 1

Study I: Summary of justifications for choices of argument orders. Of the 49 respondents, 24 placed the message identifier consistently first, seven placed it consistently second and 18 designed mixed systems. Percentages indicate the proportion of designers within each order group referring to each justification category. See text for details of justification categories

	Argument orders chosen							
	Cons. 1st		Cons. 2nd		Mixed		Total	
Justification category	*N*	%	*N*	%	*N*	%	*N*	%
Consistency	15	63	7	100	8	44	30	61
Sentence order	3	13	0	0	5	28	8	16
Ease of use	2	8	1	14	0	0	3	6
Operand should be first	1	4	0	0	5	28	6	12
Most significant arg. first	0	0	1	14	3	17	4	8
Direction of data travel	0	0	0	0	5	28	5	10
Defaultable argument second	1	4	3	43	0	0	4	8

The most common justification was the principle of consistency; this was expressed in a variety of fashions. Examples in this class are "Always have the message id in the same position", "Common theme for first position", "Consistent position" and "System consistency". It is of interest that of the 18 designers who designed a mixed argument order system, eight gave consistency as a reason for so doing. Presumably some sort of system consistency other than positional consistency (such as in terms of classes 4, 5, 6 or 7, see below) was being invoked. A second class of reason given was related to the principle of natural language sentence order, one of the principles to be examined in the experimental study. Wordings of these justifications included "Sentence order", "Direct object first", "Good English", "Natural order" and "Naturalness". A third class was a general appeal to ease of use or usability. Wording included "Easy to remember", "Simplicity" and "Logical use". The remaining four classes were more concerned with system-oriented concepts; nineteen of the 60 justifications fell within these classes. A number of designers considered that the first argument should be the operand; wordings included "Order is operand, then qualifier", "What is worked on goes first" and "Operated object first". The fifth class of justification was that the most significant argument should be placed first. Wordings included "Key operation first" and "Command-related argument next to verb". A further class concerned the notion of direction of data travel; where relevant an argument specifying a source should be placed before an argument specifying a destination. Wordings included "Source then destination" and "Order of data travel". The final reason given for choice of order was that the second argument should be the one which could be defaulted or assumed by the system; since most (but not all) designers who gave this reason felt that the message identifier could be defaulted, it tended to be a justification for placing the message identifier second.

All in all, the designers differed widely in their choices of argument orderings (although positional consistency, particularly with the recurrent argument placed first,

tended to be favoured) and in the reasons given for their choices. Nevertheless, the two most frequent classes of justification clearly indicate that the two factors identified in the introduction are significant design issues and that they merit further empirical and conceptual analysis.

4. Study II: collateral measure of natural order

4.1. AIMS AND METHOD

The objective of this second study was to validate the natural language analysis offered in the introduction (section 1.3) by reference to a sample of potential users. In that analysis it was suggested that a normative order for arguments in a natural language expansion might most plausibly involve specifying the direct object of the command verb before the indirect object. The following study was carried out (a) to assess the extent to which a sample of subjects agreed amongst themselves on a "natural" order for arguments and (b) to assess the extent to which any such agreement might correspond to the direct object/indirect object analysis which formed the basis of the argument orders to be used in Study III (see Table 3 in section 5 for a listing of the argument orders).

The method used was analogous to that of the previous study and also involved a questionnaire. However, for the sample of subjects, no reference was made to computing systems. Instead they were asked to imagine that they were working for an intelligence agency. They were informed that they had to send a series of telegrams to an agent which would tell him how to decode some secret messages. These telegrams were phrased in "telegraphese" closely resembling the command language format used in Study I. Essentially the subjects were asked to choose between two altenative telegraphese strings representing COMMAND-VERB ⟨ARG1⟩ (direct object), ⟨ARG2⟩ (indirect object) and COMMAND-VERB ⟨ARG1⟩ (indirect object), ⟨ARG2⟩ (direct object).

4.2. SUBJECTS

The questionnaire was administered to 68 subjects: 44 female and 24 male. All were members of the Applied Psychology Unit's subject panel. Their mean age was 41 years. Subjects were tested in groups varying in size between nine and 17.

4.3. QUESTIONNAIRE DESIGN

There were two parts to this questionnaire. In the first part background information (see above) was given together with detailed instructions for completing the questionnaire. In the second part, as in the previous questionnaire, there were separate sub-sections for each of the six task steps of the decoding sequence (see section 2.2). For each step subjects were shown an example of how the message should change were the step carried our correctly, and were given a choice between two alternative telegrams. The subjects' task was to indicate which of the two telegrams would be more appropriate to send to the agent. All the information was provided on a short questionnaire on which the subjects filled in their choices. An example of one of the

steps is as follows:

Step 5
Message 7 *before*:
MEE9T + D9IM9ITRI + 9A9T + ST9ATI9ON + ON9 + FRID9A9Y
Message 7 *after*: MEET + DIMITRI + AT + STATION + ON + FRIDAY

Tick your choice of telegram:

☐ ERASE DIGIT 9 MESSAGE 7 ☐ ERASE MESSAGE 7 DIGIT 9

The subjects were required to tick one of the two possible telegrams for each of the six steps. The alternatives for individual operations differed only in the order of the two arguments. As can be seen from the example, both argument labels and explicit argument values were specified in the telegram options. Values were given both to help the subject understand the operation and to minimize any tendency to interpret the two arguments as a single compound noun (such as "erase message-digit").

In order to examine the two alternative sets of command names and to control for any biases caused by questionnaire layout, four versions of the questionnaire were prepared. The four differed only in the telegram options. Two questionnaires used one set of command labels (search, trim, replace, invert, delete, save) and two the other set (select, link, convert, reverse, erase, store). The order of the steps in the telegram task was the same as the order of the operations in the interactive task (Study III). The two arguments for each telegram were the two arguments associated with that command name (see Table 3 in section 5). For the two versions of the questionnaire within each command set, the position on the page of the alternative telegrams was balanced over the two versions. Thus, for one version the telegram with "MESSAGE" as the first argument was always the left-hand option, and for the other was always the right-hand option.

4.4. RESULTS AND DISCUSSION

Subjects tended to favour the telegrams on the right-hand column of the questionnaire regardless of argument order ($F[1,64] = 10{\cdot}69$, $p = 0{\cdot}002$). Fifty-eight per cent of the 408 choices were for the right-hand telegram, and this bias held for all but one of the 12 commands, varying between 50% and 68%. There was no evidence that the bias varied systematically between subjects using the two different sets of commands ($F[1,64] < 1$) or between the individual command words themselves ($F[5,320] < 1$), so the data were combined over the two versions of the questionnaire within each command set to balance out the effect of the bias.

It is possible that subjects were influenced in their choices of argument order by the choice last made. To investigate this potential bias, the frequencies of each type of transition between choices were tabulated. The frequencies of the four possible transitions of choices were found to be almost exactly the same as those expected by chance, given the total number of choices for each argument order. Thus, there was no evidence for an influence of the preceding choice on the current choice.

The numbers of subjects who selected each argument order for each command is shown in Table 2. In 10 cases out of 12, the preferred order was that with the direct object first, the exceptions being the commands Trim and Select. In neither of these

TABLE 2
Study II: Numbers of subjects selecting the direct object first and the direct object second options for each command

Command	Numbers of Ss selecting		Proportion D. Obj. 1st	Binomial prob. (1-tailed)
	D. Obj. 1st	D. Obj. 2nd		
Search	24	10	0·71	0·012
Trim	13	21	0·38	0·115
Replace	19	15	0·56	0·304
Invert	22	12	0·65	0·061
Delete	30	4	0·88	0·001
Save	22	12	0·65	0·061
Select	13	21	0·38	0·115
Link	24	10	0·71	0·012
Convert	26	8	0·76	0·002
Reverse	26	8	0·76	0·002
Erase	31	3	0·91	0·001
Store	25	9	0·74	0·005

cases was the preference for the direct object second statistically reliable, while seven of the 10 direct object first preferences were reliable. Thus, where preferences were shown, they confirmed the hypothesis that the more natural order is that with the direct object of the command verb following immediately after the verb. The size of the preferences varied systematically across the commands however; we would expect the size of these preferences to correlate with any argument order effects which occur in interactive dialogue (Study III). The study provides some empirical evidence for the advantage of direct object ordering despite idiosyncratic choices for several of the commands, and Study III will use the principle of direct object first as an approximation for the most natural order of arguments.

There is a final point of interest in these data. Over all the commands there was a significant tendency (60% of all choices) to favour telegrams with the message identifier as the *second* argument, although the message identifier was as often the indirect object of the command as the direct object. This was true for both command sets (61% and 58%, respectively). Other things being equal, from these results we might therefore expect a slight advantage for commands with the recurring argument in the second position over those with the recurring argument in the first position. This tendency contrasts with the argument orders chosen by the computer professionals in Study I, who favoured orders with the message identifier as the *first* argument.

5. Study III: experimental investigation of argument ordering

5.1. AIMS

The aims of the third study were both methodological and substantive. Methodologically, the main aim was to show that an interactive testing vehicle could be used to investigate hypotheses concerning issues of usability in interactive systems in general.

Another methodological objective was to relate user performance to design practice (Study I) and to natural language predispositions (Study II). The substantive aims were to investigate the effects of three variables on argument ordering: (1) the internal consistency of a set of commands, in which the position of a recurrent argument is held constant, (2) the natural language compatibility, in which the position of the direct object of the command verb is held constant and (3) the display compatibility, in which the entry of arguments matches or fails to match the display format.

5.2. SUBJECTS

A further sample of 48 female subjects, all members of the Applied Psychology Unit's subject panel, participated. Their mean age was 46 years.

5.3. APPARATUS

The experiment was run on a remote terminal (IBM 3275) linked via a direct Post Office line to a time-sharing computer running the VM370/CMS operating system. As the system was capable of providing only the most coarse-grained event timing, a digital tape-recorder was used to log each subject's performance. Each keystroke together with its timing was recorded for subsequent analysis. The same recording device also monitored system response (that is, the time at which each full screen was displayed) via a photocell attached to a dedicated location on the VDU screen.

5.4. PROCEDURE

Subjects were tested individually in a single session lasting about 1 hour. After those subjects with little or no typing experience had been shown how to operate the terminal keyboard, initial instructions were presented on the display. These described the general nature of the task and the specific format of the commands to be entered by the subject.

The task was based on an on-line implementation of the decoding steps described above in section 2.2. Subjects were required to decode a series of 10 messages, each being decoded by entering a sequence of six commands into the system. Appendix 1 shows the sequence of events during the decoding of one of the messages during the training phase of the task. Each command, when correctly entered, performed one of the task steps described above. A command consisted of a name (different subjects used the two different sets of names) followed by two numeric arguments. Table 3 summarizes the two command sets, the argument labels and their order of entry in the various experimental conditions. The argument values varied with each message. Each command required as one of its arguments a message identifier, a two-digit number associated with the current message. The message identifier was automatically incremented after each operation to indicate that a new version of the message had been created. The other argument of the command differed for each operation. Only commands in the correct sequence with the correct arguments were executed and resulted in changes to the message; any other input resulted in an error message. Thus the system did not permit the subject to deviate from the expected command sequence.

Figure 5 shows an example of the terminal display during a decoding sequence in the training phase of the task. The six command labels were displayed in the command option field near the top of the screen, with the command to be used next shown

TABLE 3

Summary of the two command sets and the argument orders used by the "direct object first" and "consistent first" groups in Study III. The orders of arguments used by the "direct object second" and "consistent second" groups were the reverse of these

Set One		Set Two	
	Direct-object argument first		
SEARCH	file no, message id	SELECT	message id, file no
TRIM	message id, segment size	LINK	segment size, message id
REPLACE	message id, code no	CONVERT	message id, code no
INVERT	group size, message id	REVERSE	group size, message id
DELETE	digit, message id	ERASE	digit, message id
SAVE	message id, reference no	STORE	message id, reference no
	Consistent Argument First		
SEARCH	message id, file no	SELECT	message id, file no
TRIM	message id, segment size	LINK	message id, segment size
REPLACE	message id, code no	CONVERT	message id, code no
INVERT	message id, group size	REVERSE	message id, group size
DELETE	message id, digit	ERASE	message id, digit
SAVE	message id, reference no	STORE	message id, reference no

more brightly. The instruction field contained information for the subject concerning command implementation. The amount of information displayed decreased as the subject progressed through the task (see below). The argument field contained values for the arguments *message identifier, code number, file number* and *group size.* The values of the remaining arguments (*segment size, digit* and *reference number*) had to be deduced from the message itself. The order in which the four values in the argument

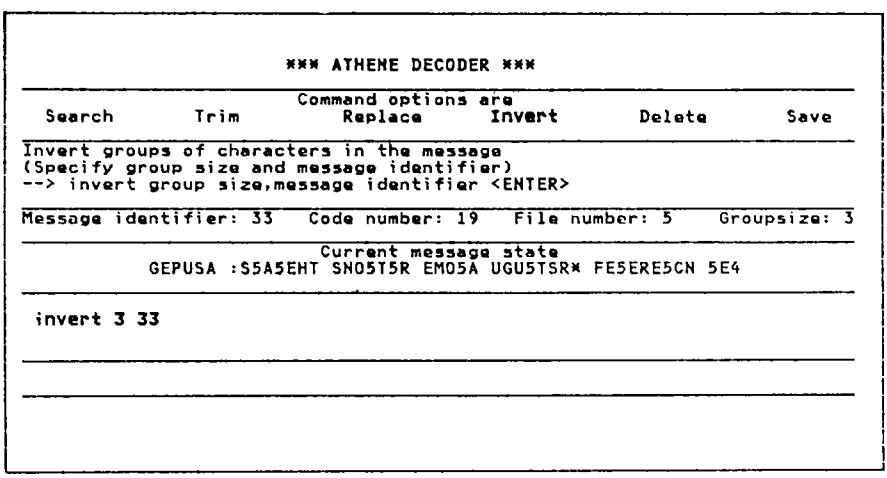


FIG. 5. Example of screen display used during the task. Bold type indicates those characters displayed at heightened brightness. The sections of the display separated by lines are, from the top, (i) Command option field, with the current command at heightened intensity, (ii) Instruction field, (iii) Argument field, (iv) Message field, (v) Input field and (vi) Error message field (blank in this example).

field were displayed varied between subjects. The input field displayed the subject's typed response, and the error field was used to give feedback in cases of erroneous input. If the subject noticed an error before terminating the input she could erase the input line by means of a key labelled ERASE INPUT and re-type the command. Otherwise the input line was terminated by pressing the ENTER key. An erroneous entry was categorized either as a command error or an argument error, and a message displayed in the error field together with the offending section of the command string. Examples of the two classes of error message are, for a command error, "***Wrong Command*** SEARCH" and, for an argument error "***Numbers specified incorrectly*** 17,12". No feedback was given for a correct entry (other than the result of the operation of the command itself).

For the training phase, during the decoding of the first two messages, instructions for the current command were automatically displayed in the instruction field. For each command during the first message this consisted of a description of the action of the command, a description of the command format in terms of the argument labels, and the literal command string to be typed in. Thus the minimal task required of the subject during the first message was to copy the commands given in the instruction field. However she was encouraged to try to understand the selection and use of the arguments and the operation of each command. Queries could be raised during the first training message only.

For the second message, a brief description of the command was given together with the command format in terms of the argument labels; the subject had to find the values of the arguments from the argument field or from the message itself. The instructions in the example shown in Fig. 5 and in Appendix 1 are of this type. No instructions were given during the subsequent eight messages unless specifically requested by the subject (an information request was made by typing in the command name with no arguments). The instructions shown on request were of the same form as those for the second training message.

Following the second training message, further information was shown on the screen telling the subject about the use of the instruction request facility. This information encouraged the subject to guess the correct solution before requesting instructions. The subject then worked her way through the remaining eight messages. As was the case with the training messages, she was always informed of the command name to be used (it was displayed more brightly in the command option field), but no information was given concerning the arguments or their order of insertion. Following the final message all the decoded messages were displayed.

5.5. EXPERIMENTAL DESIGN

There were five main experimental variables. Organization of argument order, the set of commands used and the order across the screen in which argument values were displayed were all varied between groups of subjects. The remaining two variables, messages (two training messages and eight experimental messages) and operations were varied within subjects. Each subject decoded the same ten messages, using the same six operations for each message.

The primary experimental variable was the principle of argument order used by each group of subjects. Sixteen subjects learned to use a system in which the arguments were ordered on the principle of direct object first, 16 used a system with the reverse

argument ordering with direct object second, and 16 used a system in which the recurrent argument (message identifier) was always in the same position. These groups are hereafter referred to as the direct object first group, the direct object second group and the consistent group, respectively. For half of the subjects in the last group, the message identifier was always the first argument, and for the other half of the subjects the message identifier was always the second argument (hereafter consistent first group and consistent second group, respectively). In addition, two alternative sets of the six commands were used; half the subjects in each group used each set. Finally the order across the screen in which the four explicit arguments were displayed (that is, those displayed in the argument field rather than those implicit in the message) was counterbalanced across subjects within each condition using a four-by-four Latin square. The purpose of varying this aspect of the display was to allow a comparison of those commands where the left to right order of the arguments on the screen was the same as the left to right order of the arguments to be typed with those commands where these two orders differed. Thus the example shown in Fig. 5 is an instance of an incompatible relationship between the required order of entry of the arguments and their displayed order. The subjects must enter the arguments in the order "Groupsize" "Message identifier" (i.e. 3 33), while the left-to-right order that these arguments are displayed in the argument field is "Message identifier" "Groupsize".

5.6. RESULTS

5.6.1. Analysis strategy

The time-stamped digital recording of each subject's performance was processed to form a condensed session protocol. Six measures were extracted for each correct command entry for each subject. These were:

(1) *Total time.* The total time to input the correct command from the onset of the display to the depression of the terminator key ENTER.
(2) *Viewing time.* The total time between the onset of the display and the entry by the subject of the first character of the command string. If the subject requested help information, the time spent viewing the instructions was included in this measure, but not the typing time for requesting help nor the machine response time.
(3) *First attempt correct.* The frequency with which the subject entered a command and its arguments correctly at first attempt (scored as 0 or 1 for each operation).
(4) *Information request.* Whether or not the subject requested instructional information (scored as 0 or 1 for each operation).
(5) *Reversed argument errors.* The frequency with which subjects entered the correct arguments for the command but in the wrong order.
(6) *Total argument errors.* The frequency of any class of argument error, given correct entry of the command label. This measure includes not only the data for (5) but also errors resulting from incorrect argument values and from entries with the wrong number of arguments.

These data were submitted to three classes of statistical analysis. First of all, correlational analyses assessed the degree of independence of the six measures. Second, discriminant function analyses were performed to discover which measures best discriminated between the groups, and finally analyses of variance investigated the detailed effects of the experimental manipulations.

5.6.2. Relationships among the dependent measures

The pattern of correlations among the six measures indicated that there were no trade-offs between the measures; slow subjects tended also to be less accurate, to request instructions more often and make more argument errors. One pair of measures, first attempt correct and instruction requests, were strongly related, sharing 84% of their variance in common ($r = -0{\cdot}916$). No other pair of measures shared more than 43% common variance (taking into account the part–whole relationships of viewing time and total time and of reversed and total argument errors), suggesting that the remaining measures tapped distinct aspects of performance.

Discriminant function analyses comparing each group of subjects with each other group in turn showed that no one measure was superior overall in distinguishing between the groups. Total argument errors were the best discriminant for the comparison between the direct object first group and the direct object second group, although the difference between the two groups proved to be unreliable (for this measure, Wilks' Lambda $[1,1,30] = 0{\cdot}93$, equivalent to $F[1,30] = 2{\cdot}33$, ns). The same measure was also the most sensitive index in discriminating between the positionally consistent condition and the positionally inconsistent conditions (that is, the direct object first and the direct object second groups combined; Lambda $[1,1,46] = 0{\cdot}72$, $F[1,46] = 18{\cdot}33$, $p < 0{\cdot}001$). However the "first attempt correct" measure proved to be the best discriminant between the two consistent subgroups (Lambda $[1,1,14] = 0{\cdot}70$, $F[1,14] = 5{\cdot}97$, $p = 0{\cdot}03$).

5.6.3. General effects of the experimental variables

The experimental effects themselves were explored by means of analysis of variance on the data for the eight post-training messages. Although analyses were performed using all six measures, the results for total time and first attempt correct will not be reported here. The analyses for these variables did not differ in any major respect from those for viewing time and instruction request respectively. Analyses for both reversed arguments and total argument errors are included as these measures did give rise to different effects and are both of theoretical interest.

Table 4 summarizes all the significant terms in the various analyses of variance discussed below. Non-significant terms are not included, nor are significant terms in the later analyses which merely replicate effects dealt with by the earlier analyses. Appendix 2 presents the full tables for the major analyses.

(a) *Differences between the groups.* On all measures, performance was best in the consistent group, worse in the direct object first group and worst of all in the direct object second group. Analyses of variance comparing the three groups showed that the overall performance of the three groups differed significantly on all four measures (main effect of groups, see line 1 of Table 4). Detailed comparisons between the groups are made in subsequent analyses. Overall, subjects tended to improve with practice, although the number of reversed argument errors did not significantly decrease (main effect of messages, see line 2 of Table 4). Figure 6 plots the learning curves for the four measures for each group. The results for the two consistent order subgroups (consistent first and consistent second) are plotted separately.

TABLE 4

Study III: Summary of F-ratios obtained in analyses of variance (see text for details of analyses)

Source	*df*	View time		Info req.		Total arg.		Rev. arg.	
		F	*p*	*F*	*p*	*F*	*p*	*F*	*p*
				Analyses comparing all three Groups					
1. Groups (G)	2,42	5·46	<0·01	3·77	<0·05	3·96	<0·05	5·50	<0·01
2. Messages (M)	7,294	123·7	<0·01	41·47	<0·01	10·48	<0·01	0·89	ns
3. G by M	14,294	2·74	<0·01	0·76	ns	1·29	ns	0·90	ns
4. Command (C)	5,210	15·09	<0·01	19·31	<0·01	9·96	<0·01	2·48	<0·05
5. M by C	35,1470	29·92	<0·01	1·84	<0·01	2·47	<0·01	1·50	<0·05
6. G by M by C	70,1470	1·05	ns	0·94	ns	1·40	<0·05	1·53	<0·01
7. C by C. Set	5,210	3·03	<0·025	1·37	ns	1·34	ns	0·70	ns
8. G by M by C.S	14,294	1·79	<0·05	1·55	ns	2·15	<0·025	1·26	ns
				Analyses restricted to Groups D.O.1st and D.O.2nd					
9. G by C	5,140	1·01	ns	1·25	ns	1·11	ns	2·56	<0·05
10. G by M by C	35,980	0·74	ns	1·03	ns	1·12	ns	1·70	<0·01
				Analyses restricted to Groups Con 1st and Con. 2nd					
11. Group	1,12	0·88	ns	5·71	<0·05	0·94	ns	—	—
				Analyses comparing Mean Consistent and Mean Inconsistent Groups					
12. Group	1,42	8·62	<0·01	7·53	<0·01	5·21	<0·05	—	—
				Analyses comparing Consistent First and Mean Inconsistent Groups					
13. Group	1,38	7·56	<0·01	10·31	<0·01	4·88	<0·05	—	—
				Analyses comparing Consistent Second and Mean Inconsistent Groups					
14. Group	1,38	2·96	ns	1·06	ns	1·51	ns	—	—
				Analyses on the effects of Display Compatibility					
15. Disp. Compat.	1,45	4·02	0·054	1·85	ns	4·40	<0·05	3·72	0·058

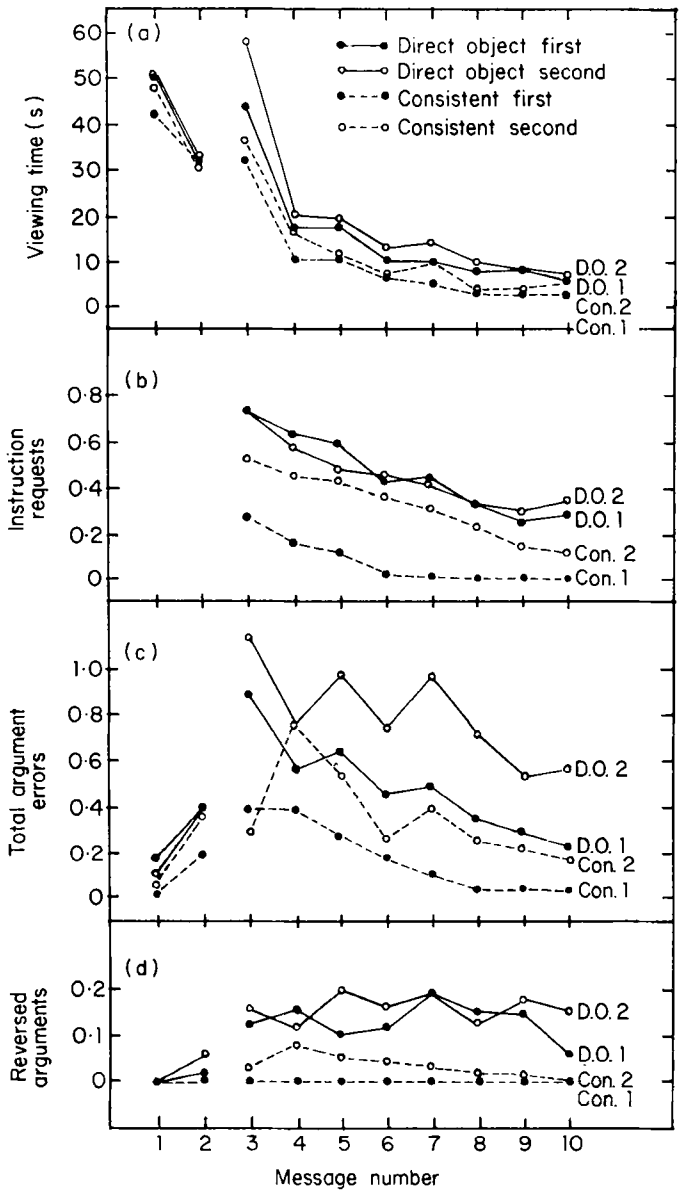


FIG. 6. Study III: Performance of each argument order group across the ten messages. (a) mean viewing time in seconds per command; (b) mean proportion of commands for which instructions were requested; (c) mean number of argument errors per command; (d) mean number of argument reversals per command. On messages one and two training information was displayed.

Although the absolute number of reversed argument errors did not reduce with practice, there was clear evidence that subjects were, in fact, learning about argument order. In the initial stages, subjects frequently requested instructions. This would have tended to reduce the number of opportunities for making reversed argument errors, since the instructions included argument order information. Taking this into account,

the corrected error rate decreased significantly from 25% on messages 3 and 4 to 13% on messages 7 and 8 ($F[7,315] = 5{\cdot}69$, $p < 0{\cdot}01$).

Finally, the learning rate differed significantly between the groups, but on the time measure only (message by group interaction, line 3 of Table 4); inspection of Fig. 6(a) suggests the differential effect is due to the long viewing times on message 3 for the direct object first and particularly the direct object second groups compared to that for the consistent groups.

(b) *Differences between the commands.* The effects discussed so far have been averaged over the six command operations. It is therefore important to determine whether the individual command operations contributed differentially to the main experimental effects. The analyses showed that the six operations did differ in their

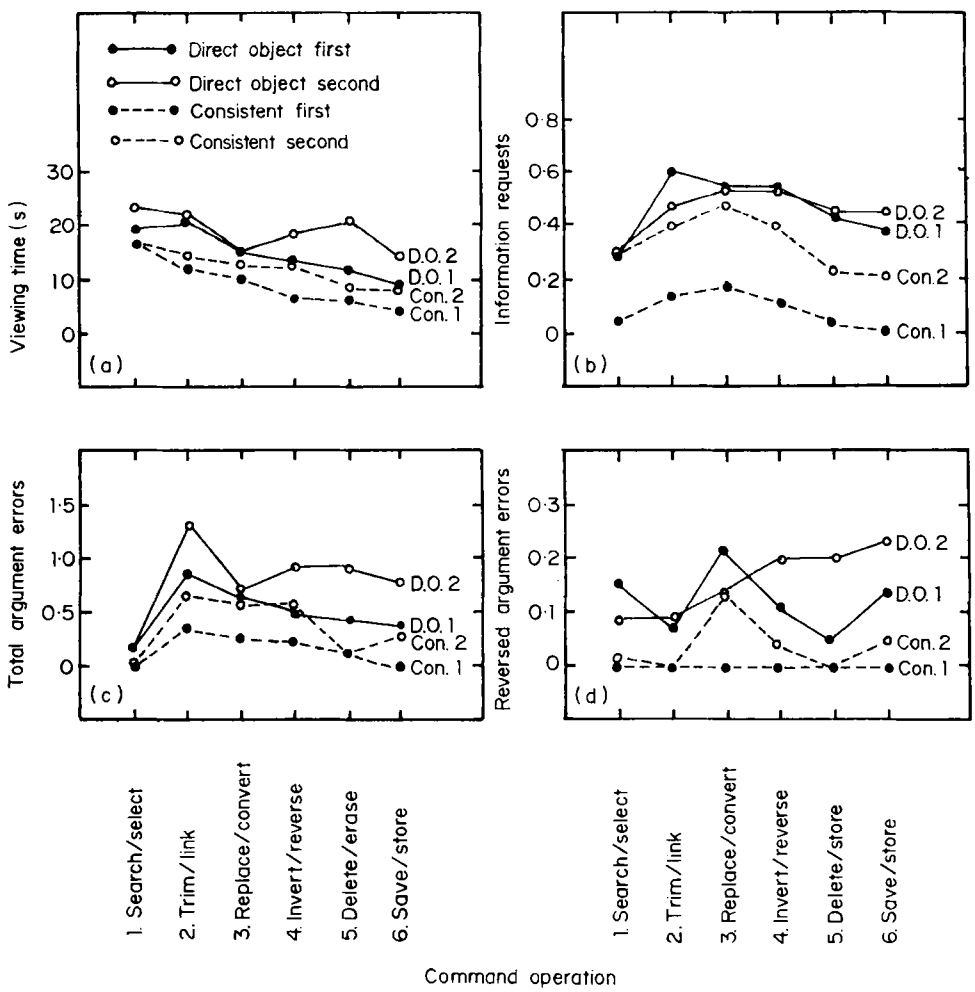


FIG. 7. Study III: Performance of each argument order group across the six command operations. Scores are averaged over messages 3–10. (a) Mean viewing time in seconds per command; (b) mean proportion of commands for which instructions were requested; (c) mean number of argument errors per command; (d) mean number of argument reversals per command.

ease of use on all measures (line 4 of Table 4). Relevant data are plotted in Fig. 7. The six command operations were also learned at different rates (command operation by message interaction, line 5 of Table 4), those operations performed least well initially improving the most quickly.

The effect of the operations did not differ significantly across the three experimental groups. This indicates that the main effects were, for the most part, independent of the individual command operations. However, there was some evidence for a differential learning effect of the six command operations across the groups in the case of reversed and total argument errors (three-way interaction of group by command operation by message, line 6 of Table 4). Inspection of the data suggested that the effect was due to differences between the direct object first and direct object second groups; the effect is considered in more detail in the section dealing with comparisons between these two groups.

The data were also examined to determine whether the different labels assigned to each command operation had any effects on performance over and above that of the operations themselves. The only effect found was relatively small. Since half the subjects used one set of names and half used another, an interaction between the command set used and the command operation would reflect this effect. The interaction was significant in the case of viewing time only (line 7 of Table 4). The main discrepancy between the two command sets was with operation 5: subjects using the *delete* label spent on average 16·3 s viewing the screen, compared with the 10·4 s spent by subjects using *erase*, a difference of 5·9 s. This contrasts with an average discrepancy of only 2·3 s for the other five operations.

Only one other source contributed significantly to the variance: the three-way group by command set by message interaction was significant both for viewing time and total argument errors (line 8 of Table 4), suggesting differential learning effects for the groups depending on the command set used. However no consistent trends were discernible in the data, and such an effect is of little theoretical importance.

5.6.4. Effects of position of the direct object

The analyses comparing the three groups of subjects demonstrated that their performances differed. More detailed analyses are therefore required to assess the statistical significance of particular comparisons. First, is there any evidence that the position of the direct object (first or second) has any distinguishable effects on performance? Analyses comparing these two groups showed that there was no *overall* difference on any measure, indicating that the organizing principle of placing the direct object first was not an aid to performance. However, the results do support the hypothesis that the "natural" ordering of arguments varied from command to command, as the difference between the direct object first and direct object second conditions varied across the six command operations. This effect was significant in the case of the reversed argument measure only (group by command operation interaction, line 9 of Table 4). This measure is the one of greatest relevance to differences between the direct object first and direct object second groups since the expected error caused by an incompatible argument order is entry of the arguments in the wrong order. Figure 7(d) plots the numbers of reversed argument errors made by each group for each command. Some operations, such as save/store, reverse/invert and delete/erase, caused more argument reversals in the direct object second group, whereas others,

such as search/select and replace/convert caused more reversals in the direct object first group. The relationship between the number of reversed argument errors and the preference result collected in Study II is presented below in section 5.6.8.

These differences between the command operations also changed over the eight messages (three-way interaction between group, command operation and message, line 10 of Table 4). However the effects are small, and no interpretable trends are apparent. No other sources in the analyses concerning the position of the direct object reached significance.

5.6.5 Effects of position of the recurrent argument

The two positionally-inconsistent systems produced comparable levels of performance. The two consistent systems, on the other hand, showed rather different patterns. On all measures, the consistent first condition gave rise to better performance than the consistent second condition. This difference proved to be reliable in the case of instruction requests (line 11 of Table 4). Consistent first subjects requested instructions at a rate of 0·08 per command compared with 0·33 for consistent second subjects. The advantage of the consistent first condition was also reliable in the case of the first attempt correct measure ($F[1,12] = 6{\cdot}11$, $p < 0{\cdot}05$). The effect did not reach statistical significance on the viewing time measure or on total argument errors, possibly due to the small numbers of subjects in each subgroup (8) and the high variability between subjects. In addition, the scarcity of reversed argument errors in the consistent conditions precluded statistical analysis.

5.6.6. Comparisons between consistent and inconsistent conditions

Since performance with the two positionally consistent conditions differed, further analyses were performed to compare these conditions with the positionally inconsistent ones. These contrasts are independent of those made so far. First, analyses showed that performance was significantly better in the consistent conditions on all measures. However, when the two consistent subgroups were taken separately, it was evident that the superiority was primarily due to an advantage of the consistent first subgroup over the inconsistent groups. On all measures this subgroup showed reliably better performance than the combined inconsistent groups. In contrast, the consistent second subgroup performed no better than the inconsistent groups on three of the four measures. Details of the analyses can be seen in lines 12, 13 and 14 of Table 4. For viewing time, instruction requests and total argument errors, the statistical comparisons were made by means of orthogonal contrasts based on the earlier analyses. The small numbers of reversed argument errors in the consistent group precluded the use of a parametric test. However, using the Fisher exact probability test to compare the numbers of subjects who made at least one reversed argument error, the consistent first subgroup (0 out of 8 subjects) and the consistent second subgroup (2 out of 8 subjects) were both significantly more accurate than the combined inconsistent group (26 out of 32 subjects; two-tailed $p < 0{\cdot}001$ for consistent first and $p = 0{\cdot}010$ for consistent second).

5.6.7. Effects of display compatibility

A further issue concerns not differences between the main experimental groups, but the effects of display compatibility on the accuracy of performance. The disadvantage

of an incompatible display format was substantial, with the average decrement in performance varying between 11% and 55% on the four measures. For three of the six command operations, the values of both the arguments had to be chosen from the argument field (for the other three commands one of the argument values had to be derived from the message itself). For these three operations (search/select, replace/convert and invert/reverse), therefore, the order in which the two arguments were required to be entered in the command string could be either compatible or incompatible with their left-to-right order on the screen. The experimental design was such that for different subjects different operations were either compatible or incompatible with display order, and therefore effects of display compatibility were confounded with differences between the commands. However across the subjects in each main group or subgroup, there were an equal number of instances of compatible and incompatible display orders for each of the three command operations. Table 5

TABLE 5

Study III: Summary of effects of display compatibility. Compatible display order (C) includes all those cases where the left-to-right order of the arguments in the argument field was the same as the left-to-right order of the arguments in the command string to be typed. Incompatible display order (I) included all cases where the two orders differed. The measures are averaged over the eight messages and the three command operations for which display compatibility could be extracted. Viewing time is in seconds per operation; instruction requests, reversed arguments and total argument errors are the mean number of instances per operation

	Argument order condition							
	Dir. Obj. 1st		Dir. Obj. 2nd		Consistent		Mean	
Measure	C	I	C	I	C	I	C	I
Viewing time	15·9	17·2	17·9	20·5	12·4	13·8	15·4	17·2
Instruction request	0·45	0·49	0·43	0·50	0·19	0·30	0·36	0·43
Reversed arguments	0·14	0·21	0·13	0·15	0·0	0·07	0·09	0·14
Total arg. errors	0·39	0·50	0·53	0·70	0·23	0·36	0·38	0·52

presents the mean data for compatible and incompatible orders for each of the three main groups on the four measures. In every case those command strings compatible with display order were performed better than those incompatible with the display order, although the effect was small in some instances. The data were subjected to an analysis of variance for each measure, with Groups (direct object first, direct object second and consistent) extracted as a between subject variable and display compatibility extracted as a within subject variable. The partial confounding of compatibility with command operation within each subject would have tended to inflate the error terms; even so, the effect of display compatibility was significant or marginally so on three of the four measures (line 15 of Table 4). There was no evidence that the compatibility effect differed between the three groups (F-values for group by compatibility interaction were less than 1 in every case).

5.6.8. Relationship with preference results of Study II

The extent to which users made reversed argument errors in the positionally inconsistent conditions varied across the individual command verbs. This variation could at least in part be due to a natural language influence of the users' mental representations of the individual command verbs. To assess this potential influence, the preferences for argument orders measured in Study II were correlated with the frequency of reversed argument errors for each command made by subjects in the inconsistent conditions. If preferences of the type shown in Study II were at work when subjects were selecting argument order, then we would expect that the stronger the preference for a particular order, the less argument reversal errors should be made when that order is the required order and the more when that order is the reverse of the required order. Thus, for each of the 12 command verbs, there are two measures which we can correlate to test this relationship. The first is the "preference score" from Study II in terms of the proportion of subjects who preferred the order with the direct object as the first argument. The second, "argument reversals", is the proportion of reversed argument errors that were reversals of direct object second into direct object first in Study III. These two measures would be predicted to correlate positively: the stronger the preference for direct object first, the more the number of argument reversals resulting in the direct object being entered first, and vice versa. The correlation between these two measures was statistically reliable in the expected direction ($r = 0{\cdot}532$, $df = 10$, $p < 0{\cdot}05$, one-tailed test).

5.7. DISCUSSION

This section will confine itself to consideration of the results and implications of Study III; the general discussion (section 6) will deal with the broader issues arising from all three studies.

5.7.1. Methodological issues

Two classes of implication follow from the study. The first class concerns the sensitivity of the experimental tool in identifying reliable and behaviourally valid effects, and the second concerns the substantive issues arising from the findings themselves. The task environment of message decyphering and the interactive test vehicle together provided an experimental context in which all the major variables examined had an effect. In one way or another, they each influenced the ease of using the interactive system. The effects obtained were not only significant in a statistical sense, they were also of a significant magnitude from a behavioural point of view. For example, even after decoding nine messages, subjects run under the positionally inconsistent conditions were still making argument errors on 40% of their command entries and requesting instructions for 30% of the operations, while by the eighth message subjects in the consistent first group were making only 4% argument errors and not requiring instructions at all. By the fifth message, subjects in the latter group were committing fewer argument errors than the number made by the inconsistent groups on message 10.

In addition to the test vehicle providing a number of reliable and sizeable performance effects, the various measures of performance showed a substantial degree of internal consistency. Measures of speed, errors and use of instructions inter-related in a coherent fashion. Some general performance effects, such as improvement with

practice or the advantage of positionally consistent over inconsistent conditions, were reflected across all the measures. In other cases, the measures played a distinctive role in highlighting more specific effects, such as the sensitivity of the reversed argument measure in distinguishing between direct-object first and direct-object second conditions. Relationships amongst the measures also illustrated the logical relationship between aspects of performance, such as the masking of reversed argument errors by other classes of errors early in performance. In these respects both task environment and test vehicle appear to have sensitivity to behavioural effects. However, unless these bear on the real world, such sensitivity is of no avail. Whilst we cannot formally demonstrate the face validity of these findings, we can point to an observational study of a real system where users produced similar errors to those investigated in the current study (Hammond *et al.*, 1980).

5.7.2. The advantage of positional consistency

The most important class of implications follow from the findings themselves. Not surprisingly, performance improved with increasing experience of the task, the different command operations varied in their ease of use and, consistent with learning in like performance tasks, the differences between the operations tended to diminish with practice. Of greater interest are the differences between the experimental groups using systems organized on different principles of argument ordering. The three main alternatives for argument formatting gave rise to large performance differences. Systems in which the recurrent argument was placed consistently in the same position were operated more accurately, with less recourse to instructions and with less time spent viewing the screen before typing a command than those systems in which the recurrent argument was not invariably in the same position. The advantage of consistency did not accrue gradually, as might be supposed if subjects required time to induce the ordering rule. Rather the rule was induced almost immediately *despite* the fact that users were not explicitly told of the rule. The advantage of consistency was as strong, or stronger in the case of viewing time, for the first post-training message as later in the session.

Memory limitations might plausibly account for the advantage of positionally consistent orderings: subjects merely have to remember a rule rather than the order for each operation separately (this hypothesis additionally assumes that the subjects in the positionally inconsistent groups did *not* induce a general rule based on the position of the direct object of the command verb or on some alternative formulation). However the memory hypothesis alone cannot deal with the second major finding concerning positional consistency. Those subjects for whom the recurrent argument occurred in the first position performed better on all measures than those with the recurrent argument placed second, although this effect was reliable only in the case of the instruction request and the first attempt correct measures. On all measures the performance of the consistent first group was superior to that of the inconsistent groups, while consistent second subjects were superior only in the case of reversed argument errors. The sizes of these differences are substantial: on average consistent second subjects had recourse to instructions four times as often and made argument errors twice as often as consistent first subjects. A memory hypothesis would make no distinction between the two consistent conditions since there is no *a priori* reason for supposing that a rule for positional formatting would be any more difficult to

remember when the recurrent argument is in the second position than when it is in the first. It is likely that some additional influence is at work which favours the consistent first condition.

The data suggest that the factor discriminating between the consistent first and consistent second conditions is of a content-free nature. In no case did the position of the recurrent argument interact significantly with the command operation: the difference between conditions was comparable for all operations (see Fig. 7). The only hint of an exception was with reversed argument errors in the case of the REPLACE/CONVERT operation. Neither was there any evidence of a bias towards the direct–indirect object order which might have modulated the difference between the two positionally consistent conditions (see below). These aspects of the data suggest that the discriminating factor is not related to the meaning of individual commands.

Similarly, there was no clear evidence that the process of abstracting the positional rule was itself the critical factor. Where the recurrent argument was in a constant position there were minimal occurrences of reversed argument errors. Errors in entering the recurrent argument itself were also infrequent when compared with errors on the variable argument. In the initial stages of learning, it is possible that the opportunities for both kinds of error were offset by users' frequent recourse to the information facility. However, it would not seem unreasonable to suppose that use of the facility was motivated by those difficulties which were also evidenced as errors: that is, difficulties with argument values rather than argument order. Nevertheless, more detailed data on the purpose of individual information requests would be needed in order to establish whether subjects were using the facility to abstract the ordering rule. This could readily be achieved in future work by splitting the information facility into separately invokable components.

One factor which could provide a plausible account of the difference between the first and second positions concerns the status of the information in the structure of the command string and the way in which information structure is conventionally expressed and interpreted in the course of understanding natural communications. First, the information expressed by the two arguments following each command obviously differs. The recurrent argument involved a repeated identification of the message whereas the variable argument concerned an attribute particular to individual command operations. Even though the numeric value of the message identifier was incremented from operation to operation, and as such was not entirely redundant, its very recurrence makes the concept of "the message" central in the context of the experimental task. In addition, in context, the variable arguments would also tend to refer indirectly to the message. For example, segments, groups and elements were parts of the message. Even a file can be construed as the location of "the message". As such the message identifier would most probably be taken for granted as representing redundant or contextual information. In contrast, the variable argument would be salient in the sense that it expressed information which is not redundant and is of particular significance to individual command operations. In these respects the experimental task would impose differential status on the information conveyed by the arguments in a way that would be equivalent irrespective of their order of entry.

Given this differential status as a function of the task, the structure of the command string itself may exert a biasing influence. Linguistic analyses of the way in which information structure is expressed in sentences (e.g. Halliday, 1967, 1968) suggest

that information which is taken for granted, or presupposed in a context, normally occurs in sentence structures *before* information which represents some novel contribution to the established context. In this respect, the structure "command ⟨message identifier⟩ ⟨variable argument⟩" is compatible with the way in which information is typically expressed in sentences. The compatibility arises since the contextually overdetermined message identifier comes before the specific information contributed by the variable argument. Furthermore, psychological studies of human understanding suggest a plausible underlying mechanism for a compatibility effect of this type. Studies show that the structure of a sentence biases people's understanding of the information being expressed. People are more likely to overlook inaccuracies in information which is structurally taken for granted than they are with information which is structurally marked as a novel contribution to the context (Hornby, 1972, 1974). The implication is that people actually process constituents marked as taken for granted less efficiently than constituents marked in sentence structure as an informative contribution.

In terms of this analysis it is not difficult to see how the advantage of having the recurrent argument in the first position could arise. The relative positions of the message identifier and the variable argument are compatible with the communicative convention in natural language of expressing information which is taken for granted before novel information. In addition, the tendency for people to process the novel information more thoroughly than that which is taken for granted would have the consequence that relatively more attention would be paid to the variable argument than to the more redundant message identifier. In contrast, when the *variable* argument occurs in the first position, rather less attention would be paid to its interpretation and rather more to the redundant message identifier. In consequence, the variable argument would be more difficult to learn and remember. Since the message identifier occurred with each command, it would be contextually overdetermined and its position would have fewer consequences.

This type of explanation is broadly consistent with the two significant features of the data mentioned earlier; the content-free nature of the effect and its invariance with experience. The differential status of the information conveyed by the two arguments would be equivalent for all the command operations, and therefore the advantage of having the recurrent argument in the first position would be comparable for all command operations. Second, any tendency to process the first argument less efficiently than the second is presumably a relatively stable attribute of human cognition and unlikely to change within the time course of the experimental task.

In sum, this explanation would attribute the relatively stable and uniform advantage of having the recurrent argument in the first position to a bias in the user's evaluation of the arguments. The structure of the command string, as a consequence of its compatibility with conventional means of expressing information, would modulate the differential status of the arguments attributable to the task itself. Obviously, at this stage, an explanation of this type must remain tentative. However, the explanation pinpoints a factor which may prove of some importance for conceptualizing human–machine dialogue. It is also a factor which is readily amenable to further empirical evaluation. For example, the explanation argues that the difference between the first and second positions for the recurrent argument is primarily one of ease of use rather than the abstraction of the ordering principle *per se*. The relationship between ease of use and abstraction could be tested quite simply by informing the subjects in

advance of the actual ordering rule. If the ease of use explanation proposed above was indeed the critical factor, then the differences in performance between the two positions of the recurrent argument should still occur.

5.7.3. Effects due to the position of the direct object

The pattern of the differences between the two positionally inconsistent groups contrasted strongly with those between the two consistent subgroups. The two consistent conditions differed on their average levels of performance, while there was no overall difference in the case of the two inconsistent groups. In addition, for the consistent conditions these differences remained invariant over all commands and messages, while differences between the direct object first and direct object second conditions were specific to particular commands, and these differences changed across messages. Finally, while subjects in the consistent groups appeared to use abstracted principles to help them remember argument ordering, there was no evidence for the abstraction of a principle in the direct object first or direct object second conditions. Subjects appeared to be remembering the argument order for each operation independently of the others. Evidence for these contrasts is now considered.

For the task used in the study, subjects inserting arguments in the "natural" order of direct object first performed on average no better than the subjects using the "unnatural" ordering of direct object second; there was no evidence that performance was generally improved by a match between the supposed natural language order and the order of entry of the arguments. Several lines of evidence also suggest that subjects were not abstracting the common position of the direct object and using this abstraction as a general rule. The preference results from Study II showed that for several of the commands subjects agreed neither among themselves nor with our suppositions as to the most natural order of arguments, suggesting that subjects may not have had a general notion of direct object which could be called upon across all the commands. If nevertheless subjects had been using such a rule with success, the numbers of errors reflecting argument order uncertainty, namely reversed argument errors, would have been as small as in the positionally consistent group. The rate of reversals was about 15 times as great in the inconsistent groups, and only 19% of these subjects made no reversal errors at all, compared with 88% in the consistent group.

Although the "natural" order of direct object first appears to give no systematic advantage over the less "natural" indirect object first ordering, the present experiment sheds no light on the issue of whether the users could have capitalized on the direct object principle had they been explicitly told it. Unfortunately, Study II is of relatively little help, since it measured preferences rather than how well subjects could have judged which argument came closest to the role of direct object. If such judgements could be made accurately, then an explicit direct object principle might prove as effective as the positionally consistent system with the recurrent argument first. On the other hand, if subjects were unable to judge which argument is the direct object, then such a principle would not be effective even if explained to subjects. Under these circumstances, the effectiveness of a direct object principle might be expected to depend on the particular command verbs.†

In the present study, the command verbs did indeed have specific effects, even though subjects were not told of the ordering principles and apparently were unable

† We thank Tom Moran for drawing this point to our attention.

to capitalize on the direct object position as a general organizing principle. For some operations, more argument reversals were made by subjects in the direct object second group while for other operations more reversals were made by subjects in the direct object first condition. In addition, this pattern tended to change across the eight post-training messages, the advantage of some orders increasing with practice. It appears that when the argument order has no easily abstracted principle, the relative advantage or disadvantage of a particular argument order will depend on specific characteristics of the command, its operation, its argument labels and perhaps other aspects of the task environment. To the extent that the order preferences discovered in Study II predicted the performance advantages of those orders in the interactive task, it is possible to conclude that subjects are making use of the common semantic features of the two tasks, that is the command names, the command operations and the argument labels. Although the relationship between the two tasks is reliable, the preference task only accounts for 28% of the total between command ordering variance, suggesting that other factors are playing a substantial role in determining choice of argument orders in the interactive task. One of these factors is the order in which the explicit argument values are displayed on the screen; this is discussed in more detail below.

A further issue concerning the role of natural language is its influence on the performance of the positionally consistent groups. We have argued that ordering effects in the two consistent subgroups are content-free and independent of the particular command operation. We would therefore expect these effects to be independent of whether the first argument happened to be the direct object or the indirect object of the command verb. Likewise, we would expect no performance effects due to relationships between the actual order of argument entry and the preferred argument order for that command measured in Study II. The results indeed provided no evidence for either of these relationships. Taking those commands for which the direct object was first and those for which the direct object was second, the mean scores for each measure, averaged over the two positionally consistent conditions, were 10·2 s versus 11·2 s in the case of viewing time, rates of 0·20 versus 0·22 in the case of instruction requests and rates of 0·29 versus 0·30 in the case of total argument errors. Thus there was no reliable advantage of having the direct object first. To evaluate the effects of order preference on performance in the positionally consistent conditions, the proportion of subjects in Study II preferring the argument order of message identifier first for each command was correlated with the advantage in the consistent conditions of having the message identifier as the first argument over that of having it as the second argument. This score was calculated for each command on each measure. The correlations between the preference scores and the performance advantage of message identifier first were small and unreliable for all three measures (−0·35 for viewing time, +0·07 for instruction requests and +0·32 for total argument errors). Thus, when users can select argument order on the basis of the consistent positioning of a recurrent argument, there is no evidence that either natural language order or preferred argument order have any systematic influence on performance.

5.7.4. The advantage of display compatibility

The final factor manipulated in the study was the relationship between the order in which the explicit argument values were displayed left to right across the screen and

the order in which the same argument values has to be entered in the command string. The design of the study was such that the effect of this factor could only be studied for three of the six command operations, and to obtain means for each subject the data had to be averaged across the three operations. Therefore only general measures of the effect of display match or mismatch could be extracted. Compatibility between the display and the argument entry order proved to be advantageous on all measures, though the difference was not reliable in the case of instruction requests and only marginally significant in the case of the other three measures. The sizes of the advantages were however of considerable behavioural significance: 12% for viewing time, 19% for instruction requests, 56% for reversed argument errors and 37% for total argument errors.

An incompatible display could have both a specific and a general influence on performance. The specific influence concerns resolving uncertainty of argument position. If the user is uncertain of the required order of entry, then an order may be adopted which reflects a left-to-right reading strategy. This would give rise specifically to an increase in argument reversals with display incompatibility. This increase was found to occur for the positionally inconsistent conditions, where there was most order uncertainty. With positionally consistent systems, where there were very few argument reversals (3%), there would be much less order uncertainty. Nevertheless, all the reversed argument errors in the positionally consistent systems did occur under conditions of display incompatibility.

A more general influence of display incompatibility may arise even when the user is certain of the order of arguments. In searching for a particular value reading from left to right, the value for the second argument would be encountered before the value for the first argument. Any mental re-ordering of these values before their actual entry would tend to slow performance and increase the likelihood of making a mistake. Indeed, both viewing time and total argument errors tended to increase for the positionally consistent and the positionally inconsistent systems under conditions of display incompatibility.

6. General discussion

The three studies in the present series have examined several issues associated with a relatively simple component of human–computer dialogue: the structuring of arguments within a set of commands. Software specialists tended to prefer positionally consistent systems in which a recurrent argument was most frequently made the first argument in the command string. There was, however, considerable divergence in their justifications for particular design decisions. Users, in a different task involving the same information, showed marked preferences for structures analogous to natural language, with the direct object of the command verb emerging most frequently as the first argument. The users also showed a small but reliable tendency to prefer expressions with the recurrent argument as the second argument. While the third study showed that both factors were important in interactive dialogue, it also underlined the importance of considering interrelationships within the total system environment. Positionally consistent systems were indeed the most readily learned, but this was primarily attributable to the condition where the recurrent argument was placed

first. In addition, direct–indirect object effects were only detected with the positionally inconsistent systems.

This divergent pattern of data reinforces the kind of methodological approach in which emphasis is placed on the integration of evidence from different techniques and over a range of variables. Furthermore, adequate interpretation of the specific effects and their interrelationships clearly requires a broader conceptual view of users' knowledge and the cognitive processes recruited in the course of human–system dialogue. The requirement for a broader conceptual view and the development of working hypotheses is not simply of theoretical interest. Each of the three studies pinpoint different aspects of the requirement.

At a practical level it could simply be pointed out on the basis of the first study that half the sample of software specialists arrived at a command structure which gave rise to suboptimal user–system performance while the other half of the sample arrived at a solution which gave the best user performance; that is, a positionally consistent command structure with the recurrent argument in the first position. As such, the performance study, Study III, could be taken as a validatory procedure for design decisions or for distinguishing designers with good user-oriented intuitions from those with less than adequate intuitions. However, a validatory procedure glosses over the problem of the information which designers are using to guide their decisions.

While the software specialists did attempt to take into account both system and user variables, the appeals to "naturalness" and consistency were formulated in an often vague and imprecise form making little or no allowance for multiple determinants of usability. Obviously, some of the system-oriented justifications such as the direction of data travel are likely to be opaque to the occasional user. More importantly, in the absence of an integrated conceptualization of the *user* as a processor of information, there is no valid basis for selecting one of the user-oriented justifications in preference to another. The observed lack of consensus between software specialists, the imprecision of their justifications and the very real performance consequences of the decisions themselves presents strong practical motivation for providing a more precise conceptual analysis to enable the designer to interpret and structure user requirements in human–system dialogue. Nevertheless, it is not clear that any other group of professionals would have done a better job.

Similarly, the methodological case for the same kind of conceptual analysis is reinforced by the study of user preferences for the order of arguments following command verbs. Although a relationship was found between the strength of the user preferences and the errors made in interactive dialogue, two important qualifications apply. First, the relationship only occurred with the positionally inconsistent systems. Second, the relationship accounted for only 28% of the variance between the command verbs in positionally inconsistent systems. As such it is best viewed as a context-dependent, secondary effect which requires interpretation within a broader conception of the dialogue. The context-dependent aspect is reflected in the fact that users' preferences in telegram ordering could be described by an implicit principle of the direct object occurring as the first argument of the command verb. This same principle did not seem to be abstracted by users as an heuristic for learning the structure of interactive dialogue. It is likely that this is a matter of possibility rather than choice, for people are typically unaware of underlying linguistic principles which govern the structure and content of their everyday use of language. Furthermore, liguistic prin-

ciples, taken in isolation from a psychological theory of language performance, often prove inadequate as a basis for analysing the ease of human understanding (for example, see Greene, 1972; Johnson–Laird, 1977).

The practical consequences of this for the methodology of system design are quite clear. Simply consulting the preferences, opinions or intuitions of a sample of users concerning design options could easily result in misleading implications concerning likely usability. Consultation of users' opinions may be important for other reasons such as identifying points of subjective concern or for validating assumptions concerning underlying variables (as was the case with Study II). However, users' subjective intuitions do not necessarily predict their objective performance in a simple one-to-one manner, nor are they an adequate substitute for an understanding of the conceptual issues. This requires interpretation of evidence obtained in live dialogue such as that provided by the third study.

At a practical level the evidence obtained in the interactive version of the task demonstrates the multiple determination of usability. Positional consistency, natural language expressions, and the compatibility of information on a display with its order of entry in the command structure each exerted some kind of influence on user performance. However, the effects of the different variables were neither equivalent in extent nor in their detailed characteristics. This means that the relative contributions of the different variables, the characteristics of the contributions and the contexts in which they occur must be carefully assessed prior to drawing any firm inferences concerning exact design features to incorporate in a real application for an interactive system. Indeed, we would not wish to claim that the effects obtained in the third study would necessarily occur with every interactive system in which similar argument ordering variables were at issue.

It will have been noted that the particular task used in the experiment incorporated a number of features uncharacteristic of real systems. For example, the command syntax was simple, the decoding sequence was predetermined and users were not required to make any decisions relating to higher level problem solving. In addition, the users did not have to select commands or to remember the specific command verbs. In effect it would have been possible for the users to complete the decoding task by learning and remembering the sequence of argument pairs. Although it is implausible to suppose that users completely ignored the meanings and consequences of the commands they were issuing, it is possible that their understanding of the system was incomplete. In consequence, the issue of generality would need to be resolved by further experimentation before design decisions could be based on this type of evidence. As we pointed out in the Introduction, design decisions in environments as complex as these can only be based on an appreciation of the *whole* system. What Study III highlights is that certain factors can affect performance under the appropriate conditions.

On the basis of this evidence we can start to characterize the ways in which the kinds of knowledge outlined in the Block Interaction Model of Fig. 2 (see section 1.3) interrelate with human cognitive processing to give rise to multiple determination of usability. In Figs 8(a) and 8(b) the components of the original Block Interaction Model are expanded. They are expanded both to specify in more detail the features of the experimental task and tc include domains of cognitive activities. Thus, "representation of the problem" in Fig. 2 is expanded to include the three task steps of

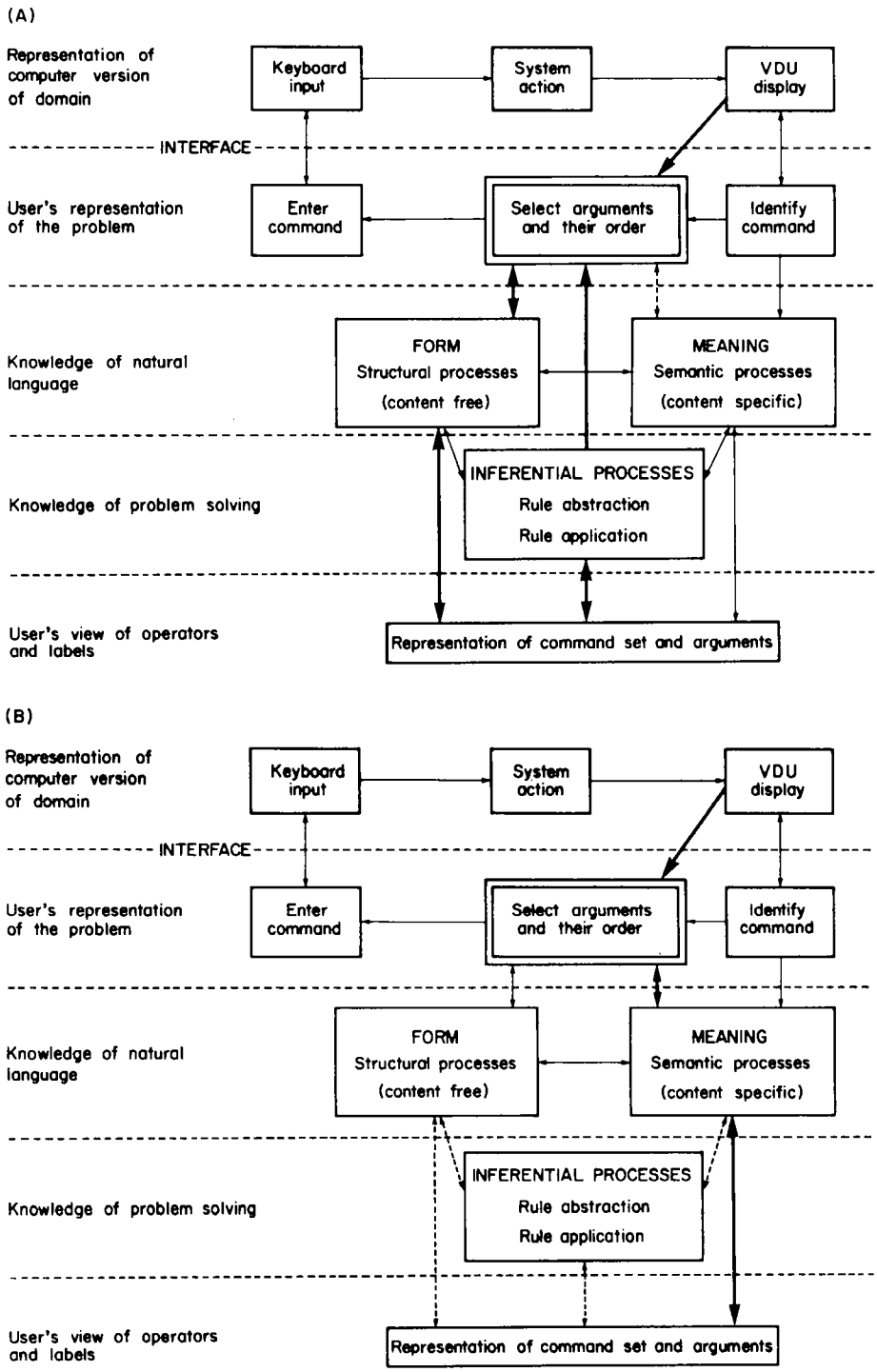


FIG. 8. Restructuring of working hypotheses. (A) Positionally consistent conditions; (B) positionally inconsistent conditions.

identifying the command to be used, the selection of its arguments and their order, and actual entry of the command. Similarly, for this simple task we can specify the user's representation of the computer version of the decoding device in terms of keyboard input, some system action consequential on issuing the individual commands (including error messages) and the terminal display resulting from that action. The broken line between these various components represents the interface between the users' representation of the task and their representation of the system.

The original Block Interaction Model is also expanded in relation to the cognitive processes recruited in human–computer dialogue. For the present purposes we can partition "knowledge of natural language" into cognitive activities concerned with the processing of form and those concerned with the processing of meaning. Only those activities concerned with inference need be considered from the "knowledge of problem solving" block. It is assumed that these processes are all brought into play as users acquire their knowledge of the specific operators, command verbs (labels) and argument structures incorporated in the decoding task. It is further assumed that the three domains of cognitive processing are mutually interdependent. The representation in the Block Interaction Model merely describes this interdependence as static relationships indicated by arrows. This type of model does not attempt to characterize the detailed dynamics of information processing, the detailed knowledge used or the precise sequencing of mental activities. Other models would be required for these purposes (see Morton *et al.*, 1979). However, the Block Interaction Model does allow us to represent and summarize different patterns of interdependencies which influence performance in human–system dialogue.

The evidence of the third study suggests that the processes of natural cognition and language give rise to different patterns of interdependencies with positionally consistent and positionally inconsistent systems. These are contrasted in Figs 8(a) and 8(b). In the case of positionally consistent systems [Fig. 8(a)] inferential processes are able to abstract and apply a rule specifying the constant position of the recurrent argument. This is indicated by the strong relationship (bold arrows) between the inferential processes and the representation of the commands and their arguments. The application of the positional rule can be viewed as a behavioural means of trapping any order errors when arguments are selected. Hence, a strong relationship is indicated between the inferential processes and the task step of selecting and ordering arguments. However, the advantage of having the recurrent argument in the first rather than the second position suggested that processes associated with the structural analysis of a command string also exert an influence on the ease of learning the details of the arguments used with the individual commands (see section 5.7). This was a content-free influence, possibly attributable to first and second arguments having differential status in the information structure of the command string, and is indicated by the bold arrows linking structural processes and argument selection in Fig. 8(a). Although content-specific semantic processes must be brought into play in the course of human–system dialogue, there was no evidence to suggest that they differentiated between performance using the two alternative positionally consistent systems. Accordingly, the role of semantic processes remains unemphasized in Fig. 8(a). In effect, any direct influence of the meaning of the commands and their arguments on the order of entry is suppressed by the presence of the positional rule.

In the case of systems with variable positioning of the recurrent argument, a rather different pattern applies. First, the evidence indicated that inferential processes were unable to abstract, either explicitly or implicitly, a rule concerning the order of direct and indirect objects of the command verbs. As such only weak relationships (broken lines) are represented between inferential processes and the other components of Fig. 8(b). This does not mean that users fail to utilize inferential processes at all. However, the weak relationship indicates that in this context the inferential processes cannot abstract and apply a systematic rule.

Similar considerations apply with the effects of structural processes. With the positionally inconsistent systems we cannot exclude a potential influence of cognitive processes which analyse the information structure of the command strings. However, this factor would not discriminate between the direct object first and the direct object second orders. In both cases the recurrent and variable arguments each occupied the first position in the argument structure for half of the commands. Hence, any differential effects on the learning and use of the first argument would cancel out. As such, structural processes are represented as only a weak influence on the representation of the command set and on the task step of selecting arguments and their order. In contrast to Fig. 8(a), the main burden of learning must fall on the representation of individual commands, their arguments and the specific argument order for each command. Obviously, in order to build up such representations, both structural and semantic information must be used. This is indicated by the relationship between structural and semantic processes. However, under circumstances involving positional uncertainty the evidence of Study III indicates that structural effects are less salient, and what remains is a content-specific semantic influence. This exerts a small but systematic bias on command representation and argument entry. This is marked by the bold arrows in Fig. 8(b).

Finally, Figs 8(a) and 8(b) also represent the effects of display compatibility in terms of a direct influence of displayed information on the process of selecting arguments and their order. There was no evidence that the effects of display compatibility interacted with the nature of the underlying command structure. Errors were as frequent with positionally consistent and inconsistent systems. This suggests that errors induced by an incompatibility between a displayed order of arguments and their order of entry are independent of the errors arising as a consequence of the user's view of the operators and labels. As such they are likely to have a different type of cause, possibly some independent factors such as momentary lapses of attention or systematic strategies for entering arguments when the user is in doubt (e.g. enter the first appropriate argument read off the VDU with a left-to-right reading pattern).

In conclusion, the picture of the multiple underpinnings of usability as represented in Figs 8(a) and 8(b) is intended as a recipe for further research and development. These figures present a rather more complex picture than a straightforward analysis of "consistency" and "compatibility". In this sense they constitute revisions of our original working hypotheses by attempting to summarise and mark critical influences and the contexts in which they occur. However, the detailed interrelationships and the effects from which they were inferred should not be taken as hard and fast guidelines for the system designer. Nevertheless, they do represent a challenge to the design process both by pinpointing objective points of concern and by suggesting ways in which alternative design options could be modelled in prototype systems, pre-tested

on a sample user population and evaluated prior to their introduction on a commercial scale.

We would like to thank Tom Moran, Thomas Green, Richard Young and four anonymous referees for their critical comments on an earlier draft of this paper.

References

ALLOWAY, R. M. (1979). Decision support systems and information flows in the 1980's. In BOUTMY, E. J. & DANTHINE, A., Eds, *Teleinformatics '79.* Amsterdam: North-Holland, pp. 3–8.

BAECKER, R. (1979). Human–computer interactive systems: A state-of-the-art review. *Proceedings of the Second International Conference on Processing of Visible Language*, Naigara-on-the-Lake, Canada, September.

BARNARD, P. J. & MARCEL, A. J. (In press). Representation and understanding in the use of symbols and pictograms. In EASTERBY, R. & ZWAGA, H., Eds, *Visual Presentation of Information.* London: Wiley.

BARNARD, P. J., MORTON, J., LONG, J. & OTTLEY, P. (1977). *Planning menus for display: Some effects of their structure and content on user performance. IEE Conference Publication Number 150*, pp. 130–133.

BENNETT, J. L. (1972). The user interface in interactive systems. In CUADRA, C. A., Ed., *Annual Review of Information Science and Technology*, **7**, Washington, D.C., pp. 159–196.

BENNETT, J. L. (1979). Incorporating usability into system design. *Design '79 Symposium*, Monterey, California, April.

BROOKS, F. P. (1977). The computer "scientist" as toolsmith—studies in interactive computer graphics. In GILCHRIST, P., Ed., *Information Processing 77.* Amsterdam: North-Holland.

CARD, S. K., MORAN, T. P. & NEWELL, A. (1980). Computer text-editing: An information-processing analysis of a routine cognitive skill. *Cognitve Psychology*, **12**, 32–74.

CHAPANIS. A. (1969). Human factors engineering: The basics and the background. *Consulting Engineer*, **32**, 117–123.

CHAPANIS, A. (1979). Quo vadis, Ergonomia? *Ergonomics*, **22**, 595–605.

CODD, E. F. (1974). Seven steps to rendezvous with the casual user. *IBM Research Report RJ-1333.*

EASON, K. D., DAMODARAN, O. & STEWART, T. F. M. (1974). A survey of man–computer interaction in commercial applications. *Human Sciences and Advanced Technology Report No 144*, Loughborough University.

ENGEL, S. E. & GRANDA, R. E. (1975). Guidelines for man/display interfaces. *IBM Technical Report TR 00.2720.*

FITTER, M. (1979). Towards more "natural" interactive systems. *International Journal of Man–Machine Studies*, **11**, 339–350.

GAINES, B. R. (1978). Man–Computer communication—what next? *International Journal of Man–Machine Studies*, **10**, 225–232.

GAINES, B. R. & FACEY, P. V. (1976). Programming interactive dialogues. In PARKIN, A., Ed., *Conference on "Computing and People"*, Leicester, December.

GREEN, T. R. G. (1980). Programming as a cognitive activity. In SMITH, H. T. & GREEN, T. R. G., Eds, *Human Interaction with Computers.* London: Academic Press.

GREENE, J. M. (1972). *Psycholinguistics.* Penguin: Harmondsworth.

HALLIDAY, M. A. K. (1967). Notes on transitivity and theme in English. Part 1. *Journal of Linguistics*, **3**, 199–244.

HALLIDAY, M. A. K. (1968). Notes on transitivity and theme in English. Part 2. *Journal of Linguistics*, **4**, 179–215.

HAMMOND, N. V., LONG, J. B. & CLARK, I. A. (1978). Introducing the interactive computer at work: The users' views. *Proceedings of Workshop on Computing Skills and Adaptive Systems*, Liverpool, March, pp. 127–144.

HAMMOND, N. V., LONG, J. B., CLARK, I. A., BARNARD, P. J. & MORTON, J. (1980). Documenting human–computer mismatch in the interactive system. *Proceedings of the Ninth International Symposium on Human Factors in Telecommunications*, Holmdel, New Jersey, September, 17–24.

HILL, I. D. (1972). Wouldn't it be nice if we could write programs in ordinary English—or would it? *Computer Bulletin*, **16**, 306–312.

HORNBY, P. A. (1972). The psychological subject and predicate. *Cognitive Psychology*, **3**, 632–642.

HORNBY, P. A. (1974). Surface structure and presupposition. *Journal of Verbal Learning and Verbal Behavior*, **13**, 530–538.

JOHNSON-LAIRD, P. N. (1977). Psycholinguistics without linguistics. In SUTHERLAND, N. S., Ed., *Tutorial Essays in Psychology (Vol 1)*. Hillsdale, New Jersey: Erlbaum.

KENNEDY, T. C. S. (1975). Some behavioural factors affecting training of naive users of an interactive computer system. *International Journal of Man–Machine Studies*, **7**, 817–834.

LONG, J. B., DENNETT, J., BARNARD, P. J. & MORTON, J. (1977). Effect of display format on the direct entry of numerical information by pointing. *IEE Conference Publication Number 150*, pp. 134–137.

MARTIN, J. (1973). *The Design of Man–Computer Dialogues*. Englewood Cliffs, New Jersey: Prentice–Hall.

MILLER, L. A. & THOMAS, J. C. (1977). Behavioral issues in the use of interactive systems. *International Journal of Man–Machine Studies*, **9**, 509–536.

MORAN, T. P. (1978). Introduction to the command language grammar. *Report SSL-78-3, AIP Memo 111*, Xerox Corporation, Palo Alto.

MORTON, J., BARNARD, P. J., HAMMOND, N. V. & LONG, J. B. (1979). Interacting with the computer: A framework. In BOUTMY, E. J. & DANTHINE, A., Eds, *Teleinformatics '79*. Amsterdam: North-Holland, pp. 201–208.

SINGLETON, W. T. (1976). *The human factors file: Unit 5*. Open University Press: Milton Keynes.

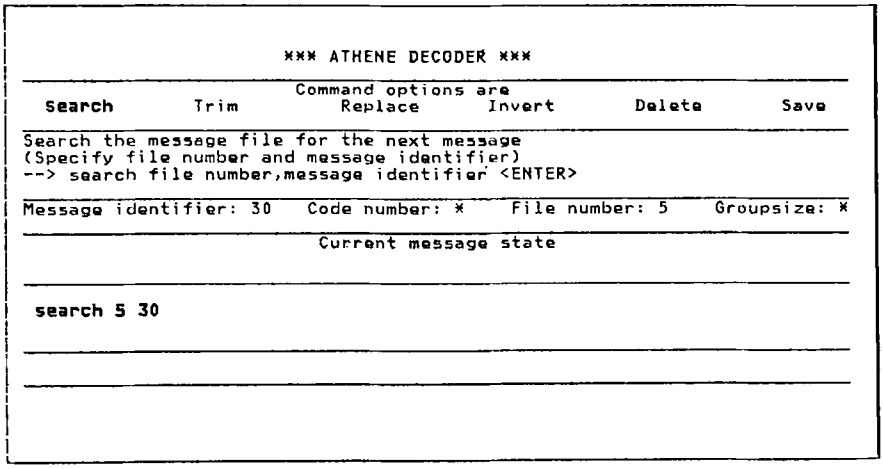
WRIGHT, P. (1978). Feeding the information eaters: Suggestions for integrating pure and applied research on language comprehension. *Instructional Science*, **7**, 249–312.

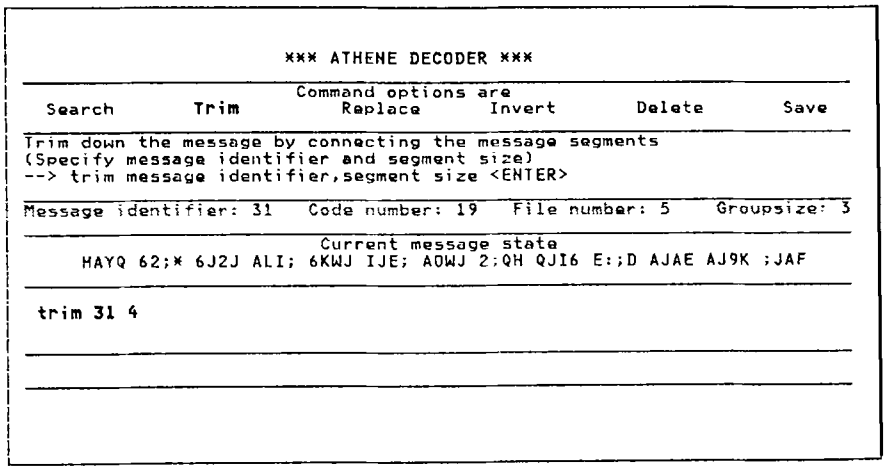
WRIGHT, P. (1980). Strategy and tactics in designing forms. *Visible Language*, ***XIV***(2), 151–193.

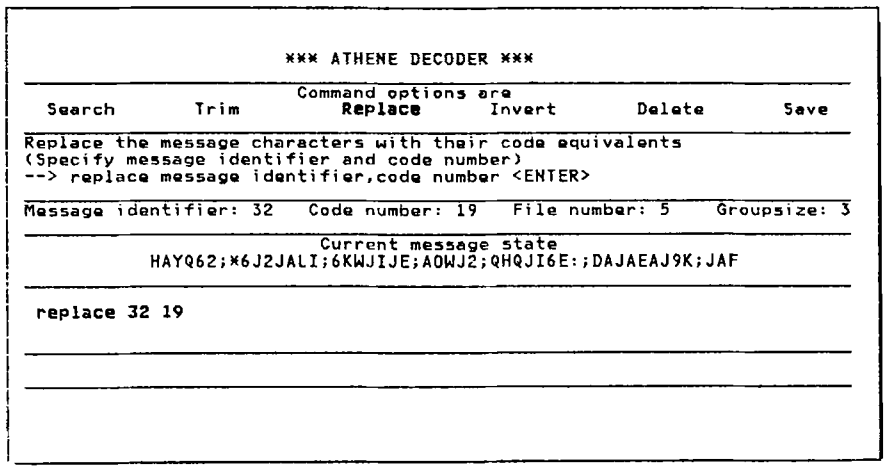
WRIGHT, P. & BARNARD, P. J. (1975). Just fill in this form: a review for designers. *Applied Ergonomics*, **6**, 213–220.

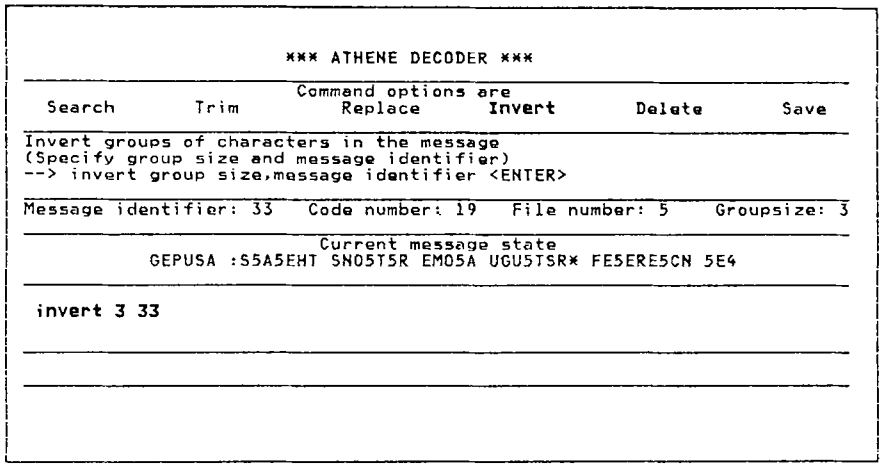
Appendix 1

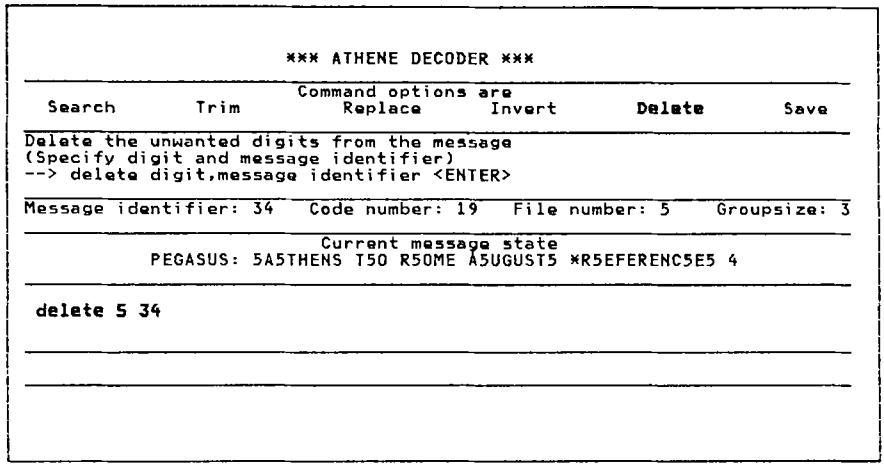
Example sequence of display panels from Study III. The example shows the decoding of the second training message. Each panel shows the state of the screen after the subject has typed the command string but before she has pressed the ENTER key which initiates the execution of the command. Arguments are organized on the principle of direct object first. Bold type indicates characters displayed on the screen at heightened brightness. The sections of each panel separated by lines are, from the top, (i) Command option field, with the current command at heightened intensity, (ii) Instruction field, (iii) Argument field, (iv) Message field, (v) Input field and (vi) Error message field.

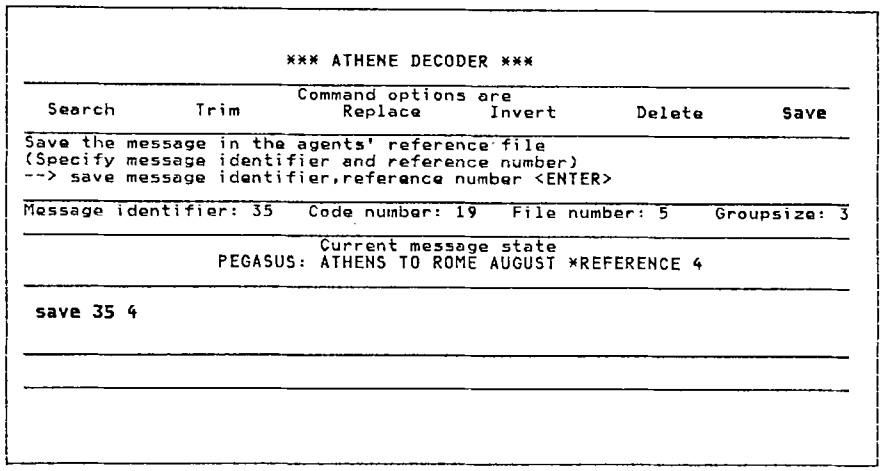

*** ATHENE DECODER ***
Command options are
Search Trim Replace Invert Delete Save
Search the message file for the next message
(Specify file number and message identifier)
--> search file number,message identifier <ENTER>
Message identifier: 30 Code number: * File number: 5 Groupsize: *
Current message state
search 5 30

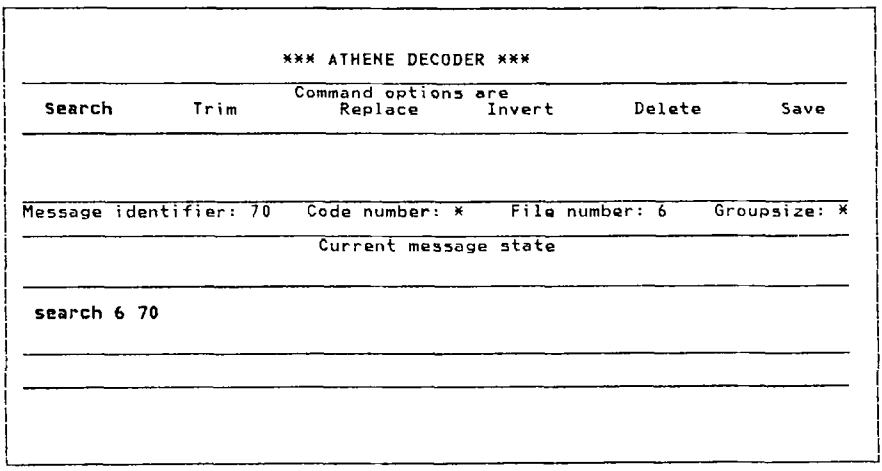

*** ATHENE DECODER ***
Command options are
Search Trim Replace Invert Delete Save
Trim down the message by connecting the message segments
(Specify message identifier and segment size)
--> trim message identifier,segment size <ENTER>
Message identifier: 31 Code number: 19 File number: 5 Groupsize: 3
Current message state
HAYQ 62;* 6J2J ALI; 6KWJ IJE; AOWJ 2;QH QJI6 E:;D AJAE AJ9K ;JAF
trim 31 4


*** ATHENE DECODER ***
Command options are
Search Trim Replace Invert Delete Save
Replace the message characters with their code equivalents
(Specify message identifier and code number)
--> replace message identifier,code number <ENTER>
Message identifier: 32 Code number: 19 File number: 5 Groupsize: 3
Current message state
HAYQ62;*6J2JALI;6KWJIJE;AOWJ2;QHQJI6E:;DAJAEAJ9K;JAF
replace 32 19


*** ATHENE DECODER ***
Command options are
Search Trim Replace Invert Delete Save
Invert groups of characters in the message
(Specify group size and message identifier)
--> invert group size,message identifier <ENTER>
Message identifier: 33 Code number: 19 File number: 5 Groupsize: 3
Current message state
GEPUSA :S5A5EHT SNO5T5R EMO5A UGU5TSR* FE5ERE5CN 5E4
invert 3 33


*** ATHENE DECODER ***
Command options are
Search Trim Replace Invert Delete Save
Delete the unwanted digits from the message
(Specify digit and message identifier)
--> delete digit,message identifier <ENTER>
Message identifier: 34 Code number: 19 File number: 5 Groupsize: 3
Current message state
PEGASUS: 5A5THENS T5O R5OME A5UGUST5 *R5EFERENC5E5 4
delete 5 34


*** ATHENE DECODER ***
Command options are
Search Trim Replace Invert Delete Save
Save the message in the agents' reference file
(Specify message identifier and reference number)
--> save message identifier,reference number <ENTER>
Message identifier: 35 Code number: 19 File number: 5 Groupsize: 3
Current message state
PEGASUS: ATHENS TO ROME AUGUST *REFERENCE 4
save 35 4



Appendix 2

STUDY III: ANALYSIS OF VARIANCE SUMMARY TABLES

1. *Analyses comparing the three groups (D.O.1st, D.O.2nd & Consistent)*

Key: G, Groups (3); CS, Command sets (2); M, Messages (8: data from training messages excluded); C, Command operations (6).

		Viewing time			Information requests		
Source	*df*	Srce MS	Err. MS	*F*	Srce MS	Err. MS	*F*
G	2,42	1·266	0·232	5·46	17·540	4·652	3·77
CS	1,42	0·047	0·232	0·20	6·355	4·652	1·37
G×CS	2,42	0·037	0·232	0·16	7·344	4·652	1·58
M	7,294	4·795	0·039	123·70	6·103	0·157	41·47
M×G	14,294	0·106	0·039	2·74	0·112	0·157	0·76
M×CS	7,294	0·053	0·039	1·38	0·032	0·157	0·22
M×G×CS	14,294	0·069	0·039	1·79	0·229	0·157	1·55
C	5,210	0·599	0·040	15·09	3·431	0·177	19·31
C×G	10,210	0·044	0·040	1·12	0·324	0·177	1·82
C×CS	5,210	0·120	0·040	3·03	0·244	0·177	1·37
C×G×CS	10,210	0·054	0·040	1·37	0·134	0·177	0·75
M×C	35,1470	0·700	0·023	29·92	0·147	0·083	1·84
M×C×G	70,1470	0·025	0·023	1·05	0·078	0·083	0·94
M×C×CS	35,1470	0·017	0·023	0·71	0·079	0·083	0·95
M×C×G×CS	70,1470	0·022	0·023	0·93	0·074	0·083	0·89

Source	*df*	Total argument errors			Reversed argument errors		
		Srce MS	Err. MS	*F*	Srce MS	Err. MS	*F*
G	2,42	53·94	13·61	3·96	4·131	0·751	5·50
CS	1,42	2·19	13·61	0·16	0·191	0·751	0·25
G×CS	2,42	2·45	13·61	0·18	0·285	0·751	0·38
M	7,294	8·98	0·86	10·48	0·071	0·080	0·89
M×G	14,294	1·10	0·86	1·29	0·072	0·080	0·90
M×CS	7,294	1·72	0·86	2·01	0·026	0·080	0·32
M×G×CS	14,294	1·84	0·86	2·15	0·100	0·080	1·26
C	5,210	21·83	2·19	9·96	0·500	0·201	2·48
C×G	10,210	2·81	2·19	1·28	0·318	0·201	1·58
C×CS	5,210	2·95	2·19	1·34	0·140	0·201	0·70
C×G×CS	10,210	1·75	2·19	0·80	0·064	0·201	0·32
M×C	35,1470	1·69	0·68	2·47	0·096	0·064	1·50
M×C×G	70,1470	0·96	0·68	1·40	0·098	0·064	1·53
M×C×CS	35,1470	0·80	0·68	1·18	0·078	0·064	1·22
M×C×G×CS	70,1470	0·48	0·68	0·70	0·072	0·064	1·12

2. Analyses comparing the groups D.O.1st & D.O.2nd

Key: G, Groups (2); CS, Command sets (2); M, Messages (8: data from training messages excluded); C, Command operations (6).

Source	*df*	Viewing time			Information requests		
		Srce MS	Err. MS	*F*	Srce MS	Err. MS	*F*
G	1,28	0·533	0·265	2·01	0·010	5·480	0·00
CS	1,28	0·055	0·265	0·21	2·042	5·480	0·37
G×CS	1,28	0·064	0·265	0·24	13·500	5·480	2·46
M	7,196	3·966	0·043	91·98	4·720	0·159	29·75
M×G	7,196	0·082	0·043	1·90	0·138	0·159	0·87
M×CS	7,196	0·120	0·043	2·79	0·060	0·159	0·38
M×G×CS	7,196	0·028	0·043	0·64	0·235	0·159	1·48
C	5,140	0·425	0·047	9·01	2·952	0·195	15·15
C×G	5,140	0·048	0·047	1·01	0·243	0·195	1·25
C×CS	5,140	0·128	0·047	2·71	0·156	0·195	0·80
C×G×CS	5,140	0·054	0·047	1·15	0·195	0·195	1·00
M×C	35,980	0·512	0·026	19·87	0·113	0·092	1·23
M×C×GS	35,980	0·019	0·026	0·74	0·094	0·092	1·03
M×C×CS	35,980	0·023	0·026	0·91	0·043	0·092	0·47
M×C×G×CS	35,980	0·023	0·026	0·89	0·082	0·092	0·80

Source	*df*	Total argument errors			Reversed argument errors		
		Srce MS	Err. MS	*F*	Srce MS	Err. MS	*F*
G	1,28	36·88	16·81	2·19	0·344	0·979	0·35
CS	1,28	0·51	16·81	0·03	0·032	0·979	0·03
G×CS	1,28	4·17	16·81	0·25	0·475	0·979	0·48
M	7,196	8·26	1·01	8·15	0·087	0·109	0·80
M×G	7,196	0·36	1·01	0·36	0·109	0·109	1·00
M×CS	7,196	2·35	1·01	2·32	0·056	0·109	0·51
M×G×CS	7,196	2·11	1·01	2·08	0·147	0·109	1·34
C	5,140	19·79	2·76	7·16	0·491	0·261	1·88
C×G	5,140	3·08	2·76	1·11	0·669	0·261	2·56
C×CS	5,140	2·45	2·76	0·89	0·098	0·261	0·37
C×G×CS	5,140	3·11	2·76	1·12	0·081	0·261	0·31
M×C	35,980	2·16	0·87	2·48	0·126	0·087	1·45
M×C×G	35,980	0·98	0·87	1·12	0·147	0·087	1·70
M×C×CS	35,980	0·88	0·87	1·01	0·112	0·087	1·30
M×C×G×CS	35,980	0·59	0·87	0·68	0·091	0·087	1·05

3. Analyses comparing the two groups Consistent 1st & 2nd

Key: G, Groups (2); CS, Command sets (2); M, Messages (8: data from training messages excluded); C, Command operations (6).

Source	*df*	Viewing time			Information requests		
		Srce MS	Err. MS	*F*	Srce MS	Err. MS	*F*
G	1,12	0·157	0·178	0·88	12·251	2·145	5·71
CS	1,12	0·002	0·178	0·01	5·501	2·145	2·56
G×CS	1,12	0·015	0·178	0·09	3·939	2·145	1·84
M	7,84	0·959	0·034	28·41	1·469	0·125	11·79
M×G	7,84	0·010	0·034	0·30	0·147	0·125	1·18
M×CS	7,84	0·044	0·034	1·31	0·195	0·125	1·56
M×G×CS	7,84	0·005	0·034	0·16	0·097	0·125	0·78
C	5,60	0·215	0·026	8·35	0·884	0·154	5·75
C×G	5,60	0·013	0·026	0·49	0·061	0·154	0·39
C×CS	5,60	0·047	0·026	1·81	0·161	0·154	1·05
C×G×CS	5,60	0·026	0·026	1·00	0·104	0·154	0·68
M×C	35,420	0·218	0·019	11·61	0·102	0·065	1·57
M×C×G	35,420	0·021	0·019	1·11	0·082	0·065	1·25
M×C×CS	35,420	0·014	0·019	0·72	0·101	0·065	1·55
M×C×G×CS	35,420	0·015	0·019	0·80	0·066	0·065	1·01

Source	*df*	Total argument errors		
		Srce MS	Err. MS	*F*
G	1,12	6·56	6·99	0·94
CS	1,12	2·41	6·99	0·34
G×CS	1,12	10·31	6·99	1·48
M	7,84	2·56	0·54	4·74
M×G	7,84	0·50	0·54	0·92
M×CS	7,84	0·96	0·54	1·77
M×G×CS	7,84	0·64	0·54	1·18
C	5,60	4·57	1·12	4·08
C×G	5,60	0·46	1·12	0·41
C×CS	5,60	0·88	1·12	0·79
C×G×CS	5,60	0·79	1·12	0·71
M×C	35,420	0·46	0·29	1·57
M×C×G	35,420	0·41	0·29	1·40
M×C×CS	35,420	0·29	0·29	0·98
M×C×G×CS	35,420	0·32	0·29	1·10