Division of Research
Graduate School of Business Administration
The University of Michigan

May 1984

KNOWLEDGE ORGANIZATION FOR COMMAND LANGUAGES

Working Paper No. 380

Barry D. Floyd

Marilyn M. Mantei

The University of Michigan

Abstract

In this study we capture the internal mental organization of a command language that is maintained by expert users of an interactive computer system. When this representation is violated, we believe that errors will occur lowering the expert's productivity.

To determine the user's internal mental organization for a command language, a reading time experiment is performed using an experimental psychology paradigm. The language under study is MTS, an operating system command language in use at the University of Michigan. Seven expert MTS users participated in the study. From the temporal pattern of reading time responses, inferences are made about how expert MTS users organize the commands in long term memory. Results indicate that a command is not treated as a unitary chunk but is composed of substructures comprised of the command verb and its required positional parameters, and of keywords and their required keyword parameters. This organization is hypothesized to result in part from the level of interaction (as measured by the instances when the system responds to the user) designed into the dialogue.

These results are expected to impact the design of command languages by providing insight into when the system should interrupt user behavior and how the system should request additional information or respecification of previously entered information, such as in error correction.

1. Introduction

With the rising cost of the human resource and the increasing use of online information systems in the business environment a serious concern over user productivity in the use of interactive systems has arisen. This concern is well-founded.

Reports indicate that improving the productivity of computer system users can have a major impact on costs. For example, Shneiderman (1982) reports on a Quality Inn study which indicates that for each second deducted from the average 150 seconds taken to make a room reservation, $40,000 per year is saved. Olson (1983) reports that for AT&T, each one second reduction in the average of 30 seconds spent per telephone operator transaction (e.g., requests for a telephone number), translates into a savings of $20,000,000 per year. In each of these two examples the operators must interact with online computer systems to perform their tasks.

Other research indicates that users of online information systems take a large amount of time to correct errors (Norman 1981; Card, Moran and Newell, 1983). Given the above costs for shaving seconds off the performance of a task, errors which typically take many seconds to correct, can be expensive. The research described in this paper addresses this error problem indirectly.

In order to understand why users make errors, we must understand how they use command languages. As a first step in this understanding, we have undertaken looking at the internal memory structure users build of command languages. We believe that this structure has a role in error generation in the following way. First, we believe that information is organized in human memory into many small groupings which psychologists call "chunks". Whenever a user is required

to access a "chunk" of information in memory, the access is complete, that is, the entire contents of the chunk are retrieved. If only a portion of the chunk is needed in the interaction, additional processing is required for the selection of the portion needed. Because the access and use of this piece of information does not typically require additional processing (or it would not have been stored as a single unit to begin with), a conflict exists between performing the usual behavior of retrieving and using the entire chunk versus that of performing the additional processing of selecting a portion of the retrieved chunk. The execution of the entire "chunk" wins this processing conflict leading to the error of outputting too much information. For example, we often give entire seven digit telephone numbers to members of a work place who know the first three digits of the exchange.

We do not show in this paper that these errors take place. Our work is rather a first step to this end. It illustrates that a memory organization of command languages exists and what this organization looks like for a typical command language.

We are applying the theories of knowledge organization found in the field of cognitive psychology to our study of command languages. The use of chunking as previously noted is among these theories. As an example of chunking, consider the group of characters "IBMRCAET". Most individuals view these characters as consisting of three groups: "IBM", "RCA", and "ET". These are very familiar character sequences found in our environment which we have learned to associate together. These groups are called "chunks" in psychology, and, as noted, humans are believed to store information in memory in these "chunked" groupings, (Simon, 1974). The chunks represent an organized unit of information which has some meaning and is "processed" as a whole. By knowing that people group information in such a way, we can predict that the sequence "IBM-RCA-ET" is much easier to perceive and to recall than the character sequence "IB-MRC-AE-T". We explain this by stating that the presentation of the material matches the internal memory organization stored by the average individual.

The theory that people chunk information into meaningful units can be applied to computer usage. If we can determine what pieces of the command language the user of an online system views as a chunk, then the system can take advantage of this knowledge and facilitate system use. For example, if the user makes an error in the specification of a word within a command sequence that is stored as a chunk, the error notification procedure should expect the corrected sequence generated by the user to begin at the chunk boundary of the command sequence in error. An example is the keyword specification of a command parameter. Many systems request the user to respecify the parameter value whenever an error is made. However, users also specify the keyword in addition to the parameter value because the keyword is a vital part of specifying the parameter value. System designs which take this effect into account might accept these command words (which are part of the same chunk) and which precede the word being corrected. Alternatively, the system might "prime" the user by typing out all the words in the chunk preceding the one needing correction and then wait for the user to type in the corrected word.

To investigate this theory in the area of online information systems, we chose to look at the type of command languages found on many systems today, a user initiated command language. This type of system forces the user to learn the syntactic structure of the command language and to repeat this structure perfectly to initiate a command. An example command from such a language is:

COPY FROM FILE1 TO FILE2

This command tells the computer system to copy the contents of the computer file named "FILE1" to the computer file named "FILE2".

We have applied methods from experimental psychology to determine what information is chunked together when learning an online system. The methodology is based on the premise that people pause between chunks. For example, in saying the characters "IBMRCAET", the average

American would likely say "IBM", pause, and then say "RCA", pause, and finally say "ET". These pauses are indicative of the chunk boundaries.

This research asks individuals who are experts in the use of a particular online system to view a series of commands from that system and to later recall them. They are shown the command sequences one word at a time. The amount of time they spend viewing each word in sequence is under their control, but is also recorded. From related work on English sentences, we expect these experts to pause longer at words occurring at chunk boundaries. We therefore analyze recorded temporal viewing patterns for extended pauses which will indicate the parts of the command that are systematically grouped together.

There are four other sections to this paper. Section 2 lays the groundwork for understanding this research giving both explicit psychological definitions to several of the terms we use and bringing in relevant research on knowledge structure from the areas of cognitive psychology and psycholinguistics. Section 3 describes the experiment we ran on our seven expert system users and Section 4 discusses the results of this study and their implications for the design of command languages. A final fifth section summarizes our research presentation and makes several suggestions for the design of interactive command languages based on our conclusions.

2. Related Work on Memory Structure

The first part of this section lays the theoretical groundwork for understanding the research presented. The second part discusses related research which supports our concluding suggestions for an interactive command language design which does not disrupt command language knowledge organization.

In this paper we focus on the structure of the human - computer interaction and its impact on the memory organization the user creates of the interaction. An important aspect of this concept is the degree to which the user instructs the computer without interruption, or a computer response. We refer to this concept as the level of interaction. The degree to which an interaction is uninterrupted is measured by the number of required constituents of the command (or commands) the user can enter sequentially without needing a system response. We believe that these uninterrupted command sequences form the basis for the memory chunks that develop for a command language and that the user uses this internalized chunk structure in subsequent productions of commands. We also believe that the substitutable constituents within the command form the basis for a more micro-organization of single command sequences.

Although we believe that disruption of this knowledge organization results in user errors, our first chore is to illustrate that the hypothesized memory organization for command language exists. The discussion of studies on memory organization which follows and the research described in this paper provide evidence for the existence of highly organized command language structures in experts. We support our belief that violations of this knowledge organization impairs performance by citing related research on language which shows this to occur. Nevertheless, we note that at some point in time, this second premise must be tested experimentally.

In looking at the relevent research literature we were interested in addressing two questions. The questions are:

1. Can support be found in the literature for the type of knowledge organization we expect to occur in the domain of command languages?
2. Does research exist which indicates that violations of the knowledge organization of a language has a negative impact on performance?

We present research support addressed by these questions in the following paragraphs.

From the work in other task domains there exists proof that humans build internal knowledge organizations. The study of expert behavior in domains such as chess, Go, and physics (Simon and Barenfeld, 1969; Chase and Simon, 1973; Reitman, 1976; Larkin, McDermott, Simon and Simon, 1980) provides us with examples of such research. In many of these studies the experts are shown a complex display of information associated with their domain of expertise and asked to reproduce it after looking at it briefly. Experts recall the display in "bursts", i.e., rapid short verbalizations or physical movements followed by a pause. It is inferred that these bursts comprise one chunk or one memory organizational unit.

For example, in the chess research by Chase and Simon (1973) where experts were asked to reproduce chessboard configuratons, expert subjects placed four or five chess pieces down, paused, and then placed another group of pieces down, and so on. These pauses are denoted as interresponse times and are believed to be the boundaries of organizational units in memory. Chess experts differ from novices in that the number of pieces they place down between pauses are larger and take the form of strategic chess patterns (as opposed to say, all the pieces in the first row). Although novices and experts are reproducing only that part of the board pattern which they can store in a temporary memory area, the differences in performance lie in what is stored in this memory area. For the novice, it is the individual chess pieces and their positional relationships. For the expert, it is pointers to previously learned chess patterns that are stored in permanent memory. The pauses between the placement of each chess pattern by the experts is believed to reflect the memory access taking place for the next organizational unit. Thus, by measuring these pauses, psychologists can obtain an idea of what the internal organizational units are.

Aaronson and Scarborough (1976, 1977) used a slightly different technique in studying how people perceptually coded sentences based upon the subsequent task the subjects were to perform. In their research they presented sentences to subjects one word at a time where the

time spent viewing each word was controlled by the subject. After viewing a sentence the subjects either answered a question about the sentence (a comprehension task) or wrote down as much of the sentence as they could remember (a recall task). The subjects knew in advance the task they were to perform.

Aaronson and Scarborough found that for the recall task, subjects' reading times reflected "the syntactic structure of the sentence, with prolonged pauses at phrase boundaries and bowed reading-time curves within phrases". For the comprehension task, subjects' reading times reflected "the semantic content with prolonged reading times occurring at important content words". They found that subject interresponse time behavior and thus, internal coding of the sentences was determined not only by the structure of the sentences, but by the demands of the task following presentation of the sentences.

We have adopted the Aaronson and Scarborough recall task methodology in our research and are using it to show the internal memory organization of command languages. We do not use the comprehension task portion of their work for the following reason. We are interested, ultimately, in the production of a command sequence, i.e., when the interactive user actually types in the set of words that form the request being made of the computer. This sequence, we believe is governed by the necessity to provide the correct syntax for computer interpretation and therefore must be based on some internal memory organizing scheme for generating this syntax. We believe that the memory organization which preserves syntax is best captured by the Aaronson and Scarborough recall task. There may very well be other internal organizations that would be pulled out by the comprehension task. These, we believe, would be useful for the problem solving part of the interaction, perhaps organizing the entire sequence of commands necessary to do a task. This is not our focus here.

Research in psycholinguistics provides support for the second question on the deleterious

effect of disrupting chunk boundaries. Aaronson and Scarborough show that the phrase is an internal organizing unit. If we interrupt the processing of a phrase during reading, we should expect a decrease in reading performance. Graf and Torrey (1966) found this to be true for comprehension. Figure 1 is an example of stimuli used in their experiment. They found that when the physical boundaries of typed text matched the phrase boundaries better comprehension scores were obtained from subjects than when the phrase boundaries were disrupted by the spatial placement of the text on the page.

During World War II, even fantastic schemes received consideration if they gave promise of shortening the conflict.	During World War II, even fantastic schemes received consideration if they gave promise of shortening the conflict.
Matched	Unmatched

Figure 1. Example forms of matching and mismatching physical and linguistic boundaries (Graf and Torrey, 1966).

Other researchers found that a violation of phrase boundaries also play an important role in the processing required to perform a task. Maclay and Osgood (1959) showed that speakers restarted at phrase boundaries when they made an error within a phrase. The complete restart is projected to be easier than the processing task of finding the restart point within the chunk. Oliver and Ericsson (1983) demonstrated that actor's recall of words from their memorized parts differed in speed when the words used to prompt them were from the same sentence as compared to when the words were from different, but still contiguous sentences. They hypothesized that the phrases and sentences were chunks, and, although they found that actors can begin recall within their parts, they only do so at these chunk boundaries.

The described research indicates that people do internally structure knowledge. Furthermore, indications are that when this structure is disrupted, our performance tends to decrease for the task at hand. The research described here studies languages of a slightly different sort than the natural languages just discussed. The languages are artificial. Moreover, they have rigid grammatical rules which must be followed and a limited set of alternative forms. Therefore, we believe that the effects found in the above research on English will be even more pronounced with the artificial interactive languages used in the world of computers. The experiment to test this belief is discussed in the next section.

3. The Experimental Capture of Command Language Chunks

This section lays the experimental basis for our claim that expert users of user initiated command language dialogues organize the commands into systematic chunks. We will first present the theory supporting our research hypotheses and then give a description of the experiment and the results obtained from the experiment.

We conducted the experiment to determine the internal chunks experts built for a user initiated command language dialogue. Before running the experiment we examined the command language and predicted where the chunk boundaries might occur. The command language we selected for our study is an interactive language in use at The University of Michigan called the Michigan Terminal System (MTS). We do not give a complete description of MTS in this paper, but we present an overview of MTS's syntax in the next three paragraphs. We then point to where in this syntax the memory organizing boundaries are expected to occur.

The syntax of the MTS command language is similar to many other command languages. All MTS command statements begin with a verb identifying the action to be performed. The verb is

followed by a list of positional parameters varying in number from one to four. These positional parameters are then followed by Keyword parameters. The system requires the positional parameters to follow a strict order while the Keyword parameters may be specified in any order. Each command verb requires the specification of associated positional parameters. However, in many cases, only the beginning positional parameters are required and the end positional parameters are optional. All Keyword parameters are optional, however, once a Keyword is specified, its argument must also be specified.

To illustrate this syntax and the memory organization we expect, we describe a command sequence from the experiment. Two commonly used commands are those which call in compilers to be executed and those which perform file transfers. In MTS, these commands with their English translations are shown in Figure 2. Both "*FTN" and "FROM FILE2 TO *PRINT*" are positional parameters. The phrases "INPUT=FILE1", "OUTPUT=FILE2", and "T=2" are Keyword parameters where "INPUT=", "OUTPUT=", and "T=" are each Keywords which declare respectively, the source file name, the source listing file name, and the compilation time limit.

Commands:

(1) RUN *FTN INPUT=FILE1 OUTPUT=FILE2 T=2

(2) COPY FROM FILE2 TO *PRINT*

English Translations:

(1) Bring in the Fortran compiler and compile the contents of FILE1 while placing the compiler listing on file FILE2. Let the maximum amount of allowable cpu time for this task be two seconds.

(2) Place a copy of FILE2 on the output queue for the printer.

Figure 2. Example MTS commands with English translations.

We do not expect the memory structure which users build from the MTS language to correspond to the structure of MTS except at select places. We expect each MTS command to be stored as a single "chunk". We believe that this occurs for three reasons. The first reason is based on the level of interaction designed into the dialogue. In MTS, the user types in the entire command, presses the enter key and waits for a system response before preceeding. Thus, for normal error-free usage, the level of interaction of the system is at the single command level. Since the wait is variable and may be long, a user may wish to do other thinking during this time. This is easiest if commands are stored as chunks.

Our second reason arises from the semantics associated with each command. MTS is designed so that each command performs a particular task, (e.g., copying, or emptying a file). It is natural for the user to store the command phrase as a distinct unit in order to facilitate rearrangements of these individual task units.

Our third reason is based on the enormous experience the user has with expressing the command. Commands have a rigid format which is typed again and again by the user whenever the command is invoked. Unlike English sentences which can be phrased differently and still express the same concept, each command always begins with the command verb and, typically, the same positional parameters. This constancy falls apart at levels above the command level, thus we expect the command to be our largest "chunk".

Although we expect a command to be a major cognitive unit for expert users of MTS, we also expect the command to consist of smaller cognitive groupings that arise from the different ways in which the command can be expressed and still be a legitimate command in the language. Whenever a user has a choice between including or excluding a particular section of a command, that section will be stored as a distinct "chunk".

In MTS, the substructures that allow this variability in expression are default options which

include some positional parameters and all keyword parameters. If a parameter is not specified, the default value for it is taken. For example, in the "COPY" command shown below, the first parameter "FILE1" is required; if omitted, MTS provides an error message. There is no default. The second parameter "FILE2" is optional; if omitted, the system assumes that the file is to be copied to the user's terminal. Therefore we would expect the organizational boundaries to occur between "FILE1" and "FILE2".

```
COPY FILE1 FILE2
```

Similarly, all keyword parameters are optional for the MTS language system. They may also be used in any order. For example, to execute the FORTRAN compiler as shown in the command below, we could enter the words "OUTPUT=FILE2" before entering the words "INPUT=FILE1". However, note that once we enter a keyword for the command, such as "OUTPUT=" the specification of the argument for the keyword is required. Failure to specify a keyword argument following the keyword gives an error. Therefore we would expect each of the keyword expressions "INPUT=FILE1" and "OUTPUT=FILE2" to be stored as a single complete unit in the MTS expert's memory.

```
RUN *FTN INPUT=FILE1 OUTPUT=FILE2
```

In summary, we expect the users' knowledge organization for MTS commands to look as follows: The primary organizational unit is the command. This is divided into substructures which consist of either the command verb and its required parameters or each of the optional parameters used with the command. For keyword parameters, this consists of the keyword and the keyword argument. The experiment we ran to test this proposed memory organization is a reading time experiment. We therefore formulate our research hypotheses in terms of our expectations of user performance for this task.

3.1 Expected Reading Times

We used a reading/recall experiment to test our chunking predictions. In the experiment, we asked experts to view each word of a sequence of MTS commands individually, i.e., one word per screen display. The expert's task was to produce a verbatim recall of the command sequence immediately after viewing the last word in the sequence. Each expert controlled the amount of time each word in the command sequence was displayed by pressing a response key which erased the current word and displayed the following word. We then calculated the viewing time differences between single words and used these differences to provide evidence for our predicted chunk boundaries.

To operationalize our hypotheses we must state them to express what we expect to occur in the experiment at the command words which border chunk boundaries. For example, based on our expectations of the command words comprising substructures, for the command group shown below, we expect "CREATE FILE2" to form one chunk, "COPY FILE1" to form a second chunk, "FILE2", a third chunk, and "COPY FILE1 FILE2", a final fourth chunk which contains the second and third chunks.

```
CREATE  FILE2
COPY FILE1 FILE2
```

To facilitate our discussion let us define those chunk boundaries occuring within a command as being "minor boundaries" and those chunk boundaries which occur between commands as being "major boundaries". Our hypotheses regarding the command words bordering these boundaries are presented below.

When viewing the words one at a time, if the current word the subject is viewing is the last word in a chunk, i.e., one that precedes a chunk boundary, this boundary should be recognized by the subject. We expect the subject to review the words just seen at these boundaries. The

reviewing allows the user to preserve these words in memory and to recode them as a pointer to a single memory unit, i.e., a chunk, in preparation for the recall task to follow. This recoding reduces the memory load for the subject but can only occur if the subject already has stored away in memory the organizational units or chunks needed for the recoding. This is true of our MTS experts and we therefore expect longer viewing times for words that are the last word of what we believe is a chunk. This gives us our first hypothesis.

H1: Viewing times for words preceding minor boundaries will be larger than the average viewing times for words in the command group.

We expect this condition to occur for all of the substructures within a command because the subject knows them perfectly. This is not necessarily true for all commands. The subject does not know when he has reached the end of a command. In these cases, we expect that the word which follows a command boundary (a major boundary) will have a larger viewing time because the same reviewing process occurs, but the subject cannot tell in many cases when the command has ended until this word appears. This situation occurs commonly in MTS. Thus, we need two hypotheses to express the major boundary condition.

H2a: Viewing time for words following command boundaries will be larger than the average viewing time for words within a command group when the number and type of command parameters commonly used is variable.

H2b: Viewing time for words preceding command boundaries will be larger than the average viewing time for words within a command group when the same command parameters are always used.

The words preceding and following the major boundaries predicted for each MTS command should have longer viewing times than those for minor boundaries because of the more complex mental task of recoding a sequence of subchunks. Thus our third hypothesis is:

H3: Viewing times for major boundary words will be longer than viewing times for minor boundary words.

Let us consider the two sample commands used in the preceding discussion. For this

command sample, we would expect the viewing times for each word to be ordered as shown in Figure 3. In this figure, the words that distinguish the major boundaries have the longest viewing time followed by the minor boundary words. The start of the command sequence is shown as having the shortest viewing time because there is nothing in memory for the subject to recode at this point.

Example MTS command sequence

```
CREATE FILE1
COPY FILE1 FILE3
```

Longest time	FILE2, COPY, FILE3
↓	FILE1
Shortest time	CREATE

Figure 3. Reading time predictions for command words forming the above two MTS commands.

Now that we have laid out the reasoning underlying the predicted outcomes of our research, the rest of this section briefly describes the procedural details of the experiment.

3.2 Procedure

Subjects viewed 47 command groups displayed one word at a time on an IBM PC with an IBM 5151 monochrome green phosphor video display. The subjects controlled the length of time they viewed each word by pressing a response key when they had completed viewing the current word. The computer recorded the time between response key presses. Instructions for the task were provided via the terminal display. In these instructions subjects were told about the tasks they

had to perform, i.e., to view words and recall the words verbatim.

Each series of words was preceded by a screen stating that a trial was ready to begin and requesting the subject to press the response key when ready. This message was followed by a blank screen for 250 milliseconds to allow erasure of the message. Then the first word in the series appeared. Each word was blanked out when the next word was requested. All words were printed in capital letters and centered on the screen. Immediately after the last word was read the characters "@@@@@@" appeared on the screen for 500 milliseconds. The character string "@@@@@@" was used to signal the end of the stimulus because words such as "STOP" could be MTS command words or keyword arguments. These characters were followed by the word "RECALL".

Subjects then wrote a verbatim recall of the words that they read on the screen. They were instructed to produce their recall in the exact order in which they had viewed the words. They had no time limit for this task.

3.3 The Command Groups Used in the Experiment

The command groups consisted of either one MTS command, a single command group, or two to six MTS commands, a multiple command group. The single command groups contained two to twelve words. The multiple command groups contained five to twenty-five words.

Commonly used commands were selected for the experiment. Four MTS experts aided in this selection by rating a command's usage frequency. All except frequently used commands were discarded. Parameter names were selected systematically reflecting the semantic category they applied to. They were two syllables long except for time limits which were one syllable long. For example, the names for the source listing output of a compile command were selected to reflect the concept of listing, e.g., "LISTING" and "PRINTOUT".

The command groups were shown to the subjects in two sessions in a sixty minute period.

The first session consisted of either all single command groups or all multiple command groups. The second session consisted of the command group not yet seen by the subject. There were 24 command groups in the single command group session and 23 command groups in the multiple command group session with the first four command groups serving as practice trials in each session.

3.4 Subjects

Seven subjects participated in the experiment. The subjects were paid volunteers from the Michigan Computing Center consultant staff. Each subject was a Senior or Graduate Student with a strong background in computer science or computer engineering. These consultants are recognized MTS language experts, having passed an extensive exam to be hired as a counselor. They also have an average of over four years of daily experience using the MTS command language.

3.5 Results

The experimentally captured word reading times varied from a low of 330 milliseconds to a high of 3430 milliseconds. On average, subjects spent 910 milliseconds viewing each word with a standard deviation of 390 milliseconds.

The data was normalized by dividing the subject's average reading time for a command group into the reading times for each word in the command group. We normalized the data to lessen the effect of reading time differences between subjects and because we are interested in reading time differences between words within a command group and not between words in different command groups. We assume that frequent users of a command language will generate similar knowledge organizations. Therefore, we also averaged the reading times across subjects in order to determine the main effects of command language knowledge organization.

The graph shown in Figure 4 illustrates the averaged normalized reading time (ANRT) values

for the "LIST" command for each of its 3 occurrences in the experiment's command groups. Subjects were shown a form of the "LIST" command three different times, once in the single command session and twice in the multiple command session.

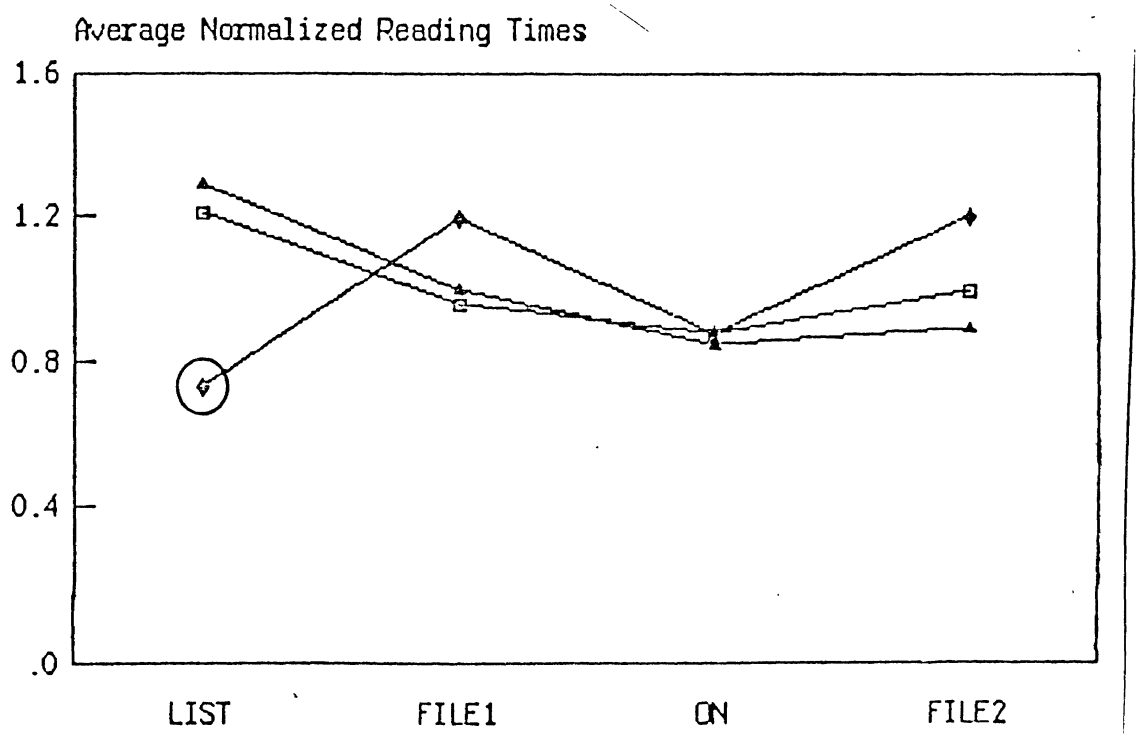

Figure 4. Example "LIST" commands.

As illustrated in the graph, reading times for words varied, however, similar reading time patterns were found for each instance of each command type, e.g., the reading time pattern for each "LIST" command resembled the reading time pattern for other "LIST" commands. The primary difference between similar commands occurred for the average normalized reading times of the verb. The ANRT for the first verb in a command group was smaller than the ANRT's for those verbs occurring later in a command group. This difference in command verb reading times holds

true for all except one command. It occurs because the subject has no information to recode at the start of a recall task and therefore does not spend any time performing recoding.

To determine whether the reading time differences shown graphically were statistically significant we constructed a linear regression model with the average normalized reading time as the dependent variable. The independent variables represented (1) the boundary conditions discussed in our hypotheses and (2) word characteristics which are known to affect reading times such as word length. These variables are shown in Figure 5 and are discussed in the following paragraphs.

Boundary features:

- Before major boundary
- After major boundary
- Before minor boundary
- After minor boundary

Position features:

- Command position
- Command group position
- Session position

Word features:

- Number of syllables
- Known parameter
- Second time seen

Figure 5. Listing of variables used to describe reading time variation.

The boundary variables indicate whether a word is located either before or after our predicted chunk boundaries. The reader will remember that we predicted two types of boundaries: minor boundaries within a command and major boundaries delimiting the command itself. We indicated that minor boundaries would occur between an optional word and a required word. To

denote the boundary, the word viewing times are coded as preceding or following a minor boundary with each condition serving as a separate independent variable.

The position features describe where the command word is located within a command, within a command group, and within a session. For example in the previous figure, the word "PROG" is the 4th word in the "RUN" command, the 6th word in the "CREATE-RUN" multiple command group, and the 145th word in the session (although this is not shown in our example). We encode the single command and command group positions as controls, reasoning that the length of the serial command list the subject is holding in memory will adversely affect how easy it is to add another item. We expect the opposite effect to occur for the session position where the subjects' storing of command words in memory is expected to take less time because of practice effects.

The word features describe the attributes of a command word which also affect reading time. The features are length (measured in number of syllables), whether the parameter is from a closed set of parameter selections, and whether the parameter has previously occurred in the command group. An example coding of these features is illustrated in Table 1 for the command sequence shown.

CREATE OUTFILE
RUN *FTN SCARDS=PROGRAM SPRINT=OUTFILE

Word	Syllable Length	Restricted Parameter	Repeat Occurrence
CREATE	2	no	no
OUTFILE	2	no	no
RUN	1	no	no
*FTN	4	no	no
SCARDS=	4	no	no
PROGRAM	2	no	no
SPRINT=	4	no	no
OUTFILE	2	no	yes

Table 1. An example of the word features coding. For the above command sequence the table shows the values coded for word length, limitations on parameter selection and the re-occurrence of the word in the command group.

We carried out a FORWARD and a BACKWARD stepwise regression on the independent variables arriving at the same contribution of variables in each case. The results are presented in Table 2.

R - Squared = .38 N = 230 F-STAT SIGNIFICANCE LEVEL < .0001

Variable	Regression Coefficient	Significance
CONSTANT	0.73	< .001
BEFORE MAJOR BOUNDARY	0.10	.003
AFTER MAJOR BOUNDARY	0.22	< .001
BEFORE MINOR BOUNDARY	0.12	< .001
AFTER MINOR BOUNDARY	0.06	.056
RESTRICTED PARAMETER	0.08	.001
SYLLABLE LENGTH	0.04	< .001
COMMAND POSITION	0.05	.027
GROUP POSITION	0.05	.002

Table 2. Results of regression analysis.

We first discuss the results of the control variables included in the regression and follow with a section of the paper which discusses the boundary variables. The "restricted parameter" variable is significant and in the predicted direction, i.e., the reading time is lessened by the restricted set of choices. The "syllable length" variable is in the expected direction. Its positive value indicates that the amount of time spent viewing each word increases with an increase in the number of syllables in the word.

Both the command position and the command group position are shown to be positively correlated with their distribution from the start of the command and command group respectively. The session position had no effect, that is the subjects did not exhibit serial list learning improvements across the experimental session. Therefore, the coefficient for the session position is not shown in Table 2.

We now turn to a discussion of the boundary variables and the research support for our hypotheses.

4. Discussion of Results

The conclusion of the previous section presented a linear regression which confirmed the significance of the boundary variables we suggested as being indicative of chunking behavior. This section will discuss these results in greater detail. The section is divided into three parts. These parts will discuss the relation of the regression results to our three reading time hypotheses. We briefly restate our three hypotheses below for reader reference.

H1: Viewing times for words preceding minor boundaries will be larger than the average viewing time for words in the command group

H2a: Viewing times for words following command boundaries will be larger than the average viewing time for words within a command group when the number and type of command parameters commonly used is variable.

H2b: Viewing times for words preceding command boundaries will be larger than the average viewing times for words within a command group when the same command parameters are always used.

H3: Viewing times for major boundary words will be longer than viewing times for minor boundary words.

4.1 Support for Hypothesis 1

Our first hypothesis predicted that subjects would pause for recoding at places in the command sequence where the subsequent parameter was optional. To test for this behavior, we coded words to indicate their occurrence both before and after such a boundary. The regression results show that words before a minor boundary give a significant positive contribution to reading time (beta = 0.12 and $p < .001$). Hypothesis 1 is therefore supported.

We were initially surprised that the coefficient for the variable "after minor boundary" was negative. We believe, in hindsight, that there is a well-founded reason for this coefficient value. First, the variable reflects, in many instances, the keywords of commands. These keywords are expected when the subject sees the command verb. Collins and Loftus' (1975) spreading activation theory of memory puts forth the idea that associated words are activated whenever one of the words in a group of related words is viewed. This would reduce the recognition time for such words and thus, decrease their reading time.

We gain further support for the spreading activation argument when we consider the variable "predicted parameter". This variable indicates those parameters which are from a restricted set of arguments for a keyword. The value of the coefficient is -.08 and significant indicating that the subject spends less time viewing these words.

4.2 Support for Hypotheses 2a and 2b

Our second set of hypotheses predicted that subjects would pause for recoding at the end of each entire command, but noted that the end of a command was not as distinguishable as the boundary distinguishing the command substructure. For many commands, the end could be determined only by seeing the start of the next command. Therefore, we predicted that the pause for recoding would occur both before and after the command boundary. This is borne out by our regression analysis where the coefficient for words coded to occur following a major boundary is positive and significant (beta = 0.22, $p < .001$) and also positive and significant for words

preceding a major boundary (beta = 0.10, p = .003) The coefficient for words preceding a major boundary contributes less because. only a few of the command boundaries in our stimulus are recognizable at the last word of the command. Hypotheses 2a and 2b are therefore supported.

4.3 Support for Hypothesis 3

The regression coefficients for major boundaries also show a greater contribution to overall reading time than those for minor boundaries (beta = 0.22 versus beta = 0.12). An F test comparing these two reading times indicates that the major boundary reading times took significantly longer ($p < .05$) as is predicted by Hypothesis 3.

Our reading time predictions are based on the theory that subjects will pause to recode information presented on a recall task at those boundaries that correspond to internal organization boundaries. Our hypotheses are supported by the experimental data and thus, we believe that we have support for our theories on how users mentally organize command languages.

One difficulty in our study is the correlation of our boundary variables with other features inherent in the command language. Almost all words preceding minor boundaries are variables which are not as familiar to the subject as the command words. This may have increased viewing time for these words. Similarly, all words following major boundaries were verbs. Verbs may take longer to process or we may be measuring the additional time it takes to recognize a new command sequence.

Further analysis of the data to pull out those values where the collinearity did not occur provides no support for the alternate explanation for the minor boundary results. For example, the time spent viewing words from the restricted set of parameters was higher than the time spent viewing words on average, even though the words are known in advance to the subjects. Furthermore, the very low reading times for the first word in a command group do not support the

alternate explanation for the results occurring at major boundaries. We believe that an examination of a command language not having these multiply correlated features at boundaries could provide stronger evidence for our knowledge organization predications. We expect such a language to be much more difficult to use.

5. Recommendations for Command Language Design

In section 2 we gave evidence from the research literature which supports the claim that disruption of a person's knowledge organization adversely affects performance. This implies that an online system should be designed to not disrupt the structure learned by the user. Any aspect of a system which requests a user to specify a subpart of a command, such as a parameter value, should take into account where the mental boundaries of that subpart are located.

Consider the following example. When entering in a command which has keyword and keyword parameters, if the user errs in entering in the correct name of the parameter, the system should allow the user to enter in both the keyword and the keyword parameter if desired. Many error correcting procedures in this situation request only the keyword parameter. This results in user errors (Floyd, 1984).

A second aspect of design is the determination by the designer of when to interrupt a user given that an error is made. Shneiderman (1982) quotes a study by Segal (1975) and states that Segal's results indicate that the error message should be issued immediately. We argue against this. The application presented in Segal's work is different from that which occurs in command language entry. In command languages, the user learns the commands and, as this study indicates, organizes the commands in a systematic fashion in memory. We suggest that in Segal's task the subjects did not have an opportunity to build such an internal structure and therefore could be

interrupted at any point. The memory organization we found, coupled with the literature on reduced performance effects in psycholinguistic tasks, indicates that immediate disruption may be incorrect for the practiced user.

Our results also relate to a third aspect in the design of command languages, that of building command languages for effective learning and later usage. We have shown how the use of optional and required parameters affects the internal organization of items in a user's memory. Such an internal structure could be too long or too short for efficent retrieval or learning of commands (Schneider, 1982). Command languages that are being designed could be examined for their positioning of optional and required parameters in order to produce the most useful internal organization for the user. This suggestion is only hypothesized, for we do not know what constitutes a too long or too short internal mental organization.

In summary, we have shown that users do build an internal knowledge organization for a command language based on what we call the level of interaction of the system with the language. We have combined these results with others' research on knowledge organization to make some recommendations for the design of interactive sessions, namely when and how to interrupt command input.

6. Bibliography

Aaronson, D. and Scarborough H. Performance Theories for Sentence Coding: Some Quantitative Models. Journal of Verbal Learning and Verbal Behavior, 16, 277-303, 1977.

Aaronson, D. and Scarborough, H. Performance Theories for Sentence Coding: Some Quantitative Evidence. Journal of Experimental Psychology: Human Perception and Performance, 2, 1, 56-70, 1976.

Card, S. K., Moran, T. P. and Newell, A. The Psychology of Human-Computer Interaction. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1983.

Chase, W.G. and Simon, H. A. Perception in Chess. Cognitive Psychology, 4, 55-91, 1973.

Collins, A. M. and Loftus, E. F. A Spreading-Activation Theory of Semantic Memory. Psychological Review, 82, 407-428, 1975.

Floyd, B. D. Dissertation Proposal: Knowledge Organization for Command Languages. Graduate School of Business Administration, University of Michigan, Ann Arbor, Michigan, 1984.

Graf, R. and Torrey, J.W. Perception of Phrase Structure in Written Language. American Psychological Association Convention Proceedings, 83-84, 1966.

Larkin, J., McDermott, J., Simon, D. P. and Simon, H. A. Expert and Novice Performance in Solving Physics Problems. Science, 208, 1335-1342, 1980.

Maclay, H. and Osgood, C.E. Hesitation Phenomena in Spontaneous Speech. Word, 15, 19-44, 1959.

Norman, D.A. Categorization of Action Slips. Psychological Review, 88, 1, 1-15, 1981.

Oliver, W. L. and Ericsson, K. A. Actor's Memory for their Parts. Presented at the 24th Annual Meeting of the Psychonomic Society, San Diego, California, November 17-19, 1983.

Olson, J.S. Reitman. Personal communication based on data gathered in Bell Laboratories study, 1983.

Reitman, J.S. Skilled Perception in GO: Deducing Memory Structures from Inter-Response Times. Cognitive Psychology, 8, 336-356, 1976.

Schneider, M. L. Ergonomic Considerations in the Design of Command Languages. NYU Symposium on User Interfaces, New York University, May 26-28, 1982.

Segal, B. Z. Effects of Method of Error Interruption on Student Performance at Interactive Terminals. Department of Computer Science Technical Report UIUCDCS-R-75-727, University of Illinois, Champagne-Urbana, Illinois, May 1975.

Shneiderman, B. The Future of Interactive Systems and the Emergence of Direct Manipulation. NYU Symposium on User Interfaces, New York University, May 26-28, 1982.

Shneiderman, B. Software Psychology: Human Factors in Computer and Information Systems. Little, Brown and Co., Boston, MA, 1980.

Simon, H.A. How Big is a Chunk? Science, 183, 482-488, 1974.

Simon, H.A. and Barenfeld, M. Information-processing Analysis of Perceptual Processes in Problem Solving. Psychological Review, 76, 473-483, 1976.