SGR 4-SEP-75 18:41 26386

File for Practice in Editing

See SGR for an edited hardcopy

File for Practice in Editing

Issues in the Design of the NLS User Interface

By Richard W Watson 1

INTRODUCTION 1a

The user interface has two sides: the input side by which the user inputs information, indicating by various conventions and controls what he wishes accomplished; and the output side by which the machine provides feedback and other assistance to the user in command specification, and provides various forms of information portrayal. Man has many motor and other capabilities that could be the basis for input and command specifications; similarly he has his full range of senses that could be targets for system output. 1a1

To date, computer information systems make use of only a few motor and sensory capabilities in their man-machine dialog. An important area of research involves exploring the advantages to be gained and the techniques to be used to extend this range. There is interesting research going on in areas of speech, eye movement, brain wave control, hand written script, and video graphics that will undoubtedly be integrated into the truly multimedia systems to be built in the near future. 1a2

We call the user's collection of input-output equipment and arrangement of work tables and work space, the workstation. At the present time, input centers around various types of keyboard devices: standard typewriter-type, function button, keyset (chord), and graphical pointing devices: mouse, electronic pen-tablet, light pen, joystick. The dominant output means are printers and displays of varying capabilities. 1a3

The present NLS user interface has been developed around this equipment, although many of the principles used in its design can be easily extended for use with other media [3]. The prime motivation for the use of the mouse for pointing and two keyboards, (standard typewriter-like and keyset), as the input devices for the display version NLS 7 (DNLS), are described in references [2][3]. NLS can also be used from typewriter terminals (TNLS). In this chapter, we concentrate on describing some of the motivations behind the design of the NLS command language and the forms of information portrayed to assist the user in command specifications. Forms of general NLS information portrayal are described in reference [1]. 1a4

The NLS is a prototype collection of tools in a growing workshop of tools and services to aid knowledge work [1][4], and we expect the number of tools and vocabulary that controls their use to grow. We further expect that the use of such a

workshop will spread throughout those occupations involved with information in various forms and that there will be infrequent and casual users of such systems, along with many people who will spend large fractions of their day using such workshops. Another goal is to match the speed of system responsiveness to the natural speed and flow of man's thought processes. It is from these basic expectations that our user interface work has developed. The sections below enumerate several assumptions and areas of concern around which the NLS user interface has developed to date. A key point to mention is that we do not consider the NLS user interface a static, finished product. It will change, based on analysis of usage experience, and the technology and media available. 1a5

HIGH LEVEL ASSUMPTIONS UNDERLYING THE DESIGN OF THE NLS USER INTERFACE 1b

First we describe a few high-level assumptions that affect the user interface design and then discuss some of the lower level issues and the specific techniques used to deal with them. 1b1

1) Coordinated Set Of User Interface Principles 1b1a

There will be a common command interaction discipline, over the many application areas in the workshop, that shapes user interface features, such as the language, control conventions, methods for obtaining help, and computer-aided training. 1b1a1

This commonality has two main implications. One, it means that while each domain within the core workshop area or within a specialized application system may have a vocabulary unique to its area, this vocabulary will be used within language and control structures common throughout the workshop system. A user will learn to use additional functions by increasing vocabulary, not by having to learn separate "foreign" languages. Two, when in trouble, he will invoke help or tutorial functions in a standard way. 1b1a2

2) Grades Of User Proficiency 1b1b

A once-in-a-while user with a minimum of learning will want to be able to get at least a few straightforward things done. In fact, even an expert user in one domain will be a novice in others. Users will be clerical workers, information specialists, executives, engineers, and others. Attention to novice-oriented, and tutorial help features is required. 1b1b1

Users also want and deserve the reward of increased proficiency and capability from improvements in their skills and knowledge, and in their conceptual orientation to the problem domain and to their workshop's system of tools, methods, conventions, etc. "Advanced vocabularies", short concise control notation and conventions in every special domain will be important and unavoidable. 1b1b2

A corollary feature is that workers in the rapidly evolving augmented workshops should be involved continuously with testing and training in order that their skills and knowledge may most effectively harness available tools and methodology. 1b1b3

3) Ease Of Communication Between Subsets And Addition Of Workshop Domains 1b1c

One cannot predict which domains or application systems within the workshop will want to communicate in various sequences with which others, or what operations will be needed in the future. Thus, results must be easily communicated from one set of operations to another, and it should be easy to add or interface new domains to the workshop. A corollary is that the total workshop may contain a very large number of tools and services. Some users may have access to only a subset of its capabilities while others will have access to many or all capabilities. 1b1c1

As described below, we expect the workshop to be embedded in a computer network and thus communication between tools and between users must take place across both process and host boundaries according to well specified conventions and protocols [5][6]. 1b1c2

4) User Programming Capability Or User Interface Extensibility 1b1d

There will never be enough professional programmers and system developers to build or interface all the tools that users may need for their work. Therefore, it must be possible, with various levels of ease, for users to add or interface new tools, and extend the language to meet their needs. They should be able to do this in either a variety of programming languages with which they may have training, or in the basic user-level language of the workshop itself. 1b1d1

5) Range Of Workstations And Symbol Representations 1b1e

The range of work stations available to the user will increase in scope and capability. These work stations will support text with large, open-ended character sets, pictures, voice, mathematical notation, tables, numbers, and other forms of knowledge. Even small portable hand-held consoles will be available. The multiplicity of possible terminals indeed raises the question of whether a consistent set of control and portrayal conventions is possible. 1b1e1

As hardware decreases in cost, more and more capabilities will be placed in the work station both in the form of user interface aids and facilities, and in the form of frequently used tools. 1b1e2

6) Distributed Nature Of The User Interface Processes 1b1f

The collection of facilities to support interfaces with the system of tools can be conceived of as a single service as seen by the user. These facilities may all reside in a processor in the work station or be distributed in two or more processors, depending on the level of their sophistication and state of the art with respect to cost, hardware capability, and so forth. 1b1f1

7) Embedded In a Computer Network 1b1g

The computer-based tools of a knowledge workshop will be provided in the environment of a computer network, such as the ARPANET [7]. For instance, the core functions will consist of a network of cooperating processors performing special functions, such as editing, publishing, exchanging documents and messages, data management, and so forth. Less commonly used, but important functions, might exist on a single machine. The total computer-assisted workshop will be based on many geographically separate systems. 1b1g1

Once there is a "digital-packet transportation system," it becomes possible for the individual user to reach out through his processor to other people and other services scattered throughout a "community". The "labor marketplace" where he transacts his knowledge work will be literally independent of geographical location. 1b1g2

Specialty application systems will exist in the way that specialty shops and services now do--and for the same

reasons, when it is easy to transport the material and negotiate the service transactions, one group of people will find that specialization can improve their cost/effectiveness, and that there is a large enough market within reach to support them. And, in the network-coupled computer-resource marketplace, there will be a growth of specialty shops, such as application systems specially tailored for particular types of analyses, or for checking through text for spelling errors, or for doing the text-graphic document typography in a special area of technical portrayal, and so on. There will be brokers, wholesalers, middle men, and retailers. 1b1g3

The key point to emphasize is that even when hardware costs decrease to the point where a user can perform 90% of his work using tools and information that operate in the processor in his work station, he will want to have access to a computer network to:

a) Communicate in various forms with others
b) Access very large or special data bases
c) Access special tools that run elsewhere 1b1g4

8) Problem Orientation Of The Command Language And Tolerance For Ambiguity 1b1h

The user has a task that he wishes performed by the system. Depending on the nature of the task and operations available to him on the system, he may be able to express what he wants accomplished in a single "statement" or command to the machine, or it may require a series of commands. 1b1h1

One of the goals of the designers of the command language and system is to understand the nature of the user's application domain so that the user can express his needs with words that are similar to his natural problem solving vocabulary and language forms. The machine should then break down the request into smaller steps as required. 1b1h2

If there is ambiguity in the user's command, the machine should recognize it, if possible, and prompt appropriately for clarification. There is still much research and development required to fully meet this goal. 1b1h3

Many people hope to allow novice users or users in certain applications to use natural language in making

statements to the machine. This capability will require models of the user and task domains for understanding. 1b1h4

Even when systems are able to interpret commands given in natural language, the precision and usage efficiency of appropriate artificial languages will make the latter's continued use preferable, especially for skilled users. 1b1h5

Given the above general considerations as background, we can move on to examine features of the NLS user interface in more detail. 1b1i

MORE DETAILED DISCUSSION OF THE NLS USER INTERFACE 1c

A command language must allow unambiguous specification of what the user wishes accomplished. The operation to be performed, and the entities or information items (arguments) to be acted upon, or used to determine what is to be acted upon, must be specified. These can be specified in a variety of ways: by typing them in in full or in some form of abbreviation, by pointing at them on a screen, by pronominal reference, or by use of default values where appropriate. The order of their specification, the syntax or grammar of the language, can have various forms. For example, operational keywords can be specified, followed by the arguments, or vice versa. Arguments can be in fixed positions or explicitly named and occur in any order. Some arguments or keywords can be optional and require special characters to indicate their presence. Arguments or keywords can have defaulted values under certain conditions. Pronominal references can be allowed to refer to previous occurences. Arguments may be given types by the system and language designer for more extensive error checking and feedback. 1c1

Arguments and keywords can be specified by complete or partial typein (there are a variety of forms of command recognition that are discussed later) or designated by pointing to representations on a display or by use of specially coded function keys. Or, the machine may ask questions and the user just fill in the blanks. 1c2

Depending on the characteristics of the computer and communications system, it may or may not be possible to provide command word or keyword completion, prompts or other feedback, argument checking, default value fill in, and so forth, during the command specifications. 1c3

For example, in line-at-a-time, half-duplex systems, the user usually must complete the entire specification of the command

before transmission to the system, while in character-at-a-time, full-duplex systems, the system can react to each character received and provide more extensive aids to the user during command specification. 1c4

The above discussion outlines just a few of the many choices available to the language designer. As the purpose of this section is not to be a complete tutorial on all possible choices available and their advantages and disadvantages, the following discussion only gives main NLS command language features and the motivation for their adoption. 1c5

THE NLS COMMAND LANGUAGE 1d

The NLS command language generally has the following form, where angle brackets group meta symbols: 1d1

<operation specification> <operand specification> <command completion> 1d1a

The fields in a command are of a fixed order, although some commands have optional fields that can be specifically requested. Other fields can have a system-supplied default value. Because NLS operates from a character-at-a-time, full-duplex system, several levels of help are available, as described later, for giving cues and prompts, explicitly listing options or syntax, and giving full documentation on what the system expects next during command specification. It was not felt that much would be gained for novice users by allowing fields to be specified in any order by using explicit field names. Novice users do not need to be aware of optional fields. 1d1b

As much as possible NLS makes the operational specification of the form verb-noun followed by arguments and possibly other keywords. We have also tried to maximize the fullness of the verb-noun matrix. 1d2

This approach seemed to be natural, and follows normal English imperative forms to aid learning. The choice of verb-noun form seemed to fall out naturally when considering such important areas as editing. A given verb, such as DELETE, can naturally be applied to many entities, such as statement (a paragraph, title, equation), character, number, text, file etc. Learning is easier if the user can form a model of how the system works that can be consistently applied. In this case, a user can learn n verbs and m nouns and understand that generally, if it is meaningful, they can

be used in pairs. Having learned n+m vocabulary terms, he can apply them in the form of n x m commands. 1d2a

We have tried to pick command keywords that have normal usage related to the operation described. A synonym capability would be easy to implement. 1d2b

Four forms of command keyword recognition are provided to enable the user to choose the one most appropriate to his terminal type, system response, previous system experience, and present NLS experience level. We have worked to pick an operational vocabulary for the present system that guarantees keywords to be unique in a maximum of three characters: 1d3

1) A single-character mode allowing high-speed single-character recognition of the most commonly used commands; less commonly used commands require an escape character followed by enough characters for unique recognition: With large and expanding command sets one cannot choose keywords with mnemonic value and guarantee uniqueness with the first character. This mode is generally preferred by experienced users because of the simplicity and speed with which frequently used operations can be expressed. We find that experienced users are very concerned that commands be formed with the minimum number of input operations, and that commands have the richness needed to specify adjective or adverb type operations as needed. There is thus some conflict in certain commands between these goals for the experienced user and the need for command simplicity for the novice. 1d3a

2) A demand mode requiring a right delimiter to initiate recognition: This has proved to be popular for new users of typewriter terminals, particularly those with experience using the TENEX operating system. Modes C and D have not turned out to be heavily used. 1d3b

3) An anticipatory mode requiring the user to type enough characters until the command is uniquely specified; the system then automatically fills in the remainder. 1d3c

4) A fixed mode that guarantees recognition on entry of three characters. 1d3d

Given the implementation approach outlined later, it is quite easy to add other recognition modes, such as allowing the user to choose keywords from a menu displayed on the screen. However, experiments have shown that the time it takes to point at some item on the screen is equivalent to

several keystrokes and thus would be disadvantageous to skilled users, although possibly of value to novices [2][3]. 1d3e

Operand argument specification is contained in a number of fields that are variable with the type of command. All commands of a similar type have had the order of the operands made as consistent and as natural (relative to normal English usage) as possible. Infrequently used operand fields are optional and novice users need not be aware of their existence. 1d4

Related to argument specification is the problem of choosing argument delimiters. One can recognize the following delimiting functions. 1d4a

1) Delimiting command words
2) Delimiting arguments
3) Delimiting optional arguments, selection type, or command word fields
4) Delimiting commands
5) Selecting arguments off a display screen, and confirming the selections 1d4a1

One could choose separate characters (codes) to represent each of these functions. To do so seemed to us to add an unnecessary complication for the user and so, except for using a special character to indicate an optional argument, selection type, or command word, a single code is used for the other function in NLS. We call this code "Command Accept" (CA) even though it is used for other purposes as well. The system allows the user to define which keyboard character is to serve this function if he finds the system default to be inconvenient. One of the buttons on the mouse also serves this function. 1d4b

Arguments can be typed in, defaulted where appropriate, or specified by pointing to appropriate entities on the display screen. 1d4c

There are three flavors of command completion. 1d5

1) Completion of the command indicating execute the command and return to the base state to await input of the next command: The default indication for this form is one of the buttons on the mouse in DNLS, which is translated into a control character, or CR in TNLS. The use of CR in TNLS is quite natural and generally does not conflict with textual input as most text in NLS is typed in without explicit CRs and is appropriately formatted by the system for various output devices. If the TNLS user wishes to input an

explicit CR in his text file, he must precede it with an escape character. If he has need to enter many CRs in his text string, he can redefine the completion character, Command Accept, to be some other character. 1d5a

2) Completion of the command and return to an appropriate point for quick repetition of the command. Repetition mode continues until explicitly commanded to delete out of it. This mode is very useful when a delete or other operation is repeated several times. 1d5b

3) Completion of the command and entry to insert-statement mode for addition of new paragraphs or other text statements: This mode is like command repeat above except that it always takes you to the insert command. It is used frequently when one adds, replaces, or moves text, and then wants to follow it with new statements. It speeds text input when inserting sequences of paragraphs. 1d5c

The system is to be used from a variety of terminal types, including both typewriter-type terminals and displays. The two-dimensional displays are to be the preferred work station types whenever a design decision must be made between language forms possibly favoring one type or the other. 1d6

It was decided to make the command language syntax for the typewriter (TNLS) version and the display (DNLS) version as close as possible, except where the difference between the one-dimensional and two-dimensional media clearly prohibits this or would seriously limit one or the other version. This decision was made to allow people working in environments consisting of both typewriter and display terminals to be able to move back and forth with ease. 1d6a

The system has been organized into clearly defined subsystems with uniform rules for their entry and exit. Any subsystem can be entered from any other, either to "execute" a single command with automatic return or to perform a chain of commands. The user can return, either to a specifically named subsystem in the path of subsystems traversed or enter a new subsystem. The issue of how to group commands into subsystems has to do with training and patterns of use rather than system constraints. It relates to learnability and, to some extent ease of command specification using single characters, and to "knowing where you are" in a command or operational space. 1d7

One could construct a system where all commands were in a single subsystem. Study of the command set of a large system particularly conceived of as a set of tools shows

that operations tend to group together such that to perform a given task, such as sending a message or calculating a budget, generally require several related suboperations. Certain operations, such as moving in information space or seeking help, tend to be used as suboperations of many or all tasks. This latter observation has led to "universal" commands available from within any subsystems. One can also imagine certain commands to be needed frequently in just two or more subsystems and thus implemented in each subsystem having the need. There are now no instances of this case in NLS. The ability to execute a single command in another subsystem with automatic return has been very useful. 1d7a

Provision has been made for user-controllable options on prompting, feedback, and other parameters whenever it seemed a single option, might not be appropriate to some significant class of users. 1d8

A mechanism is implemented that enables the user, or someone acting in his behalf, to create a file stating what options he wants to run. The system automatically sets his options when he enters. This facility can also be used with small extensions to subset commands. This user option capability, when coupled with the ease by which the user interface can be redefined using the Control Meta Language described below, makes possible tailoring the user interface to specific users or groups of users. 1d8a

All operations that have a natural inverse command have been given one. (NLS still does not have an "undo" facility.) A general undo/redo facility has a number of technical difficulties and its value can be questioned. However, the ability to undo or redo the last one, two, or three commands would clearly be useful. 1d9

User Programming: As indicated earlier the ability of the user to extend the system himself is important. There is a tradeoff between ease of extension specification and operational efficiency. In providing such a facility one does not have to be deeply concerned with efficiency if the task handled by the extension is performed infrequently. If the operation is performed frequently, then it should probably be inserted as a system feature and implemented efficiently by professionals. This area is ripe for much additional development. The extensions must be specified in some language to indicate what sequence of events is to take place, what arguments to collect, and so forth, when a given user action is performed. 1d10

NLS now offers two forms of extensibility. The first allows

users with some basic programming knowledge to write programs in the Algol like L10 language in which the system is implemented, calling on NLS system primitives as needed. They can use the Control Meta Language to specify a user interface if desired. These programs can be installed by the user as part of his default subsystems, loaded as subsystems as needed, or used as content analyzer patterns [8]. 1d10a

The user can also write sequences of NLS commands and have these sequences executed at will. A specific sequence of commands can be automatically invoked when the user first enters NLS. 1d10b

HELP, STATUS, AND PORTRAYAL FACILITIES 1e

ORGANIZATION of the TERMINAL DISPLAY AREA

The NLS display screen is organized into windows as described in some detail in [9] These windows are arbitrary rectangles. Windows can be displayed essentially all the time or overlayed with others. Windows can grow dynamically. Some windows are allocated and displayed or not displayed under system control for status and feedback information. Others can be created and manipulated by the user for display in his information space. With typewriter terminals, one does not have this two-dimensional random display capability and while the same information can be given to him, less can be given automatically or must be given in an altered form. Let us now consider each of the information spaces and the type of feedback, help, and other status information available to the user. 1e1

1) Information space

The present NLS information space is hierarchically organized. A user has a directory or directories within which there are files. A file can contain notes on many subjects stored under various headings, his mail, or single documents. Files in turn are hierarchically organized as a tree of information nodes (now text strings but soon to be generalized to include illustrations and other entities). 1e1a

Files can contain cross citations to specific points within other files or the same files, thus creating networks. NLS has appropriate commands for moving within and between files and for obtaining a display of the path over which one has traveled and commands for backtracking along this path [3]. 1e1b

Display screens have a limited number of lines within which to display information, and typewriters, even at 30

chars/sec or higher, cannot quickly and easily print out large documents. Also, the user often wants to see a summary or overview of a document or have it formatted in special ways to aid his understanding. To meet this need for easy control of information portrayal, NLS has a concept called "view specification". The user can change his "view" within the commands for moving in information space or by separate command. So that he can be reminded of his current view, the most commonly used view parameters are fed back to him in a small window in the upper right hand corner of the screen. When he is at a point in a command where it is permissible to change views, this fact is fedback both by prompt (if prompts are turned on) and by enlarging the characters in the view-feedback window. For more discussion on moving, viewing, and portrayal in NLS see [3][6]. 1e1c

2) Subsystem or tool space
NLS is viewed as a collection of tools (subsystems) that can be used cooperatively or stand alone. Each subsystem contains a number of logically related commands and has a name, such as Base (the collection of editing and file manipulating commands), Calculator, etc. All the tools work on information in the same file structure and the user can move from one tool to another, or execute commands on a single command basis in any tool from any other tool, as mentioned earlier. The user can receive a display of subsystems available to him or an ordered list of the subsystems in which he has previously been. 1e1d

The current subsystem within which he is operating is fed back in a small window in the upper left-hand corner of the screen in DNLS and as a four-character prompt in TNLS. 1e1d1

3) Command syntax space
There are several levels of feedback and Help available to the user in formulating a command to the system (14,). Each is described below. The Help data base clearly is also generally useful for understanding the system as a whole. 1e1e

a) Command keyword recognition:
The options here were described earlier and this mode is primarily useful in minimizing keystrokes and in triggering additional feedback. 1e1e1

b) Noise words:
When the system recognizes a keyword or field it generates what we call "noise words" set off in parentheses so the user can distinguish between what he

has input and what the system has added. The noise words aid the user in remembering what to do next. Novice users report that noise words are one of the most useful initial aids. As more experience is gained, the other aids take on more importance. This is an important point to note: users at different levels of experience value different forms of feedback. Usefulness is not only determined by the inherent characteristics of the aids, but also, by how they are implemented. 1e1e2

c) Prompts:
When the user completes the specification of a field in a command, he is prompted with some terse characters indicating the type of thing expected next and the alternatives available to him for how he can specify, select, or address the needed argument. Users can turn prompts off, which some users of TNLS do when they reach a certain level of proficiency, although many highly skilled users always operate with them on. DNLS users tend to always operate with them on because the high speed of the display does not slow down work while providing useful information. Users can also specify terse prompting in which case optional fields are not prompted for.
Beginning users have indicated that prompting is useful, but would like them to be more mnemonic and of word length. 1e1e3

d) Next Options and Syntax:
If the noise words and prompts are not sufficient to jog a user's memory about what options are available to him next, he can strike a ? or a <Control-S>. If he strikes a ?, the system displays, in alphabetical order, all the command keywords that are legitimate for the next field or more extensive information than is available in the prompts for other fields. If he strikes <Control-S>, the system prints out the syntax of the command from his present position to the end of the command. The ? facility is extensively used and is very useful in refreshing one's memory about infrequently used commands or new commands for a user with only a basic knowledge of command system concepts and vocabulary. The <Control-S> feature does not seem to be extensively used at present and may indicate that the ? facility is sufficient. 1e1e4

e) Help Data Base:
If the above facilities are not sufficient because of uncertainty about a basic concept or vocabulary word or the user wishes more information about the effects or use

of a command, he can enter the the Help tool. Entry can be from the basic command level or from any point during command specification. In the latter case, the system utilizes the information input at this point to take the user to an initial point that describes the command and field where he is at. (15) 1e1e5

Once in the Help Data Base, a simple set of command conventions and the organization of the data base allow the user to easily move to reference related subjects or move to new subjects or back up to higher level descriptions (15). 1e1e6

f) Active Tutorial Help:
The next level of Help facility would be an active tutorial facility. We have not yet implemented such a facility but can see its value. An example of such a facility is the work going on at BBN on the NLS-Scholar system [10]. 1e1e7

ERROR MESSAGES AND RECOVERY 1e1f

Error messages indicating an incorrectly spelled file name or improperly specified entity are fed back to the user in a window at the top of the screen. The user is left at an appropriate point within the command specification or where necessary he must start over again to respecify the command. The text of error messages is important and should be as specific to the problem as possible. This has implications within the system design for trapping error conditions as early as possible and determining the appropriate message for the specific error and total context of the user. While we have made progress in this area, there is much more that could be done to meet the need stated above. 1e1f1

There are now no automatic error correction mechanisms built into the system, such as spelling correction or "Do What I Mean" type facilities. These would probably be useful to add when resources permit. 1e1f2

EDITING AND BACKUP DURING COMMAND SPECIFICATION 1e1g

The user can perform certain simple editing and backup operations during command specification. At any point during command specification he can command delete, which will take him back to the basic command level. This is useful if he gets confused and wants to return to a known

state or changes his mind about which command to perform next. 1e1g1

The user can delete the last character input or last selection made on the screen with another character or button push on the mouse. He can repeat this process and continue the incremental backup process to the basic command state. 1e1g2

The user can delete the last word input, or the field specified to date, or the field specified with another character or button push on the mouse. He can also repeat this process backwards to the basic command state as well. 1e1g3

IMPLEMENTATION 1e1h

The mechanisms and data bases needed to implement the user interface have been modularized and isolated. This "Frontend" can run on a separate computer, such as a mini-computer close to the user, and communicate with the basic tool information processing routines ("Backend") over a communication network. The Frontend consists of terminal handling capabilities [9], a command language interpreter (2a1), and two data bases, a Grammar representing the language syntax and noise words; and a User Profile indicating how the user wants various parameters set for him, such as his prompt and command recognition modes, keyboard key translations, etc. The Grammar is generated from a high-level description of the user interface written in a language special for this purpose we call Control Meta Language (2a1,). 1e1h1

Given this particular system organization it is very easy to tailor, subset, or modify the user interface for individuals or groups, or to create interfaces for new tools. 1e1h2

Further all the levels of help information, except the Help Data Base, are derived from the Grammar, which guarantees correctness of these levels of documentation as the system changes and is debugged. Various forms of hard copy documentation, such as command summaries, are also derived from the Grammar representation. 1e1h3

The user interface must implement a man/machine dialog. In this section, we discuss issues from machine to man. The discussion centers around the use of displays, with comments on how the problem is dealt with for typewriters. Let us examine

some of the types of information that the user needs in order to keep his bearings. 1e2

There are four main areas or dimensions along which the user needs information to help him a) to know where he has been, b) to know where he is, and c) to know where he can go from here. Clearly the command language and user interface must offer provisions to move in these spaces as well as obtain status. 1e3

1) Information Space
The user needs to know where he is in his information space, and what view or portrayal of the many possible is being displayed to him. Generally he arrived at his present position from previous points and he may want to be able to backtrack to previous points or views as well as to move on. 1e3a

2) Subsystem or Tool Space
In workshops containing many tools and commands, the user needs to know which tool is active and possibly needs to know which ones he was in previously and their order, and which ones he can enter from here. 1e3b

3) Command Syntax Space
During the specifications of a command, the user may need to know what he can or is expected to do next and how to back up to a previous point. 1e3c

4) Information Input Space
During input of information, drawings, tex, etc,, the user needs to have ways to see and possibly modify, in simple ways, information that he is entering. 1e3d

REFERENCES 1f

(8A4) (8A5) (8E1B) (8E1C) Douglas C. Engelbart. A Research Center for Augmenting Human Intellect. Augmentation Research Center, Stanford Research Institute, Menlo Park, California 94025. 68. (3954,) 1f1

(8A4) (8D3E) William E. English. Display-Selection Technique for Text Manipulation. Augmentation Research Center, Stanford Research Institute, Menlo Park, California 94025. MAR-67. (9694,) 1f2

(8A4) (8D3E) Douglas C. Engelbart. Design Considerations for Knowledge Workshop Terminals. Augmentation Research Center, Stanford Research Institute, Menlo Park, California 94025. 14-MAR-73. (14851,) 1f3

(8A5) (8E1C) Douglas C. Engelbart, Richard W. Watson, James C. Norton, The Augmented Knowledge Workshop, Augmentation Research Center, Stanford Research Institute, Menlo Park, California 94025, 1-MAR-73, (14724,) 1f4

(8B1C2) James E. (Jim) White, Version 2 of the Procedure Call Protocol (PCP), Augmentation Research Center, Stanford Research Institute, Menlo Park, California 94025, PCP-COVER.NLS;5,, (24590,) 1f5

(8B1C2) Jonathan B. Postel and James E. (Jim) White, Notes on a Distributed Programming System, Augmentation Research Center, Stanford Research Institute, Menlo Park, California 94025, 21-MAR-75, (25613,) 1f6

(8B1G1) Lawrence G. Roberts and Barry D. Wessler, (University of Utah, Computer Science Department), The ARPA Network, Advanced Research Projects Agency, Information Processing Techniques, Washington, D.C. MAY-71, (7750,) 1f7

(8D10B) No Author, L10 Users' Guide: Content Analyzer, Augmentation Research Center, Stanford Research Institute, Menlo Park, California 94025, L10,NLS;7,, (24426,) 1f8

(8E1) (8E1H1) Charles H. Irby, Display Techniques for Interactive Text Manipulation, Augmentation Research Center, Stanford Research Institute, Menlo Park, California 94025, 15-NOV-73, (20183,) 1f9

(8E1E7) M. C. Gringnetti et. al, An Intelligent Online Assistant and Tutor--NLS Scholar, AFIPS Conference Proceedings, Vol. 44, Anaheim, California, MAY-75, (25054,) 1f10

1f11

File for Practice in Editing

(J26386) 4-SEP-75 18:41;;;; Title: Author(s): Susan Gail Roetter/SGR; Sub-Collections: SRI-ARC; Clerk: SGR; Origin: < ROETTER, EDITSAMPLENODIRS.NLS;3, >, 22-AUG-75 17:58 SGR ;;;; ####;

RLL 4-SEP-75 19:38 26387

VISIT: Al Watson of AMes coming Tuesday 9 Sept

Anyone else interested in seeing AL pleaselet me know Thaks, Rob

RLL 4-SEP-75 19:38 26387

VISIT: Al Watson of AMes coming Tuesday 9 Sept

Al Watson of Ames will be visiting ARC next Tuesday) Sept. He is interested in interfacing scientists with computers. That is, having an editor and compiler appear coherent to the user. In any case, DCE has asked me to host him with Ken V, availbale for backup support in the area of programming support packages within our AKW system. 1

RLL 4-SEP-75 19:38 26387

VISIT: Al Watson of AMes coming Tuesday 9 Sept

(J26387) 4-SEP-75 19:38;;;; Title: Author(s): Robert N. Lieberman/RLL; Distribution: /KEV([ACTION]) ARC-LOG([INFO-ONLY]) SRI-ARC([INFO-ONLY]) ; Sub-Collections: SRI-ARC ARC-LOG; Clerk: RLL;

26387 Distribution

Kirk E. Kelley, N. Dean Meyer, James E. (Jim) White, Douglas C. Engelbart, Martin E. Hardy, J. D. Hopper, Charles H. Irby, Harvey G. Lehtman, James C. Norton, Jeffrey C. Peters, Dirk H. Van Nouhuys, Kenneth E. (Ken) Victor, Richard W. Watson, Don I. Andrews, Kenneth E. (Ken) Victor, James C. Norton, Log Augmentation, David C. Smith, Mary Ann Kellan, Buddie J. Pine, Andy Poggio, David L. Retz, Laura J. Metzger, Karolyn J. Martin, Jan A. Cornish, Larry L. Garlick, Priscilla A. Wold, Pamela K. Allen, Delorse M. Brooks, Beverly Boli, Rita Hysmith, Log Augmentation, Joseph L. Ehardt, Raymond R. Panko, Susan Gail Roetter, Robert Louis Belleville, Rene C. Ochoa, Ann Weinberg, Adrian C. McGinnis, Robert S. Ratner, David S. Maynard, Robert N. Lieberman, Sandy L. Johnson, James H. Bair, Jeanne M. Leavitt, Rodney A. Bondurant, Jeanne M. Beck, Marcia L. Keeney, Elizabeth K. Michael, Jonathan B. Postel, Elizabeth J. Feinler

SGR 4-SEP-75 19:39 26388

Course Outline for AKW Seminar - August 25-29

This is a copy of what was passed out to attendees. It isn't an accurate picture of what actually happened each day. A description of what happened will follow in a course report with comments on how the whole course went.

Course Outline for AKW Seminar
Monday 1

For details of 1-3 below see page 2. 1a

1) Use of the interface devices 1b

special keys on keyboard 1b1

buttons on mouse 1b2

what you see on display screen 1b3

lp lights and reset button 1b4

keyset use 1b5

2) Command syntax and command word alternatives 1c

What prompts are 1c1

How commands are typed 1c2

? 1c3

Glossary - definitions of command words 1c4

Available in hardcopy or online 1c4a

3) Pointing and viewing information 1d

How to point (BUG), BC, BW, and CD 1d1

Jump to BUG with possibly Jump to Origin, Back, Next, and Return 1d2

(AKW document as example) 1d2a

4) Organization of info in NLS 1e

(See Section E, Figure 1 - next to last page for graphic representation of most of the following terms) 1e1

directories, files 1e2

statements, groups 1e3

substatements, branches 1e4

text, word, character 1e5

Build on earlier jumping with additional commands (successor, predecessor, etc.) and viewspecs (See Section M, pp. 1-2 for summary of Jump commands; see Section L for list of Viewspecs) 1e6

Course Outline for AKW Seminar - August 25-29

SPECIAL FUNCTION KEYS 1f

OK 1f1

OK key on keyboard; right button on mouse 1f1a

In general, it indicates to DNLS "I am finished with this," and causes DNLS to accept or do something specified. It may be thought of as a "confirmation" or "OK". 1f1b

It is used to terminate strings of characters you input, to BUG something on the screen, and to give a confirmation anytime the prompts in your command feedback line ask for an OK. 1f1c

BC (Backspace Character) 1f2

BACKSPACE on keyboard; left button on mouse 1f2a

One character of input is deleted each time this is pressed. 1f2b

BW (Backspace Word) 1f3

BACKSPACE WORD on keyboard; left and middle buttons on mouse 1f3a

One word of input is deleted each time this is pressed. 1f3b

CD (Command Delete) 1f4

CMD DEL on keyboard; middle button on mouse 1f4a

This is used to abort a command sentence. 1f4b

PROMPTS 1g

C: COMMAND needed (? for possibilities) 1g1

B: BUG (Point) 1g2

T: TYPE IN something; end with OK 1g3

A: ADDRESS; end with OK 1g4

L: LEVELADJUST (d, u, or OK); end with OK 1g5

V: VIEWSPECS; end with OK 1g6

OK: Confirmation required 1g7

Y/N: Answer required; Y for Yes, N for No 1g8

Any two or more of the above separated by a / indicates a choice. 1g8a

Any of the above surrounded by [] are optional. 1g8b

Course Outline for AKW Seminar - August 25-29

Tuesday 2

1) How to build structured data bases 2a

How to insert statements at different levels 2a1

Use previously input data base as example and build on it 2a1a

Commands Used: Insert Statement with INSRT and levels; review jumping commands 2a1b

Statement names 2a2

(Section K, pp. 6-7 of The Intermediate TNLS Course Outline) 2a2a

Content analysis capability (not details on use) 2a3

(Section O, Part One, pp. 1-13) 2a3a

2) Intra file editing - (Chapter from Final Report for practice) 2b

Text editing 2b1

Structure editing 2b2

Commands Used: Verbs - Insert, Delete, Move, Copy, Replace, Break, Append; Nouns - Word, Character, Text, Statement, Branch, Group 2b2a

3) Multifile editing 2c

Moving and copying cross files by addressing cross files 2c1

(Section K, p. 9 of The TNLS Course Outline: Introduction to Structure and Viewing) 2c1a

Content analysis option 2c2

(Section O, Part One, pp. 1-13) 2c2a

Split Screen 2c3

Commands Used: Insert, Move, and Delete Edge 2c3a

Clipping Viewspecs 2c4

4) Links (examples embedded in text of file being used) 2d

Course Outline for AKW Seminar - August 25-29

(Section K, pp. 9-10 of The TNLS Course Outline: Introduction to Structure and Viewing) 2d1

to other files 2d2

to change or direct view 2d3

with content analysis 2d4

Course Outline for AKW Seminar - August 25-29

Wednesday - Communicating 3

1) Sendmail (Section K, pp, 14-15 of The TNLS Course Outline: Introduction to Structure and Viewing) 3a

Commands from above course adding Forward, Authors, Private 3a1

Read mail by using Jump Link and Jump File Return 3a2

2) Sndmsg (Section K, p, 15 of The TNLS Course Outline: Introduction to Structure and Viewing 3b

Send in Tenex 3b1

Read with mess 3b2

3) Message program (Section I, pp, 2-3) 3c

4) Process branches 3d

Course Outline for AKW Seminar - August 25-29

Thursday	4
1) Output Processor and Directives	4a
Add directives to Final Report Chapter and print in TNLS	4a1
(Section N)	4a2
2) Format, Letter, and Modify programs	4b
(Section I)	4b1
3) Wrap up of content analyzer (review discussions of previous days)	4c
Statement signatures	4c1
(Section O, Part One, pp, 1-13)	4c2
4) TNLS session (Section K - relevant sections of Course Outlines)	4d
Printing	4d1
Addressing	4d2
Useroptions	4d3

Course Outline for AKW Seminar - August 25-29

(J26388) 4-SEP-75 19:39;;;; Title: Author(s): Susan Gail Roetter/SGR; Distribution: /US([INFO-ONLY]) ; Sub-Collections: SRI-ARC US; Clerk: SGR; Origin: < ROETTER, OUTLINE,NLS;5, >, 29-AUG-75 13:24 SGR ;;;; ####;

26388 Distribution
Susan Gail Roetter, Priscilla A. Wold, Jeanne M. Beck, Pamela K. Allen, Rita Hysmith, Sandy L. Johnson,

Weekly Report for last two weeks

The week ending 9/5/75 and the week prior to that 1

The last two weeks have involved continuing work on the projects that were begun at Gunter during my first visit. I have continued the formatting of three volumes of 66-1. One of the volumes is done and the other is having the spaces for figures inserted now. 1a

The Base Tops document that was due 8/29/75 was finished and formated in time to reach the deadline. Apparently, very few of the other groups at Gunter were able to reach this deadline. 1b

The PR group have continued to capture the real property document. They have been creating tables at ISIC and I have been transfering them over to the correct files at office-1. 1c

The last week, I have spent time with Lynne Simms and Pete Lambert, getting them ready to start another rewrite of one of the volumes of 66-1. I have tried to teach them as much as possible from getting the files ready for the writers up through the final editing, proofing and com stages. They will begin this rewrite Monday morning. 1d

I sent the Preface to Com, the proofs came back and it is now ready to be printed. 1e

Gunter for the nxt two weeks: 1f

I will be at Gunter for the next two weeks. We have two more volumes of 66-1 that will be reworked in this time. I will continue to work on the formatting of the volumes and get as much ready to be printed as possible. 1f1

I will work with the PR group in helping them to format their document and get other documents into production. 1f2

There is a demo planned for Col. Bruner (commander of Gunter) next Thursday or Friday. 1f3

I plan to meet with Larry, Lynne and Maj. Hearn to talk more about some long range plans and goals as far as my work and NLs at Gunter. 1f4

I hope not to meet up with George Wallace. 1f5

POOH 5-SEP-75 10:10 26389

Weekly Report for last two weeks

(J26389) 5-SEP-75 10:10;;;; Title: Author(s): Ann Weinberg/POOH;
Distribution: /SRI-ARC([INFO-ONLY]) ; Sub-Collections: SRI-ARC;
Clerk: POOH;

26389 Distribution

Douglas C. Engelbart, Martin E. Hardy, J. D. Hopper, Charles H. Irby, Harvey G. Lehtman, James C. Norton, Jeffrey C. Peters, Dirk H. Van Nouhuys, Kenneth E. (Ken) Victor, Richard W. Watson, Don I. Andrews, David C. Smith, Mary Ann Kellan, Buddie J. Pine, Andy Poggio, David L. Retz, Laura J. Metzger, Karolyn J. Martin, Jan A. Cornish, Larry L. Garlick, Priscilla A. Wold, Pamela K. Allen, Delorse M. Brooks, Beverly Boli, Rita Hysmith, Log Augmentation, Joseph L. Ehardt, Raymond R. Panko, Susan Gail Roetter, Robert Louis Belleville, Rene C. Ochoa, Ann Weinberg, Adrian C. McGinnis, Robert S. Ratner, David S. Maynard, Robert N. Lieberman, Sandy L. Johnson, James H. Bair, Jeanne M. Leavitt, Rodney A. Bondurant, Jeanne M. Beck, Marcia L. Keeney, Elizabeth K. Michael, Jonathan B. Postel, Elizabeth J. Feinler, Kirk E. Kelley, N. Dean Meyer, James E. (Jim) White

DVN 5-SEP-75 12:53 26390

Dialog Concerning Copy Directory

In reply to your message of 3-SEP-75 2222-P DVN Journal: (26373,)
Subject: Dialog Concerning Copy Directory 1

Dirk, I need more facts for Dave Hopper about your problem with the Copy Directory command. What directory were you in and how many files are in that directory? Pam 1a

>>>>>I was logged in and connnceted to vanNouhuys at BBNB, there are now 44 files in my directory and must have been about that number then. It has happened again since.----- 1b

DVN 5-SEP-75 12:53 26390

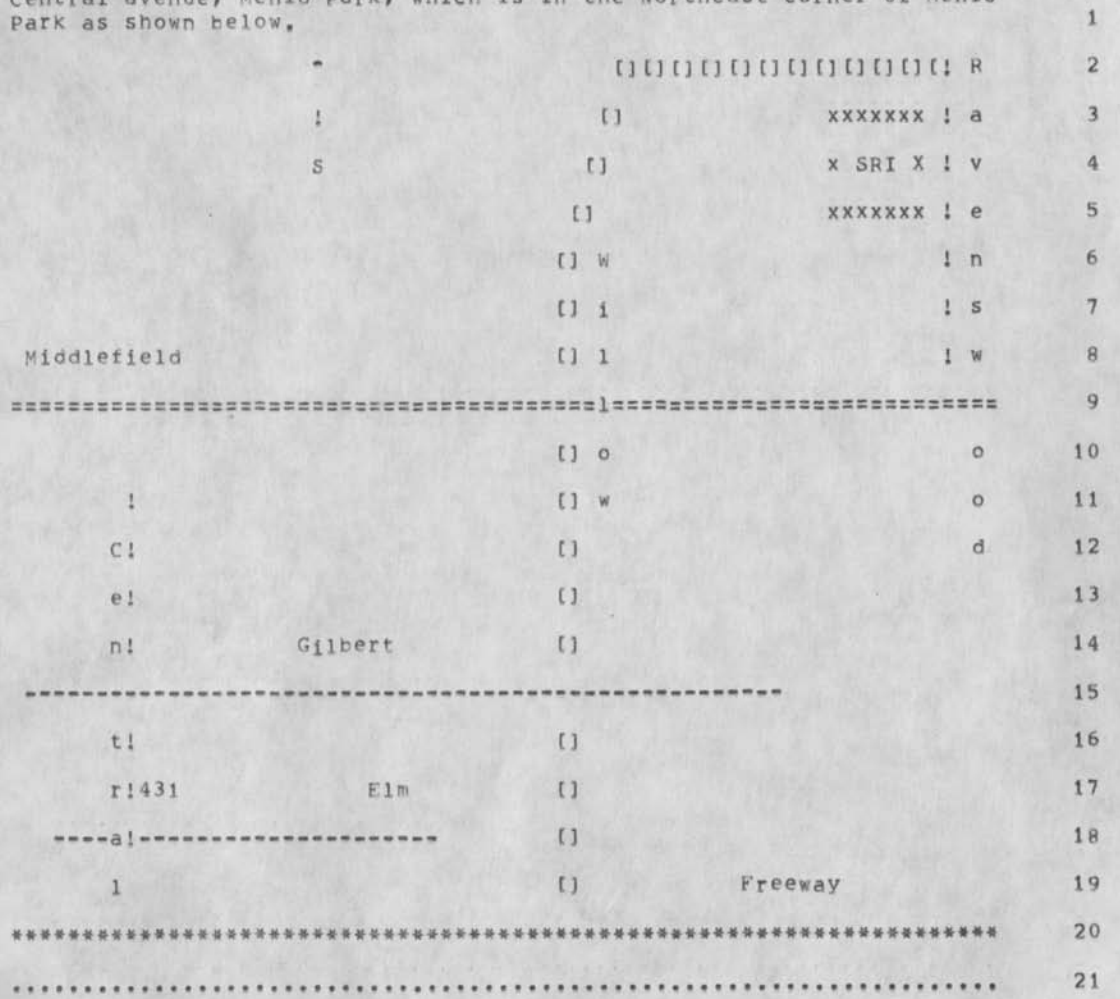
Dialog Concerning Copy Directory

(J26390) 5-SEP-75 12:53;;;; Title: Author(s): Dirk H. Van Nouhuys/DVN; Distribution: /FEEDBACK([ACTION]) JDH([INFO-ONLY]) ; Sub-Collections: SRI-ARC FEEDBACK; Clerk: DVN;

26390 Distribution
Special Jhb Feedback, J. D. Hopper,

Party

Although he explains that through the miracle of modern electronic communication he is not really going away, on Saturday September 13 in the evening were are giving a going away patry for Kirk. It is also for my daughter's birthday. You are invited. There will probably be a lot of people and dancing. Sangria and video tapes of the Whole Universe Catalog will be avaialble. Our address is 431 Central avenue, Menlo Park, which is in the Northeast corner of Menlo Park as shown below.



Party

..... 22
..... 23
..... 24

Party

(J26391) 5-SEP-75 13:09;;;; Title: Author(s): Dirk H. Van Nouhuys/DVN; Distribution: /SRI-ARC([ACTION]) PGK([ACTION]) PWO([ACTION]) TLH([ACTION]) DCW([ACTION]) LPD([ACTION]) GA([ACTION]) RE([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: DVN; Origin: < VANNOUHUYS, MYLIN,NLS;126, >, 5-SEP-75 13:05 DVN ;;;;(; EXTERNAL LINKS: <vannouhuys,dvn,>####;

26391 Distribution

Douglas C. Engelbart, Martin E. Hardy, J. D. Hopper, Charles H. Irby, Harvey G. Lehtman, James C. Norton, Jeffrey C. Peters, Dirk H. Van Nouhuys, Kenneth E. (Ken) Victor, Richard W. Watson, Don I. Andrews, Pamela G. Kruzic, Pat Whiting O'Keefe, Thomas L. Humphrey, Donald C. (Smokey) Wallace, L. Peter Deutsch, Gerald Agin, Robert Engelmore, David C. Smith, Mary Ann Kellan, Buddie J. Pine, Andy Poggio, David L. Retz, Laura J. Metzger, Karolyn J. Martin, Jan A. Cornish, Larry L. Garlick, Priscilla A. Wold, Pamela K. Allen, Delorse M. Brooks, Beverly Boli, Rita Hysmith, Log Augmentation, Joseph L. Ehardt, Raymond R. Panko, Susan Gail Roetter, Robert Louis Belleville, Rene C. Ochoa, Ann Weinberg, Adrian C. McGinnis, Robert S. Ratner, David S. Maynard, Robert N. Lieberman, Sandy L. Johnson, James H. Bair, Jeanne M. Leavitt, Rodney A. Bondurant, Jeanne M. Beck, Marcia L. Keeney, Elizabeth K. Michael, Jonathan B. Postel, Elizabeth J. Feinler, Kirk E. Kelley, N. Dean Meyer, James E. (Jim) White

RLL 5-SEP-75 15:53 26392

CONFERENCE: WP conf 16 Sept. - someone should go.

I strongly urge that one or more people from ARC attend the WP conference on 16 Sept. Since it is in San Fran, it will be easy. I do not understand why the cost if so high? Could it be that it is not a conference but rther a workshop? If so I have reservations on the need to go. Has anyone followed up on this??? Rob 1

RLL 5-SEP-75 15:53 26392

CONFERENCE: WP conf 16 Sept. - someone should go.

(J26392) 5-SEP-75 15:53;;;; Title: Author(s): Robert N. Lieberman/RLL; Distribution: /DCE([ACTION]) RWW([ACTION]) DVN([INFO-ONLY]) SGR([INFO-ONLY]) ARC-LOG([INFO-ONLY]) ; Sub-Collections: SRI-ARC ARC-LOG; Clerk: RLL;

26392 Distribution
Douglas C. Engelbart, Richard W. Watson, Dirk H. Van Nouhuys, Susan Gail Roetter, James C. Norton, Log Augmentation,

Weekly Report for Aug 11-15

Blank form 1

USER SERVICES WEEKLY REPORT for Week of August 11-15: 1a

from JMB 1a1

Groupstat document finished. 1a1a

DEX document finished 1a1b

Spent one day recovering from work lost during Tuesday's Office-1 crash. 1a1c

Worked on methods for inputing and editing tables (columnar material) for the people at Gunter; found a pretty good way to input the stuff. 1a1d

Got ready for next week's trip to Gunter. Found out that trip to AMC in New Jersey is cona 1a1e

from SGR 1a2

Monday - Talked with US people on the alledged AMC trip to find out the latest word. Talked with JCN on the switch of US people to BBNB and proceeded to get that all going. Had a meeting with JCN, EKM, BJP on problems at Gunter and how many programming resources could be directed that way. Did a final review of viewgraphs and started work on the tripreport. 1a2a

Tuesday - Talked with JHB on Gunter, the seminar and other issues. Talked with Rita on the state of things at ARPA. Worked a bit on table helps produced by Kirk and RLL. 1a2b

Wednesday - More work on tables, met with RWW and EKM and JCN to discuss the Crabtree application at Gunter. It was decided that PKA and PAW2 would make a stab at it. Spent more time getting things together for the AMC and Gunter trips. Took Pam to the hospital... 1a2c

Thursday - More work on the tripreport, using columns, the Crabtree application, and getting ready for the seminar. 1a2d

Friday - Meetings with JCN and JHB, then with US and the above on procedures for the interaction of the two groups. Final trip stuff and more work on the seminar. 1a2e

from RH 1a3

Weekly Report for Aug 11-15

Had a very enjoyable week in California. Nice to get back "in touch" with people. Learned MSG with Pam, Pricilla and Jeanne, which was nice since we were able to iron out all of our questions together. Met with Susan on ARPA, Dean on some additional Output Processor questions, Jim and Susan on ARPA and Jim Bair. Also attended the first "complete" User services and Applications documentation meeting, which cleared up many questions, I'm sure for all of us. I also received my copies of the viewgraphs which are beautiful and gathered documentation that I need in Washington. I also went around getting questions answered on particular problems that needed answering. 1a3a

from PKA 1a4

Spent more time with MSG all week. It was decided that I would take the trip to teach MSG and Tenex next week instead of Jeanne. So I worked quite a lot on preparing to teach these courses. Had a meeting with user services and documentation and jcn to discuss general policies. Had a meeting with Susan and other trainers early in the week to discuss what had happened while she and Priscilla had been at Gunter. Spent a few hours off and on talking to Rita about her work in Washington and things that are going on there. 1a4a

from PAW2 1a5

Busy, hectic week- good to visit with Rita, to hear briefly of the activities in Washington. 1a5a

Monday- Got caught up on mail/messages I'd neglected while at Gunter, jotted down notes from trip to Gunter, met with Toni again about viewgraphs. Final drafts needed approval before graphs could be produced. 1a5b

Tuesday-Wednesday looked over the MSG document developed at ISI. Can foresee a situation in the near future where the third course may be needed, so started through that systematically, marking sections that were unclear to me and areas where I felt I needed practice. Glanced over the old tenex userguide in conjunction with the MSG document, again noting new or unclear material. 1a5c

Thursday- spent time with Raby Panko on the statistics report for Office-1. That afternoon went over MSG with Rita and Pam. Spoke with Susan about writing a process commands branch for J. Crabtree at Gunter. He needs to have a branch written that will sort and organize all the problems and

solutions he handles in his work with the Direps at Gunter. Will be using part of the retrieve subsystem but am unclear of any details. 1a5d

Friday- sent off the statisitcs report for Office-1 to all architects. Read over retrieve subsystem document. Viewgraphs finally arrived, four sets of each, they look pretty good. We're still lacking a couple of graphs but should get them next week. Glad to see them through the production process, I think they'll be very useful. Interesting meeting Friday afternoon to discuss our boundaries and roles between training and documentation. 1a5e

SGR PKA PAW2 RH JMB 5-SEP-75 20:10 26393

Weekly Report for Aug 11-15

(J26393) 5-SEP-75 20:10;;;; Title: Author(s): Susan Gail Roetter, Pamela K. Allen, Priscilla A. Wold, Rita Hysmith, Jeanne M. Beck/SGR PKA PAW2 RH JMB; Distribution: /US([INFO-ONLY]) JCN([INFO-ONLY]) JHB([INFO-ONLY]) ; Sub-Collections: SRI-ARC US; Clerk: PKA;

26393 Distribution
Susan Gail Roetter, Priscilla A. Wold, Jeanne M. Beck, Pamela K. Allen, Rita Hysmith, Sandy L. Johnson, James C. Norton, James H. Bair,

Weekly Report for Aug 18-22

Blank form 1

USER SERVICES WEEKLY REPORT for Week of August 18-22: 1a

from JMB - on travel at Gunter; see trip report 1a1

from SGR 1a2

I contacted Connie McLindon on how best to renew our efforts at ARPA (followed by a discussion with Rita). Also contacted Carol Mahoney to discuss training next week for what was alledgedly a new secretary at the Pentagon and turned out to be two guys from Nebraska who want to learn NLS in one day and return to edit a volume of 66-1. Also contacted Dave Potter about ETS training the second week of September when one of us will be in Ft. Monmouth, New Jersey. 1a2a

Spent time working with Priscilla on Crabtree's application and other miscellaneous talks with Ann, Jim N, and Elizabeth about various Gunter problems. Planned for next week when Grace and Joann will be here from Gunter to become "expert" editors. Requested work from ARC'ers for them to practice on. 1a2b

Worked to get ready for the seminar. This involved attending several meetings and expanding the outline in the agenda and getting files collected to have in their directories. 1a2c

Transferred files to BBNB and notified people of our move. 1a2d

Journalized the tripreport from the course at Gunter. 1a2e

from RH 1a3

Gave the Basic Course to Fred Hollister of ARPA. Talked with Susan about increasing the slot usage at ARPA; also talked to Connie on the same subject. Got ready for the training at the Pentagon next week. Transferred files over to BBNB. Delivered some proofs for Elizabeth to the Pentagon. As usual, trouble-shooted around ARPA with the users, especially trying to get them more active. Had one surprise, Hilda called me for help with NLS, it seems she is entering a small book in the system for Dr. Licklider. 1a3a

from PKA- On travel to AMC sites. See Trip report. 1a4

from PAW2 1a5

Weekly Report for Aug 18-22

Mon- Gave my initial file a major face lift... tried to catch up on backlog of unread mail. Went over Tenex User's Guide with Pam, she pointing out what she thought was essential information for MSG training. Read over Retrieve Subsystem documentation in anticipation of working on Crabtree's process command branch. Tues-Wed Made up a file of questions to send to Crabtree, questions dealing with format and file names etc. Met with Susan to discuss the process involved, and began work using the Retrieve Subsystem. Thurs. Did some things in preparation for next weeks Seminar. Forwarded some mail to different directories set up for the seminar using forward command in Sendmail, a new one to me. Met briefly with Jim Norton about Crabtree's branch. He gave me some useful hints and suggestions. Fri. Worked on my own message commands branch, still not working correctly. Set up message handling branches in phony directories for the Seminar. Continued on Crabtree's branch.

1a5a

Weekly Report for Aug 18-22

(J26394) 5-SEP-75 20:18;;;; Title: Author(s): Susan Gail Roetter, Priscilla A. Wold, Rita Hysmith/SGR PAW2 RH; Distribution: /US([INFO-ONLY]) JCN([INFO-ONLY]) JHB([INFO-ONLY]) ; Sub-Collections: SRI-ARC US; Clerk: PKA;

26394 Distribution

Susan Gail Roetter, Priscilla A. Wold, Jeanne M. Beck, Pamela K. Allen, Rita Hysmith, Sandy L. Johnson, James C. Norton, James H. Bair,

Weekly Report for Aug 25-29

Blank form 1

USER SERVICES WEEKLY REPORT for Week of August 25-30: 1a

from JMB- See Course Report 1a1

from SGR 1a2

Most of my time was spent involved some way with the AKW Seminar, Getting materials together, revising the course outline to reflect changes in the agenda, listening to various sessions and conducting the on-line sessions took at least 3/4 of my time. 1a2a

Other time was spent coordinating Sandy's vacation and Pam's taking over feedback, getting the trip planned to New Jersey which Priscilla will be making the second week of September, and meetings with Jim Norton to discuss plans to be made while he was gone for training of new slot buyers and the NSA programmers. A little time was spent with Grace and JOann, primarily to see how they were doing, and discussing problems of teaching DNLS with Jeanne. 1a2b

from RH 1a3

Unbelievably busy week. Spent day and a half at the Pentagon giving training. Worked with Hilda on the book. Made arrangements to get together with different individuals for additional training here at ARPA. And now the big surprise, Cyr, the Acting Director of PM, has decided that the budget review should be coordinated and performed by MIS using NLS. I met with CYR myself the latter part of the week. We have been going like crazy, setting up the basic form, how and who is going to input the information etc. Everybody is interested in NLS now. I guess everyone was just waiting for the end of the summer to get energetic. Anyway there is definitely a renewed interest in NLS here at ARPA. 1a3a

I was also very busy with Dean and his demos. My display went out and we had to borrow one from ARPA and then the Modem went out at the Mitre end. Such confusion and bad timing. Anyway, I ended up working a full week and was never able to take the vacation time as planned. I also fixed up the MRAO's again, there was still structure problems that kept popping up, I'm 99% sure that they are all cleared up now. 1a3b

from PKA 1a4

Weekly Report for Aug 25-29

Spent Monday morning listening to Seminar talks. Then started working with Sandy very seriously to learn Feedback. Did this part of every day and all day on Thurs and Fri. Spent one afternoon with Priscilla working on Crabtree's branch. She had already done most of the work and we were just ironing out a few bugs. It's almost ready to work. In order to understand the branch had to read up on retreive and content patterns. 1a4a

from PAW2 1a5

Monday- began giving Basic Course to three in-house people. Attending were Dan Lynch and Geoff Goodfellow both from AI, and Lance Murphy from Operations Evaluations. Pretty well covered the Basic Course in the morning with the exception of TENEX material. Both Dan and Geoff knew TENEX very well. Spent afternoon with Lance finishing up the Basic Course. Tuesday morning covered Second Course with Dan and Geoff, Lance unable to attend. That afternoon worked on Office-1 statistics with Ra3y Panko. Met with Lance Wed. morning to begin the Second Course, got about half way through it, to be finished up next week. Wed. afternoon worked on Crabtree's branch with Pam explaining the process and the bugs to her. She will be working on it the latter part of this week. It's almost there. Spent a good hour linked to Crabtree trying to straighten out his problems with content analyzer patterns. Took Thurs. and Fri. off. 1a5a

Weekly Report for Aug 25-29

(J26395) 5-SEP-75 20:34;;;; Title: Author(s): Priscilla A. Wold, Rita Hysmith, Pamela K. Allen, Susan Gail Roetter/PAW2 RH PKA SGR; Distribution: /US([INFO-ONLY]) JCN([INFO-ONLY]) JHB([INFO-ONLY]) ; Sub-Collections: SRI-ARC US; Clerk: PKA;

26395 Distribution
Susan Gail Roetter, Priscilla A. Wold, Jeanne M. Beck, Pamela K. Allen, Rita Hysmith, Sandy L. Johnson, James C. Norton, James H. Bair,

no changes from last time but a reminder to look at it.

Introduction 1

Data structures will be encoded according to PCPB8 when the physical channel transmits messages which are streams of 8-bit bytes. 1a

Data Structure Encoding 2

The first byte of a data structure is a type code, with the type zero having the special interpretation indicating that a key is present for this data structure, non-zero codes indicate element types. 2a

Key 2b

FLAG (1 byte) = 0 2b1

VALUE (any element) 2b2

Elements 2c

EMPTY 2c1

TYPE (1 byte) = 1

VALUE (none) empty

BOOLEAN 2c2

TYPE (1 byte) = 2

VALUE (1 byte) boolean

FALSE=0

TRUE =1

INDEX 2c3

TYPE (1 byte) = 3

VALUE (2 bytes) index

The value represents a positive integer in the range 1 through 2**15 - 1

INTEGER 2c4

TYPE (1 byte) = 4

VALUE (4 bytes) two's complement integer

BITSTR 2c5

TYPE (1 byte) = 5

COUNT (3 bytes)

VALUE (count bits) left adjusted in ((count+7)/8) bytes)

CHARSTR 2c6

TYPE (1 byte) = 6

COUNT (3 bytes)

VALUE (count bytes) ascii text

LIST 2c7

TYPE (1 byte) = 7

COUNT (3 bytes)

Note: Lists of unspecified length are specified by setting the COUNT to all ones, The end of such a list is indicated by a byte of all ones in place of a TYPE field,

REPEAT (1 byte)

SPECIFIEDELEMENTS=0

Count Data Structures

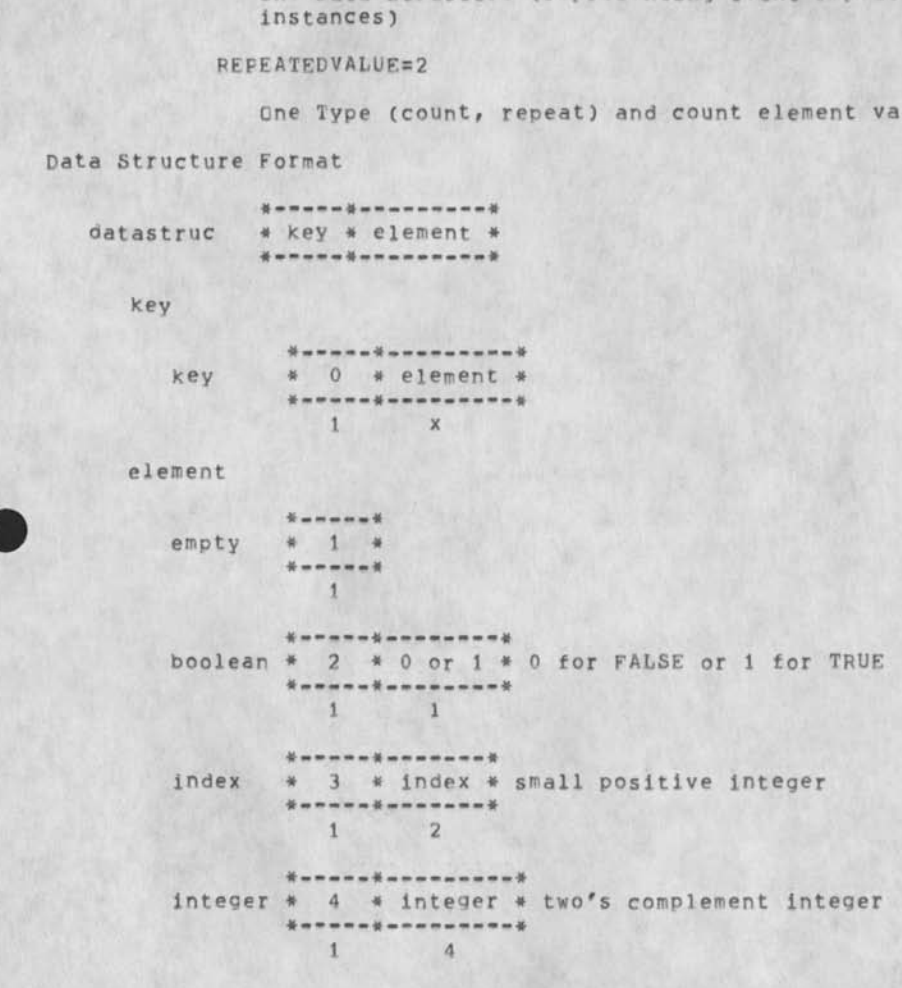
REPEATEDELEMENT=1

One Data Structure (representing count repeated instances)

REPEATEDVALUE=2

One Type (count, repeat) and count element values

Data Structure Format 3



```
            *-----*-------*------*
bitstr      *  5  * count * bits *
            *-----*-------*------*
               1      3     count   ((count+7)/8 bytes)

            *-----*-------*------*
charstr     *  6  * count * text * Network ASCII
            *-----*-------*------*
               1      3     count

            *-----*-------*--------*------------------*
list        *  7  * count * repeat * count-structures *
            *-----*-------*--------*------------------*
               1      3       1
```

Examples 4

Empty 4a

```
*-----*
*  1  *
*-----*
```

4a1

Boolean "TRUE" 4b

```
*-----*-----*
*  2  *  1  *
*-----*-----*
```

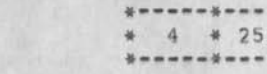
4b1

Index "7" 4c

```
*-----*-----*-----*
*  3  *  0  *  7  *
*-----*-----*-----*
```

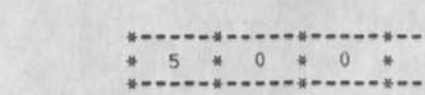
4c1

Integer "-3" 4d



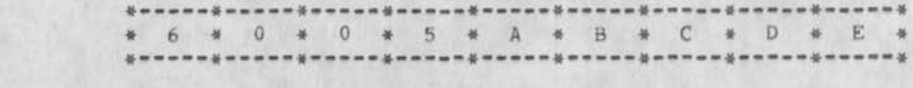
4d1

Bit string "10001111101011" 4e



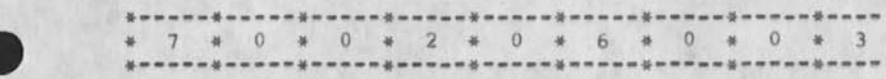
4e1

Character string "ABCDE" 4f

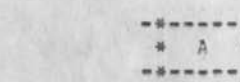


4f1

List of a character string "ABC" and a boolean "FALSE" 4g

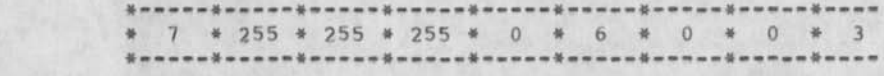


4g1

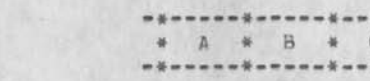


4g2

or 4h



4h1



4h2

List of two the booleans "TRUE", "FALSE" 4i

7	0	0	2	0	2	1	2	0

4i1

or 4j

7	0	0	2	2	2	1	0

4j1

Boolean "TRUE" with character string Key "X1" 4k

0	6	0	0	2	X	1	2	1

4k1

(J26396) 6-SEP-75 23:06;;;; Title: Author(s): Jonathan B. Postel/JBP; Distribution: /CHI([INFO-ONLY]) DLR([INFO-ONLY]) DAV([INFO-ONLY]) JEW([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: JBP; Origin: < POSTEL, PCPB8.NLS;7, >, 17-JUL-75 16:44 JBP ;;;;####;

26396 Distribution
Charles H. Irby, David L. Retz, David C. Smith, James E. (Jim) White,

JBP 7-SEP-75 01:28 26397

Whats with Forward ???

I am getting0the error message "no such document" when i try to forward journal items, and when i jump to link on the journal number i ge "not online use interrogate" but if i jump to link on the directory,number, link i get the document loaded, this happens with 1journal,33153 and with 1journal,26231 these are not very recent documents but dated 6-aug-75 and 4-aug-75 respectivly, --jon 1

JBP 7-SEP-75 01:28 26397

Whats with Forward ???

(J26397) 7-SEP-75 01:28;;;; Title: Author(s): Jonathan B. Postel/JBP; Distribution: /FEED([ACTION]) JCP([ACTION]) BJP([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: JBP;

26397 Distribution
Special Jhb Feedback, Jeffrey C. Peters, Buddie J. Pine,

documents you should read

You shoul be sure that you have seen all of the following (26232,) (33153,) (26283,) (26285,) (26282,) (26293,) (26284,) (26271,) (26249,) (26229,) (26231,) (26222,); also (26267,) is the code of DPS-10 its lots of paper so think before printing. --jon. 1

documents you should read

(J26398) 7-SEP-75 01:35;;;; Title: Author(s): Jonathan B. Postel/JBP; Distribution: /DAV([ACTION]) DLR([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: JBP;

26398 Distribution
David C. Smith, David L. Retz,

JBP 7-SEP-75 01:37 26399

design ideas ?

where is the file of design ideas so carefully culled from all the messages to feedback kept in case i ever get time to read it ? --jon. 1

JBP 7-SEP-75 01:37 26399

design ideas ?

(J26399) 7-SEP-75 01:37;;;; Title: Author(s): Jonathan B. Postel/JBP; Distribution: /FEED([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: JBP;

26399 Distribution
Special Jhb Feedback,

JBP 7-SEP-75 01:40 26400

shopping list updates needed

Elizabeth: you should still make the improvements to the shoppinglist files this week, as the files will be used in discussions with carlson next week --jon. 1

JBP 7-SEP-75 01:40 26400

shopping list updates needed

(J26400) 7-SEP-75 01:40;;;; Title: Author(s): Jonathan B. Postel/JBP; Distribution: /EKM([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: JBP;

26400 Distribution
Elizabeth K. Michael,

RLL 7-SEP-75 18:33 26401

BUG: in journal files...

In journal items 33422, and <33420,> I got an error message when attempting to print plex, "illegal string designation", Iit appears to be in statement 3 of JHB's item and in statement 0 of DCE's. 1

RLL 7-SEP-75 18:33 26401

BUG: in journal files...

(J26401) 7-SEP-75 18:33;;;; Title: Author(s): Robert N. Lieberman/RLL; Distribution: /FEED([ACTION]) ARC-APP([INFO-ONLY]) ; Sub-Collections: SRI-ARC ARC-APP; Clerk: RLL;

26401 Distribution

Special Jhb Feedback, Buddie J. Pine, Laura J. Metzger, Priscilla A. Wold, Pamela K. Allen, Rene C. Ochoa, Jeffrey C. Peters, Marcia L. Keeney, Jeanne M. Beck, Geoffrey S. Goodfellow, Rodney A. Bondurant, Douglas C. Engelbart, Jeanne M. Leavitt, Susan Gail Roetter, Raymond R. Panko, Adrian C. McGinnis, James C. Norton, J. D. Hopper, Elizabeth J. Feinler, James H. Bair, Robert N. Lieberman, N. Dean Meyer, Sandy L. Johnson, Martin E. Hardy,

Visit Log: Sept. 5, 1975 Dr. Robert Herriot University of Washington

(visit-log) Sept 5, 1975 1

Dr. Robert Herriot 1a

Computer Science Department 1b

University of Washington 1c

Seattle Washington 98195 1d

Computer Science Faculty 1e

Interests: 1f

Programming Languages 1f1

Text Editing Systems 1f2

Also developing their own mini-computer, LM2. Will perhaps implement a Text Editor on this machine. Current Hardware includes a Sigma 5. 1f3

Log: 1g

I gave Dr. Herriot a short (30 minute) demonstration of NLS. We discussed the system design, particularly that of the command language parser. His interest in NLS stems from his desire to provide the users of the University of Washington's Computer facility with more sophisticated editing tools. He showed little interest in becoming a subsciber to NLS, because the University is planning to develop their own text editor either on their Sigma 5 or perhaps on a mini they are building. He did express interest in obtaining more documentation on NLS and CML. 1g1

Documentation: 1h

The Augmented Knowledge Workshop 1h1

Coordinated Information Services for a Discipline or Mission Oriented Community 1h2

NLS-8 Command Summary 1h3

Cue Card 1h4

Output Processor Guide 1h5

Investments in Tomorrow 1h6

Visit Log: Sept. 5, 1975 Dr. Robert Herriot University of Washington

L10 document 1h7

CML chapter from NDM's L10-Guide 1h8

Follow-up: 1i

Robert has requested a copy of NDM's L10-Userguide when it is published. 1i1

DSM 7-SEP-75 23:56 26402

Visit Log: Sept. 5, 1975 Dr. Robert Herriot University of Washington

(J26402) 7-SEP-75 23:56;;;; Title: Author(s): David S. Maynard/DSM; Distribution: /DCE([INFO-ONLY]) JHB([INFO-ONLY]) RLL([INFO-ONLY]) ARC-LOG([INFO-ONLY]) ; Sub-Collections: SRI-ARC ARC-LOG; Clerk: DSM;

26402 Distribution

Douglas C. Engelbart, James H. Bair, Robert N. Lieberman, James C. Norton, Log Augmentation,

DSM 8-SEP-75 04:13 26403

Rough Draft of Documentation for the Middle-End, an L10 interface to existing NSW protocols.

Comments? Suggestions? A note will follow describing the special case of the CLI/NLS interface.

Rough Draft of Documentation for the Middle-End, an L10 interface to existing NSW protocols,

Introduction 1

A body of code has been written which can serve as an interface between any tool written in L10 to the DPS protocol or to the MSG protocol. This body of code was originally designed to interface the NLS - 9 Back End to the CLI through the DPS protocol. It has since been generalized to support the interaction between DPS or MSG and an arbitrary L10 Back-End. For the special (but common) case of an L10 tool communicating with the Command Language Interpreter (CLI) a third protocol called Shared page or simply SHARED is also supported. It is hoped that this code will make it easier to install L10 tools which interact with the CLI and possibly with other NSW processes, such as the Works Manager. 1a

It is assumed that the reader is familiar with the DPS protocol, the MSG protocol, and the extended version of L10. References to the appropiate documents are contained in the Bibliography. 1b

Functional Capabilities 2

The primary function of the middle-end is isolate a tool written in L10 from the protocol used for interprocess communication. Total isolation from the protocol is impossible because the various protocols support different functional capabilities. The basic function which is common to all three protocols is that of a procedure call, either from a remote process to a procedure in the L10 tool, or vice versa. This call may have arguments, and may return results. The middle-end succesfully isolates the tool from the protocol for this basic function. 2a

The middle-end also supplies a set of protocol dependent routines which allow the tool to take advantage of the additional functional capabilities of the particular protocol being used. These protocol dependent routines will be documented in a later note. 2b

Overview 3

The middle-end is a set of object files which one link loads with an L10 tool. This makes available to the tool a set of procedures which can be used to communicate with other processes via the DPS, MSG or SHARED protocol. The tool itself when first started performs its own initialization and then makes calls on procedures within the middle-end providing information about the tool, such as the protocol desired, and the names and address of those procedure within the tool which are capable of being called remotely. 3a

If the tool is a "passive tool", that is one being driven by requests from another process (such as the CLI) it then transfers control to a procedure in the middle-end. The procedure then acts a a dispatcher, receiving procedure call requests from external processes and performing the call of the local procedure. The middle-end handles conversion of arguments and results from the protocol format to it's own internal L10 format. A handle is also provided if the tool wants to perform additional transformations on its arguments. The local procedure may also make calls on remote procedures through the middle-end. 3b

Argument Conversion and Conventions for Externally Callable Procedures 4

All of the protocols supported by the middle-end use the PCPB36 encoding for arguments and results of procedures. Data elements are fully typed in PCPB36. This represents somewhat of a problem in the L10 environment in which data elements are not fully typed. This problem was solved by making use of the extensions to L10 providing typed lists, and list handling primitives. Thus an L10 LIST provides a convenient way to store both the type and value of a data element. 4a

When the middle-end receives a request for the execution of a local procedure it is given a PCPB36 LIST of arguments. It converts this list to an L10 LIST, and passes each element of the list to the procedure as an argument. In addition the middle-end passes one extra argument to each externally callable procedure. This is the address of an L10 LIST into which the procedure stores its results. The middle-end then translates this list into a PCPB36 LIST which it returns as the result of the procedure call. The procedure should RETURN[TRUE] if it succeeds and RETURN[FALSE] if it fails. 4b

The following table defines the transformation between PCPB36 and L10 list types. Note the L10 list types mentioned below include the user settable bits in the List element descriptor and is accessed as desr.ledubv where descr is a DESCR of a list element , see (andrews,l10lists,). The lower case identifiers used below beginning with the letter u are external constants defined within the middle-end. 4c

PCPB36	L10 LIST type	value passed as argument	4c1
			4c2
EMPTY	unull	0	4c3

BOOLEAN	uboole	1 or 0 for True or False	4c4
INDEX	uindex	value (>0 & < 2**15)	4c5
INTEGER	uinteg	value	4c6
BITSTR	ubitst	address of a block whose first	4c7
		word contains a bit count	4c8
CHARSTR	ustrin	address of an L10 string	4c9
LIST	ulist	address of an L10 list	4c10

Tool Dependent Argument Conversion 5

The middle-end provides a handle in case a tool wishes to perform additional transformations on the arguments passed to its procedures. The handle is the address of a tool supplied procedure which is called to process each argument before being passed to the externally callable procedure. This tool supplied procedure must take the following formal arguments: 5a

rawarglist 5a1

The address of the L10 LIST produced by the middle-end from the PCPB36 argument list for the procedure being called. 5a1a

convertedarglist 5a2

The address of an L10 list which is used to store the converted arguments. 5a2a

listindex 5a3

The index of the list element to be converted. 5a3a

errstring 5a4

The address of an l10 string into which any error messages should be stored. 5a4a

The tool dependent argument converter should have the following semantics: 5b

This procedure will be called once for each argument. It should compute a transformed argument from #rawarglist#[listindex] and store it in #convertedarglist#[listindex]. It should

RETURN[TRUE] if it succeeds and RETURN[FALSE] otherwise in which case it should store an error diagnostic in errstring. 5b1

Using the Middle-End 6

A tool process is started at location specified by the first location of its entry vector. The tool should provide its own entry procedure, and initialize the L10 environment. It should then perform any tool specific initialization tasks. The tool should then call the following procedures in the middle-end: 6a

(initmiddle) PROCEDURE %initialize tool parameters% 6a1

FORMALS 6a1a

processname, 6a1a1

the address of an L10 string containg the process name for this process, 6a1a1a

argconverter, 6a1a2

the address of a procedure which will perform tool dependent argument conversion, or 0 implying no tool dependent argument conversion, 6a1a2a

toolcleanup 6a1a3

the address of a procedure supplied by the tool which will be invoked by the middle-end after each externally called procedure returns to the middle-end, or 0 implying none. The toolcleanup procedure takes no arguments, 6a1a3a

(crtlpkg) PROCEDURE %specify externally callable procedures% 6a2

FORMALS 6a2a

pkgname, 6a2a1

the address of an L10 string containing the name of the group of procedures being defind in this call to crtlpkg. Note: when using DPS crtlpkg may be called several times to define different packages within the tool. When using MSG or SHARED crtlpkg may only be called once, 6a2a1a

packagedescriptor, 6a2a2

the address of a block containg a package descriptor. The package descriptor consists of two words for each procedure which is externally callable. The first word is the address of an L10 string containing the name (upper case) by which the procedure is known externally, the second word contains the address of the procedure. The block must be terminated by two zero words. 6a2a2a

hashtableaddress, 6a2a3

The address of a data block which will be used by the middle-end as a hash table for the procedure names. If 0 (or if the middle-end decides the table specified is not large enough) a hash table will be allocated by the middle-end. 6a2a3a

hashtablesize 6a2a4

The size of the hash table in words whose address was passed in hashtableaddress. Not used if hashtableaddress = 0. 6a2a4a

(execmiddle) PROCEDURE % Dispatches Remote calls to Local procedures% 6a3

FORMALS 6a3a

howpcp, 6a3a1

= 1 implies SHARED 6a3a1a

= 2 implies DPS 6a3a1b

= 3 implies MSG 6a3a1c

priority 6a3a2

used only for DPS (howpcp = 2). Specifies the interupt level at which externally called procedures execute. Legal values are 1, 2 or 3. Used to allow externally called procedures to trap interupts which occur at a higher priotity level than the one passed. The lower the priority number, the higher the priority of the interrupt. 6a3a2a

For example if a tool wanted to trap write psuedo interupts which occur at priority level 2 , a 3 must

be passed as priority to execmiddle (otherwise the externally callable procedure will run as a priority level 2 interrupt and would therefore not see another priority level 2 interrupt such as a write psuedo interrupt) 6a3a2b

Additional notes for the DPS case: The middle-end defines the forks psuedo-interrupt table. This table is called chntab. Channels 4 and 5 are used by the middle-end. The tool is free to use other channels. 6a3a2c

The procedure execmiddle will not return to its caller. It will instead wait for requests from remote processes to perform local procedure calls and dispatch these calls. 6b

Calling External Procedures 7

The following procedure can be used to call procedures in other processes. 7a

```
(extcall) PROCEDURE %Calls External procedure MSG, SHARED or
PCP protocol%                                                    7a1

    ( dmail, %MSG: destination mailbox PCP: process handle%      7a1a

    pname REF,  %addr of string contain procedure name%          7a1b

    callmode,  %MSG:=1 no acknow,%
               %MSG:=2 in-line w/ ack%
               %MSG:=3 out-of-line w/ ack%
                   %PCP: package handle%                         7a1c

    argl REF,  %addr of arg list%                                7a1d

    resl REF  %adr of result list%);                             7a1e

                                                                 7a1f

                                                                 7a1g
```

Files 8

The following files must be loaded to use the middle-end. Object files are specified. In addition for those data files which contain data blocks whose size is optional I have also specified the source file. The tool builder may edit the source file to specify the desired size of these data blocks and compile the

files to his own object file which should then be loaded with the others. All files are maintained at ISIC. 8a

<rel-nls>l10lrp.rel - list run time package 8a1

<nsw-sources>festgmgt.rel - dynamic storage allocator 8a2

<relnine>middle.rel - middle-end code 8a3

<relnine>mconst.rel - middle-end constants 8a4

<relnine>mdataf.rel - middle-end data 8a5

(nine,mdataf,nls,) the array fsblist is used as a freestorage area for all list elements. Default size is 10000 octal 8a5a

<relnine>mshared.rel - protocol data area 8a6

(nine,mshared,nls,) the array dpsresults is used as a data area for the protocols. Default size is 10000 octal. 8a6a

Note: For howpcp=1 i.e. SHARED protocol this file must be loaded at location 300000 octal (shares pages with CLI). 8a6b

References 9

Error Handling 10

DSM 8-SEP-75 04:13 26403

Rough Draft of Documentation for the Middle-End, an L10 interface to existing NSW protocols.

(J26403) 8-SEP-75 04:13;;;; Title: Author(s): David S. Maynard/DSM; Distribution: /KEV([ACTION]) DLR([ACTION]) NPG([INFO-ONLY]) ; Sub-Collections: SRI-ARC NPG; Clerk: DSM; Origin: < MAYNARD, MIDDLEDOC.NLS;2, >, 8-SEP-75 03:55 DSM ;;;;####;

26403 Distribution

Kenneth E. (Ken) Victor, David L. Retz, David C. Smith, Andy Poggio, David L. Retz, Jan A. Cornish, Larry L. Garlick, Robert Louis Belleville, Elizabeth J. Feinler, Joseph L. Ehardt, Jonathan B. Postel, Kirk E. Kelley, Karolyn J. Martin, David S. Maynard, Kenneth E. (Ken) Victor, James E. (Jim) White, Elizabeth K. Michael, Don I. Andrews, J. D. Hopper, Charles H. Irby, Harvey G. Lehtman,

DMB 8-SEP-75 12:42 26404

Biographies

I am still in the process of updating ARC Bio's. However, due to the expected upcoming Proposals, the need for new and up-to-date Biographies has increased.

Please help me by checking your Bio's for errors and for updates, then take a copy to RWW to check so I may put them online. Any other Professional Staff members that do not have existing Bio's should see me ASAP. Thanks-Dee 1

Biographies

(J26404) 8-SEP-75 12:42;;;; Title: Author(s): Delorse M. Brooks/DMB; Distribution: /SRI-ARC([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: DMB;

26404 Distribution

Douglas C. Engelbart, Martin E. Hardy, J. D. Hopper, Charles H. Irby, Harvey G. Lehtman, James C. Norton, Jeffrey C. Peters, Dirk H. Van Nouhuys, Kenneth E. (Ken) Victor, Richard W. Watson, Don I. Andrews, David C. Smith, Mary Ann Kellan, Buddie J. Pine, Andy Poggio, David L. Retz, Laura J. Metzger, Karolyn J. Martin, Jan A. Cornish, Larry L. Garlick, Priscilla A. Wold, Pamela K. Allen, Delorse M. Brooks, Beverly Boli, Rita Hysmith, Log Augmentation, Joseph L. Ehardt, Raymond R. Panko, Susan Gail Roetter, Robert Louis Belleville, Rene C. Ochoa, Ann Weinberg, Adrian C. McGinnis, Robert S. Ratner, David S. Maynard, Robert N. Lieberman, Sandy L. Johnson, James H. Bair, Jeanne M. Leavitt, Rodney A. Bondurant, Jeanne M. Beck, Marcia L. Keeney, Elizabeth K. Michael, Jonathan B. Postel, Elizabeth J. Feinler, Kirk E. Kelley, N. Dean Meyer, James E. (Jim) White

welcome

welcome

(J26405) 8-SEP-75 13:37;;;; Title: Author(s): E, TS ETSPeople/ETSP;
Distribution: /PAW2([INFO-ONLY]) ; Sub-Collections: NIC; Clerk:
ETSP;

26405 Distribution
Priscilla A. Wold,

practice message

practice today

PAW2 8-SEP-75 13:41 26406

practice message

(J26406) 8-SEP-75 13:41;;;; Title: Author(s): Priscilla A. Wold/PAW2; Distribution: /PAW2([INFO-ONLY]) PKA([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: PAW2;

26406 Distribution
Priscilla A. Wold, Pamela K. Allen,

GRB 8-SEP-75 13:47 26407

our black dog is lost

our black dog has been lost in your area if you find him please let me know,at number 921-9000 ex, 2219, 1

GRB 8-SEP-75 13:47 26407

our black dog is lost

(J26407) 8-SEP-75 13:47;;;; Title: Author(s): Greta R. Brown/GRB;
Distribution: /PKA([INFO-ONLY]) ; Sub-Collections: NIC; Clerk: GRB;

26407 Distribution
Pamela K. Allen,

sri
menlo park
ca

(J26408) 8-SEP-75 13:52;;;; Title: Author(s): Priscilla A. Wold/PAW2; Distribution: /PKA([INFO-ONLY]) SGR([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: PAW2;

26408 Distribution
Pamela K. Allen, Susan Gail Roetter,

biographies

Where are the Bios located? 1

JMB 8-SEP-75 16:15 26409

biographies

(J26409) 8-SEP-75 16:15;;;; Title: Author(s): Jeanne M. Beck/JMB; Distribution: /DMB([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: JMB;

26409 Distribution
Delorse M. Brooks,

JMB 8-SEP-75 16:41 26410

Journal BUG (I think in both O-1 & BBNB)

It seems (tho I have not exhaustively checked all possible situations) that when the Journal has an item that's short enough to put into the citation as a Message, it omits from the citation the ACTION or INFO-ONLY line. This is a great problem because that line (starting with *****) is the one where any comments associated with a particular IDENT in distribution would appear (as *****NOTE:...*****). These are not the same comments that are input with the Comment command, they are input after the IDENT with the Distribute command. This can cause me to miss important stuff (like notes from my boss that I'm the one who should do something about the item) that the sender assumes I have read when I get the item. One usually doesn't bother to jump to the actual journal file when the message appears in its entirety, one assumes, in one's initial file. 1

If you would like a sample of an item where this problem occurred, compare this citation as I received it --bbnb, jmb, 0653:gw>-- with the actual journal item cited in that branch to see the missing comment. 2

JMB 8-SEP-75 16:41 26410

Journal BUG (I think in both O-1 & BBNB)

(J26410) 8-SEP-75 16:41;;;; Title: Author(s): Jeanne M. Beck/JMB;
Distribution: /FEEDBACK([ACTION]) ARC-APP([INFO-ONLY]) ;
Sub-Collections: SRI-ARC FEEDBACK ARC-APP; Clerk: JMB;

26410 Distribution

Special Jhb Feedback, Buddie J. Pine, Laura J. Metzger, Priscilla A. Wold, Pamela K. Allen, Rene C. Ochoa, Jeffrey C. Peters, Marcia L. Keeney, Jeanne M. Beck, Geoffrey S. Goodfellow, Rodney A. Bondurant, Douglas C. Engelbart, Jeanne M. Leavitt, Susan Gail Roetter, Raymond R. Panko, Adrian C. McGinnis, James C. Norton, J. D. Hopper, Elizabeth J. Feinler, James H. Bair, Robert N. Lieberman, N. Dean Meyer, Sandy L. Johnson, Martin E. Hardy,

ARC 26422

Editing Sample Session II

SRI-ARC

19 SEP 75

Augmentation Research Center

STANFORD RESEARCH INSTITUTE
MENLO PARK, CALIFORNIA 94025

EDITING SAMPLE SESSION II

This is another document belonging to the Secretarial Functions Guide.

EDITING SAMPLE SESSION II

INTRODUCTION

You will want to work on this sample session after you have completed "Editing Sample Session I." In this session you create a first draft of a report and revise it into a more polished form. To accomplish this, the sample session shows you how to perform several editing commands which can be applied generally whenever you want to modify text online.

Throughout this sample session we spell out the sequence of keys you will strike to make something happen and explain to you what the results should be. Keys that do not print, such as carriage return and escape (also called altmode), are named inside angle brackets, e.g. <CR>, <ESC>. <SP> represents a space. The control key <CTRL> is used like the shift key. You hold it down while you type the letter that is after the hyphen. The notation for control key is <CTRL-(some character)>.

Some control keys to remember...

<CTRL-X> aborts commands before you have typed a <CR>.
<CTRL-O> stops printing.
<CTRL-A> deletes the character you have just typed.
<CTRL-W> deletes the word you have just typed.
<CTRL-Q> provides an explanation of the current location in command space.

When you see <CR>, use the return or carriage return on your keyboard.

INSTRUCTION

1. To begin this sample session you first need to create a work space. To do so, create a file called "Editing".

```
You type:
          <SP>crfediting<CR>
You see:
          BASE C: Create C: File T: editing

          <DIRECTORYNAME, EDITING,NLS;1,>
```

2. You first want to transcribe, or write, a draft of the report online. You will "write" your initial draft by using the Insert Statement command and <CTRL-E> (OKINSERT) to put you into the "enter mode".

```
You type:
          is0<CR>This report will cover a few commands
          we've learned thus far.<CTRL-E>
You see:
          BASE C: Insert C: Statement (to follow) A: 0
          L:
          T: This report will cover a few commands we've
          learned thus far.
```

Notice that you have departed slightly from the way you performed this command in "Editing Sample Session I", when you typed a <CR> after the prompt for LEVEL-ADJUST (L:). A faster way is to simply ignore the prompt L: when you do not want to change the level of your statement, and begin typing in your text.

3. Since you ended your last command with the <CTRL-E>, you can now insert a series of statements without repeating the commandwords. There are some intentional errors in the text which you will be inserting. Try to type in exactly what you see so that you can successfully work through all of the later steps in the sample session. When you are finished inserting statements, you leave the "enter mode" by typing a <CTRL-X>.

.....

You type:

Insert: This command allows yu to add, duplicate, or create information in a file. The command Insert Statement was presented in the "Editing Sample Session I." This sample scenario adds Insert Word and Character,<CR>

You see:

L:
T: Insert: This command allows yu to add, duplicate, or create information in a file. The command Insert Statement was presented in the "Editing Sample Session I." This sample scenario adds Insert Word and Character.

.....Continue in this mode,.....
adding the following six statements:

Replace: This command allows you to erase a STRING or STRUCTURE at a specified DESTINATION and put in some other content.

Delete: Delete erases something that you specify, such as a character or statement, from the DESTINATION you specify. This command was introduced in the "Editing Sample Session I."

Copy: The Copy command is used to reproduce a SOURCE (such as STRING or STRUCTURE) at a specified place.

Substitute: The command Substitute allows you to put a new STRING in the place of an old STRING everywhere it appears in the STRUCTURE you specify. Substitute is the most common editing command used on the typewriter terminal.

Move: This command is being introduced in this sample scenario. Move transfers a specified SOURCE (such as STRING or STRUCTURE) to a DESTINATION you specify.

Transpose: Transpose allows you to make STRINGS or STRUCTURES trade places.

.....

4. Now that you've written your first draft, you will want to see it. The quickest way to have your whole file printed is through the Print File command.

.....

You type:

pf<CR>

You see:

BASE C: Print C: File OK:
< DIRECTORYNAME, EDITING,NLS;1, >

.....

Following the origin statement (shown above), appears the contents of your new file. You can now review it for corrections and additions.

5. The first obvious error is in statement 2, where "you" is misspelled. The command Insert Character can correct this error. You type the command with the statement number and the letter "y" for your ADDRESS. The character will be inserted after the last character in your ADDRESS.

.....

You type:

ic2<SP>"y"<CR>o<CR>

You see:

BASE C: Insert C: Character (to follow) A: 2 "y"
T: o

.....

You must always be sure that your ADDRESS is unique to insure that the character is inserted in the right place. In this case "y" was sufficient since there were no other y's in the statement.

6. The next revision you want to make is also in statement 2. To clarify the last sentence, you decide to add the word "Insert" after "and". To do so, use the Insert Word command.

.....

You type:

iw2<SP>"<SP>and"<CR>Insert<CR>

You see:

BASE C: Insert C: Word (to follow) A: 2 " and"
T: Insert

.....

Again, you needed a unique STRING for your ADDRESS. You typed "<SP>and" to distinguish the STRING from the letters "and" in the word "command".

7. Your next correction is in statement 4, where you want to delete the unnecessary word "that". You will use the Delete Word command.

```
You type:
          dw4<SP>"that"<CR><CR>
You see:
          BASE C: Delete C: Word (at) A: 4 "that"
          OK:
```

8. In statements 2 and 7 you want to substitute the word "session" for "scenario". To make both edits simultaneously, execute the Substitute Word (in) Branch command, using your origin statement (statement 0) as your ADDRESS.

```
You type:
          swb0<CR>session<CR>scenario<CR><CR>
You see:
          BASE C: Substitute C: Word (in) C: Branch (at)
          A:0

          (New WORD) T: session

          (Old WORD) T: scenario
          (Finished?) S/Y/N:
          Substitute in Progress

          Substitutions made: 2
```

A word of caution here. You must be careful when you are using the Substitute command, particularly in potentially large STRUCTURES such as a Branch. Since every place the old STRING appears in the specified Branch will be changed, be very sure that you want to change all instances.

9. You would like your report to tell which of the commandwords the user has been introduced to in "Editing Sample Session I." Statements 2 and 4 already include this information, but you have neglected to add it to statement 6. The most efficient way to do this might be to simply copy some appropriate text from statement 2 or 4. The last sentence in statement 4 is suitable.

.....

You type:

```
ct4<SP>"y,<SP>"<CR>4<SP>+e<CR>6<SP>+e<CR>
```

You see:

```
BASE C: Copy C: Text (from) A: 4 "y, "
(through) A: 4 +e
(to follow) A: 6 +e
```

.....

Again, you must be careful to use a unique STRING in the ADDRESS. Notice that this command copies text beginning with the last character in the first ADDRESS STRING. By typing "y,<SP>" you copied the two blank spaces at the end of the sentence, providing proper spacing at the DESTINATION.

10. The next revision is to completely rewrite statement 1, which may seem inadequate to you. You will delete it and put a new one in its place by using Replace Statement.

.....

You type:

```
rs1<CR>This report defines some of the editing
commandwords presented in the "Editing Sample
Sessions I and II."
```

You see:

```
BASE C: Replace C: Statement (at) A: 1
(by) T: This report defines some of the editing
commandwords presented in the "Editing Sample
Sessions I and II."
```

.....

11. The text looks pretty good to you now, so you examine the structure of your report. You decide to reorder your statements according to the commandwords learned in each sample session; the statements defining "insert," "substitute," and "delete" will precede the other statements. First move statement 6 to follow statement 2.

.....

You type:

```
ms6<CR>2<CR><CR>
```

You see:

```
BASE C: Move C: Statement (from) A: 6
(to follow) A: 2
L:
```

.....

Notice that you are prompted for a level. You can ignore the prompt by typing a <CR> and statement 6 will remain on the same level as statement 2.

12. You need to make one more structural change. This time you can simply switch the order of two statements. However, you must remember that the numbers of some of your statements have changed, since you moved statement 6 to follow 2. The former statement 6 is now statement 3; former statement 3 has become 4, etc.

If you think you might become confused you could print your file beginning with statement 3 by using the Jump command and Print Rest (see "Editing Sample Session I"). However, since you have only made one change at this point, this is probably not necessary.

You now want to transpose the statements beginning with "Replace:" and "Delete:". You make a note of their new statement numbers, and are ready to go.

```
.....................................................
You type:
          ts4<CR>5<CR><CR>
You see:
          BASE C: Transpose C: Statement (at) A: 4
          (and) A: 5
          OK:
.....................................................
```

13. You have completed your revisions and would like to print your new version at your terminal. One way of doing this is to Jump to the beginning of your file (the origin statement), and use the Print Rest command.

```
.....................................................
You type:
          jo<CR><CR>
You see:
          BASE C: Jump (to) C: Origin A: V:
.....................................................
```

This command asks for an ADDRESS and allows you to change your viewspecs. You type a <CR> to indicate the ADDRESS of the file you have loaded, and another <CR> to keep the same viewspecs. Now you are ready to print the file.

```
.....................................................
You type:
          pr<CR>
You see:
          BASE C: Print  C: Rest OK:
          <DIRECTORYNAME, EDITING,NLS;1,> , DATE
          TIME IDENT ;;;;
.....................................................
```

This is then followed by the contents of your file.

14. Now that you have completed work on your file you will want to do Update File, a command that creates a "new" version of your file by incorporating into it all of the modifications you have made at this time.

```
.....................................................
You type:
          uf<CR>
You see:
          BASE C: Update C: File OK/C:
          <DIRECTORYNAME, EDITING,NLS;2, >
.....................................................
```

SAMPLE SESSION SUMMARY

Insert Character:

Inserts a character to follow the last character you specified in the ADDRESS.

Insert Word:

Inserts a word to follow the word that contains the last character in the ADDRESS. Insert Word will not break a word in the old text.

Delete Word:

Deletes the word containing the last character in the ADDRESS typein.

Substitute Word:

Substitutes a word in one designated STRUCTURE for another word.

Copy Text:

Copies specified text from a SOURCE to follow the last character in a DESTINATION (the ADDRESS where you want the text to appear).

Replace Statement:

Replaces an existing statement with a new one.

Move Statement:

Transfers a specified statement to a specified ADDRESS.

Transpose Statement:

Changes the order of two designated statements.

Jump (to) Origin:

Moves you to the origin statement (header) of the file you specify.

Print File:

Prints your entire file with default viewspecs without affecting your current viewspecs or your location.

Update File:

Creates a new version of a file by incorporating into it all of the modifications made since its creation or the last update.

EDITING SAMPLE SESSION II

```
(J26422)  18-SEP-75 14:02;;;;   Title:  Author(s): Beverly Boli/BEV;
Distribution: /SRI-ARC( [ INFO-ONLY ] ) DIRT( [ INFO-ONLY ] ) ;
Sub-Collections:  SRI-ARC DIRT; Clerk: BEV;          Origin: < BOLI,
EDIT2-SS,NLS;17, >, 9-SEP-75 12:40 BEV ;;;;
 ####;
```

26422 Distribution

James C. Norton,

N. Dean Meyer, James E. (Jim) White, Douglas C. Engelbart, Martin E. Hardy, J. D. Hopper, Charles H. Irby, Harvey G. Lehtman, James C. Norton, Jeffrey C. Peters, Dirk H. Van Nouhuys, Kenneth E. (Ken) Victor, Richard W. Watson, Don I. Andrews, Jonathan B. Postel, Priscilla A. Wold, Rita Hysmith, Pamela K. Allen, Delorse M. Brooks, Elizabeth F. Finney, Beverly Boli, Lawrence A. Crain, Kirk Sattley, Susan Gail Roetter, Robert N. Lieberman, Ann Weinberg, Kenneth E. (Ken) Victor, Douglas C. Engelbart, James H. Bair, Elizabeth K. Michael, Richard W. Watson, Elizabeth J. Feinler, Harvey G. Lehtman, Kirk E. Kelley, Laura E. Gould, Jeanne M. Beck, Dirk H. Van Nouhuys Susan K. Ocken, Raphael Rom, David C. Smith, Mary Ann Kellan, Buddie J. Pine, Andy Poggio, David L. Retz, Laura J. Metzger, Karolyn J. Martin, Jan A. Cornish, Larry L. Garlick, Priscilla A. Wold, Pamela K. Allen, Delorse M. Brooks, Beverly Boli, Rita Hysmith, Log Augmentation, Joseph L. Ehardt, Raymond R. Panko, Susan Gail Roetter, Robert Louis Belleville, Rene C. Ochoa, Ann Weinberg, Adrian C. McGinnis, Robert S. Ratner, David S. Maynard, Robert N. Lieberman, Sandy L. Johnson, James H. Bair, Jeanne M. Leavitt, Rodney A. Bondurant, Jeanne M. Beck, Marcia L. Keeney, Elizabeth K. Michael, Jonathan B. Postel, Elizabeth J. Feinler, Kirk E. Kelley

ARC 26423

Editing Sample Session III

SRI-ARC

9 SEP 75

Augmentation Research Center

STANFORD RESEARCH INSTITUTE
MENLO PARK, CALIFORNIA 94025

EDITING SAMPLE SESSION III

As explained in comment on Sec. Func. Guide, this is one module.

EDITING SAMPLE SESSION III

INTRODUCTION

"Editing Sample Session III" will be most helpful to you if you work on it after you have completed "Editing Sample Sessions I and II". This session introduces you to file structure. It also illustrates the commands Break and Append Statement, and shows you how to Move STRUCTURES within your file. To accomplish this, you will create and revise a multi-leveled outline.

EDITING SAMPLE SESSION III

INSTRUCTION

1. You will begin this sample session by creating a file for your outline. You give your file the name "MENUE" and are ready to begin inserting statements.

.....

You type:

<SP>crfmenue<CR>

You see:

```
BASE C:  Create C: File T: MENUE
< DIRECTORYNAME, MENUE.NLS;1, >
```

.....

2. To transcribe an outline you will use Insert Statement and <CTRL-E> (OKINSERT). This is the same command you used in the "Editing Sample Session II". This time, however, you will be inserting statements at different levels. Thus you must have a clear idea of the structure of your outline and the relationships of the statements in it to one another.

You specify a LEVEL-ADJUST when prompted by L:. The level which you specify is in relationship to the preceding statement. To change the level of a statement you use either "u" (up) to indicate a higher level, or "d" (down) for a lower level. You can type more than one u to go up more than one level. (For example, to insert a statement three levels above the preceding statement, you would type "uuu" at the prompt L:.) To insert a statement at the same level as the one preceding it, simply ignore the LEVEL-ADJUST and begin typing your text.

For this sample session you will work with the following outline, which is given here in its entirety. The first few commands will be illustrated for you; you will then continue on your own until all statements of the outline are inserted in your file. After you have finished inserting the outline you will find a copy of how your file should look. A reproduction of what you see on your terminal as you copy the outline into your file is also provided. We recommend that you only use this to check your work. It should not be depended upon to create the outline.

EDITING SAMPLE SESSION III

.....OUTLINE TO BE TRANSCRIBED.....

- Appetizers
  - antipasto
  - asparagus vinaigrette
  - escargot
- Soups
  - Cold Soups
    - vichysoisse
    - gazpacho
      - chopped green peppers, tomatoe, and cucumber
  - Hot Soups
    - borscht
      - with sour cream
    - chicken
    - spinach
- Salads
  - caesar
  - bean
  - greek
    - with anchovies
  - tossed green
- Entrees
  - Fish and shellfish
    - sole
    - scallops
      - broiled
      - sauteed
    - lobster
      - 1 lb.
      - 2 lb.
  - Meat
    - filet mignon
    - prime ribs
- Starch
  - potatoes
    - baked
      - with sour cream and chives
      - with melted cheese
    - french fried
    - whipped
    - creamed
  - rice
    - plain
    - pilaf

.....

3. You begin by inserting the first statement of your outline below the origin statement. This will be statement 1 in your outline.

```
You type:
          is<CR>Appetizers<CR>
You see:
          BASE C: Insert C: Statement (to follow) A: L:
          T: Appetizers
```

4. The next statement you insert will be a substatement of statement 1. This will become statement 1A. You specify its level by typing a "d" after the prompt L:.

```
You type:
          is<CR>d<CR>antipasto<CR>
You see:
          BASE C: Insert C: Statement (to follow) A: L: d
          T: antipasto
```

5. The third and fourth statements you will be inserting will also be substatements of statement 1. They will follow statement 1A on the same level, becoming statements 1B and 1C. Use Insert Statement for the third statement, followed by <CTRL-E> (OKINSERT) to continue in the enter mode. Since both statements are on the same level as statement 1A, ignore the LEVEL-ADJUST and keep on typing.

```
You type:
          is<CR>asparagus vinaigrette<CTRL-E>
          escargot<CR>
You see:
          BASE C: Insert C: Statement (to follow) A: L:
          T: asparagus vinaigrette<^E>
          L:
          T: escargot
```

6. You may wish to check your work at this point. Type a <CTRL-X> to take you out of the enter mode, then use the Print File command.

```
You type:
          <CTRL-X>pf<CR>
You see:
          L:
          BASE C: Print C: File OK:
          < DIRECTORYNAME, MENUE.NLS;1, >, DATE TIME
          IDENT ;;;;
          1 Appetizers
             1A antipasto
             1B asparagus vinaigrette
             1C escargot
```

7. Continue to insert statements, adjusting the level of each to the proper level of your outline. Begin again using Insert Statement, followed by <CTRL-E>.

```
You type:
          is1c<CR>u<CR>Soups<CTRL-E>
You see:
          BASE C: Insert C: Statement (to follow) A: 1c
          L: u
          T: Soups<^E>
```

CONTINUE IN THIS MODE UNTIL THE ENTIRE OUTLINE SHOWN ON PAGE 3 IS COPIED INTO YOUR FILE.

8. You should now have the whole outline--from Appetizers to Starch--inserted into your file. To see your file, first type a <CTRL-X> to take you out of the repeat mode, then give the command Print File.

```
You type:
          <CTRL-X>pf<CR>
You see:
          L:
          BASE C: Print C: File OK:
          < DIRECTORYNAME, MENUE.NLS;1, >, DATE TIME
          IDENT ;;;;
```

EDITING SAMPLE SESSION III

```
1 Appetizers
   1A antipasto
   1B asparagus vinaigrette
   1C escargot
2 Soups
   2A Cold Soups
      2A1 vichysoisse
      2A2 gazpacho
         2A2A chopped green peppers, tomatoe,
              and cucumber
   2B Hot Soups
      2B1 borscht
         2B1A with sour cream
      2B2 chicken
      2B3 spinach
3 Salads
   3A caesar
   3B bean
   3C greek
      3C1 with anchovies
   3D tossed green
4 Entrees
   4A Fish and shellfish
      4A1 sole
      4A2 scallops
         4A2A broiled
         4A2B sauteed
      4A3 lobster
         4A3A 1 lb.
         4A3B 2 lb.
   4B Meat
      4B1 filet mignon
      4B2 prime ribs
5 Starch
   5A potatoes
      5A1 baked
         5A1A with sour cream and chives
         5A1B with melted cheese
      5A2 french fried
      5A3 whipped
      5A4 creamed
   5B rice
      5B1 plain
      5B2 pilaf
```

EDITING SAMPLE SESSION III

9. The following is a transcription of what your commands should have looked like as they appeared on your terminal. As stated earlier, use this only to check your work if you find mistakes after you print your file.

.....

COMMAND TRANSCRIPTION

You see:

```
L: d
T: Cold Soups
L: d
T: vichysoisse
L:
T: gazpacho
L: d
T: chopped green peppers, tomatoe, and cucumber
L: uu
T: Hot Soups
L: d
T: borscht
L: d
T: with sour cream
L: u
T: chicken
L:
T: spinach
L: uu
T: Salads
L: d
T: caesar
L:
T: bean
L:
T: greek
L: d
T: with anchovies
L: u
T: tossed green
L: u
T: Entrees
L: d
T: Fish and shellfish
L: d
T: sole
L:
T: scallops
```

```
L: d
T: broiled
L:
T: sauteed
L: u
T: lobster
L: d
T: 1 lb.
L:
T: 2 lb.
L: uu
T: Meat
L: d
T: filet mignon
L:
T: prime ribs
L: uu
T: Starch
L: d
T: potatoes
L: d
T: baked
L: d
T: with sour cream and chives
L:
T: with melted cheese
L: u
T: french fried
L:
T: whipped
L:
T: creamed
L: u
T: rice
L: d
T: plain
L:
T: pilaf
```

10. An inital version of the outline now exists in your file. The remainder of the sample session will illustrate some commands to change the structure of your outline. The first task will be to add another statement to the outline. You will use the Insert Statement command to add a substatement to statement 2b2.

```
.....................................................
You type:
          is2b2<CR>d<CR>with dumplings<CR>
You see:
          BASE: Insert C: Statement (to follow) A: 2b2 L: d
          T: with dumplings
.....................................................
```

11, You would like part of statement 5A1A to be set off as a separate statement, Break Statement allows you to divide a statement into two, It will break at the next space after the last character you specify in the ADDRESS, You may also specify the level of the second statement relative to the first one, You will want to make your second statement a substatement of the first, and will indicate this by typing a "d" after the prompt L:,

```
.....................................................
You type:
          bs5a1a<SP>"cream"<CR>d<CR>
You see:
          BASE C: Break C: Statement (at) A: 5a1a "cream"
          L: d
.....................................................
```

By breaking statement 5A1A and putting the new statement on a lower level, you have created a new Branch, You can see it by using Print Branch,

```
.....................................................
You type:
          pb5a1a<CR><CR>
You see:
          BASE C: Print C: Branch (at) A: 5a1a V:
             5A1A with sour cream
                5A1A1 and chives
.....................................................
```

12, In this step you will reverse the process carried out in the previous step, Using statements 2B1 and 2B1A, you will attach one statement to another with the Append Statement command, The appended statement is added to the end of the receiving statement, and is joined with what you type when prompted for CONTENT, You will type a <SP> where you are prompted for CONTENT, since you do not need to add any additional text for the new statement to make sense,

.....

You type:

```
as2b1a<CR>2b1<CR><SP><CR>
```

You see:

```
BASE C: Append C: Statement (at) A: 2b1a
(to) A: 2b1
(join with) T:
```

.....

To see the appended statement, type a \.

13. As stated above, you may insert some text to join together two statements when you use the Append Statement command. This will be illustrated by appending statement 2A2A to 2A2.

.....

You type:

```
as2a2a<CR>2a2<CR><SP>with<SP><CR>
```

You see:

```
BASE C: Append  C: Statement (at) A: 2a2a
(to) 2a2
(join with) T:  with
```

.....

To see the appended statement, type a \.

14. Thus far most of the structures you have been handling in your commands have been statements. In this step you will use the Move Group command. A group is defined as a series of consecutive statements (and their substructure) at the same level. You will Move Group 5A2-5A4 to follow statement 5A. Don't forget to indicate the proper level.

.....

You type:

```
mg5a2<CR>5a4<CR>5a<CR>d<CR>
```

You see:

```
BASE C: Move C: Group (from) A: 5a2
(through) A: 5a4
(to follow) A: 5a
L: d
```

.....

If you want to see the new structure, type Print Branch, using branch 5A as your ADDRESS.

15. The next command uses Branch (a statement plus all its substatements, all their substatements, and so on). You will Move Branch 4B (which includes the substatements 4B1 and 4B2) to follow statement 4.

.....

You type:

```
mb4b<CR>4<CR>d<CR>
```

You see:

```
BASE C: Move C: Branch (from) A: 4b
(to follow) A: 4
L: d
```

.....

To see the new structure, type Print Branch, using branch 4 as your ADDRESS.

16. In this step you will again use Group and Branch, this time to Delete portions of your outline.

.....

You type:

```
dg3b<CR>3c<CR><CR>
```

You see:

```
BASE C: Delete C: Group (at) A: 3b
(through) A: 3c
OK:
```

.....

Again, if you want to see the new structure use the Print Branch command. Now try the Delete Branch command.

.....

You type:

```
db1<CR><CR>
```

You see:

```
BASE C: Delete C: Branch (at) A: 1
OK:
```

.....

Notice that you have been working from the end of your outline toward the beginning. This is the most efficient approach when you are making several structural changes at once, since the statement numbers at the beginning of the file do not change as you make changes at the end. Thus you can work from one initial copy without constantly checking revised statement numbers preceding your changes.

Another approach to addressing for making structural changes also avoids the inconvenience caused by statement number revision. You can identify your statements by SIDs rather than statement numbers, since the SID for a statement remains unchanged as long as that statement exists. See the "File-Viewing Sample Session" or use the Help command to learn how to make the system number statements with SIDs.

17. You can now use the Print File command to see your revised outline.

```
.....................................................
You type:
          pf<CR>
You see:
          BASE C: Print C: File OK:
          < DIRECTORYNAME, MENUE,NLS;1, >, DATE TIME
          IDENT;;;;
.....................................................
```

This is then followed by your revised outline, beginning with the new statement 1 "Soups".

18. You have completed this sample session and are familiar with some of the commands useful in creating and modifying multi-leveled outlines online. You may now wish to delete this file from your Directory.

```
.....................................................
You type:
          dfmenue<CR><CR>
You see:
          BASE C: Delete C: File T: menue
          OK:

          Deleted Files Are:
             <DIRECTORYNAME, MENUE,NLS;1,> and its
             partial copy
.....................................................
```

SAMPLE SESSION SUMMARY

LEVEL-ADJUST:

Prompted by L:, asks you to specify the level you wish statements to occupy relative to where you pointed. If you want to change the level, type a lowercase d for down, or a lowercase u for up. You can type more than one u to go up more than one level. If you don't want to change the level, ignore the LEVEL-ADJUST and start typing in your text.

Break Statement:

Allows you to divide a statement into two. It will break at the next space after the last character you specify in the ADDRESS. You may also specify the level of the second statement relative to the first one.

Append Statement:

Attaches one statement to another. The appended statement is added to the end of the receiving statement. You may join the two statements by typing something when prompted for CONTENT.

Branch:

A statement plus all its substatements, all their substatements, and so on.

Group:

A series of consecutive statements (and their substructure) at the same level.

Print Branch:

Prints at your terminal the branch you specify when prompted for an ADDRESS.

Print File:

With your current viewspecs, prints at your terminal the entire file you have loaded without affecting your current location.

Move Group:

Moves a Group from a SOURCE to follow a DESTINATION.

Move Branch:

Moves a Branch from a SOURCE to follow a DESTINATION.

Delete Group:

Erases a specified Group from your file.

Delete Branch:

Erases a specified Branch from your file.

Delete File:

Removes the file named for CONTENT from normal use. To delete the file you have loaded, type a <CR> for CONTENT. If no directory is named, it assumes a file in your own directory.

EDITING SAMPLE SESSION III

```
(J26423) 9-SEP-75 20:11;;;;  Title:  Author(s): Beverly Boli/BEV;
Distribution: /SRI-ARC( [ INFO-ONLY ] ) DIRT( [ INFO-ONLY ] ) ;
Sub-Collections:  SRI-ARC DIRT; Clerk: BEV;          Origin: < BOLI,
EDIT3,NLS;16, >, 9-SEP-75 12:43 BEV ;;;;
 ####;
```

26423 Distribution

Douglas C. Engelbart, Martin E. Hardy, J. D. Hopper, Charles H. Irby, Harvey G. Lehtman, James C. Norton, Jeffrey C. Peters, Dirk H. Van Nouhuys, Kenneth E. (Ken) Victor, Richard W. Watson, Don I. Andrews, Jonathan B. Postel, Priscilla A. Wold, Rita Hysmith, Pamela K. Allen, Delorse M. Brooks, Elizabeth F. Finney, Beverly Boli, Lawrence A. Crain, Kirk Sattley, Susan Gail Roetter, Robert N. Lieberman, Ann Weinberg, Kenneth E. (Ken) Victor, Douglas C. Engelbart, James H. Bair, Elizabeth K. Michael, Richard W. Watson, Elizabeth J. Feinler, Harvey G. Lehtman, Kirk E. Kelley, Laura E. Gould, Jeanne M. Beck, Dirk H. Van Nouhuys, James C. Norton,
David C. Smith, Mary Ann Kellan, Buddie J. Pine, Andy Poggio, David L. Retz, Laura J. Metzger, Karolyn J. Martin, Jan A. Cornish, Larry L. Garlick, Priscilla A. Wold, Pamela K. Allen, Delorse M. Brooks, Beverly Boli, Rita Hysmith, Log Augmentation, Joseph L. Ehardt, Raymond R. Panko, Susan Gail Roetter, Robert Louis Belleville, Rene C. Ochoa, Ann Weinberg, Adrian C. McGinnis, Robert S. Ratner, David S. Maynard, Robert N. Lieberman, Sandy L. Johnson, James H. Bair, Jeanne M. Leavitt, Rodney A. Bondurant, Jeanne M. Beck, Marcia L. Keeney, Elizabeth K. Michael, Jonathan B. Postel, Elizabeth J. Feinler, Kirk E. Kelley, N. Dean Meyer, James E. (Jim) White

ARC 26424

File-Viewing Sample Session

SRI-ARC

9 SEP 75

Augmentation Research Center

STANFORD RESEARCH INSTITUTE
MENLO PARK, CALIFORNIA 94025

FILE-VIEWING SAMPLE SESSION

INTRODUCTION

The "File-Viewing Sample Session" illustrates a variety of ways to see and print your files and Journal mail. Some of the most frequently used viewspecs are introduced and the Print and Jump commands are explored further. (Note: This sample session covers some of the commands taught in the second TNLS Course, "Introduction to Structure and Viewing", and adds some alternative ways of doing things. You may also want to refer to "Editing Sample Sessions I and II".) You will find it useful to be at a typewriter terminal, typing in the commands and text as the sample session describes them.

FILE-VIEWING SAMPLE SESSION

INSTRUCTION

1. You will begin this sample session by loading a file named Vacation which you will view in later steps. Use the Load File command (see "Editing Sample Session I").

.....

You type:

lfuserguides,vacation,<CR>

You see:

BASE C: Load C: File T: userguides,vacation,

< USERGUIDES, VACATION.NLS;1, >

.....

2. The online system you are using allows you to "view" your files in several different ways. The appearance of your file is controlled by single-letter codes called viewspecs. Some viewspecs are set automatically when you log in. These are called the default viewspecs. To see a list of the viewspecs currently in force, use the Show Viewspecs (status) command.

.....

You type:

<SP>shv<CR>

You see:

BASE C: Show C: Viewspecs (status) OK:
levels: ALL, lines: ALL, hjmpuzACEHJLP

.....

You can use the Show Viewspecs command at any time. If you have just logged in, or have been working for a while but have not changed your viewspecs since logging in, you will see exactly what is shown above. These are the default viewspecs. Viewspec w means "levels: ALL, lines: ALL". Some of the other default viewspecs you will be working with in this sample session are listed below with their values:

m - statement numbers/SIDs on
z - blank lines between statements off
H - statement numbers/SIDs left
J - show statement numbers, not SIDs

For more information about the other default viewspecs, see the "TNLS-8 Quick Reference" card, or use the Help command.

Notice that some viewspecs are lowercase letters while others are uppercase. Uppercase viewspecs do different things than do lowercase viewspecs.

3. To see your file with the default viewspecs use the Print File command.

```
You type:
          pf<CR>
You see:
          BASE C: Print C: File OK:
          < USERGUIDES, VACATION,NLS;1, >, DATE TIME
          IDENT ;;;;
```

The header is then followed by the rest of the file, each statement beginning with a statement number.

4. One way to change your viewspecs is to use the Set Viewspecs command to enter new codes. In this step you will alter the appearance of your file a great deal by setting viewspec x, which shows one line and one level only. Follow the Set Viewspecs command with Print File.

```
You type:
          <SP>sevx<CR>pf<CR>
You see:
          BASE C:  Set C: Viewspecs V: x

          BASE C: Print C: File OK:
          < USERGUIDES, VACATION,NLS;1, >...
          1 This report will describe three classifica...
```

Since you have only one top level statement in this file, viewspec x shows you only one line in your entire file.

5. There are other ways to change your viewspecs. Some commands include the prompt "V:" which allows you to type in new viewspecs. Print STRUCTURE is one of those commands. This time you will type in the viewspecs "dt". Notice their effect.

```
You type:
          pb0<CR>dt<CR>
You see:
          BASE C: Print C: Branch (at) A: 0
          V: dt

          < USERGUIDES, VACATIONS,..
          1 This report will describe three,..
```

The two viewspecs d and t have the same effect as viewspec x: viewspec d shows one level only, and t one line only.

6. This step will experiment with different "clipping" viewspecs--viewspecs that cut off lines or levels. Begin with viewspecs b and r (show one level more; show one line more). You can use more than one viewspec b or r to indicate more than one additional line or level. To see how this works, type in the viewspecs "brr" when prompted in the Print Branch command. You will see one more level and two more lines, or a total of two levels and three lines for each statement. (Statements 1A through 1C will, of course, each print out as a single line.)

```
You type:
          pb<CR>brr<CR>
You see:
          BASE C: Print C: Branch (at) A: 0
          V: brr

          < USERGUIDES, VACATION,..
          1 This report will describe three,..
          MA, and PP), defining and setting,..
          hoped that this document will aid,..
             1A AOE (All-Out Extravaganzas)
             1B MA (Moderate Adventures)
             1C PP (Penniless Pilgrimages)
```

Viewspecs a and q show one level less and one line less.

.....

You type:

pb0<CR>aq<CR>

You see:

```
BASE C: Print C: Branch (at) A: 0
V: aq

< USERGUIDES, VACATION...
1 This report will describe three...
MA, and PP), defining and setting...
```

.....

As with viewspecs b and r, you can use more than one viewspec a or q to indicate fewer levels or lines (for example, typing "qq" in this step would have shown two lines less, leaving only one line to be printed).

7. Viewspecs c and t are the last set of clipping viewspecs with which you will experiment. Viewspec c will show all levels, while viewspec t will show first lines only, as illustrated in step 2. This combination of viewspecs provides a quick skeleton view of your file structure.

.....

You type:

pb0<CR>ct<CR>

You see:

```
BASE C: Print C: Branch (at) A: 0
V: ct

< USERGUIDES, VACATION...
```

.....

The header is followed by the contents of your file with all levels showing, but only one line of each level.

8. You will work with viewspecs that number your statements in this step. As you learned in step 2, the default viewspecs for statement numbering are m, J, and H. This means statement numbers are turned on and will appear on the left side. You will first change both the kind of numbering and the side on which it is to appear.

```
You type:
          pb1b<CR>IG<CR>
You see:
          BASE C: Print C: Branch (at) A: 1b
          V: IG

          MA (Moderate Adventures)              014
             Definition:                        015
          *****
```

This is followed by the remainder of the branch, showing SIDs on the right side following each statement. Note that only one line is showing for each statement since viewspec t is still in force.

9. You may use SIDs in your address rather than statement numbers. Try Print Statement (which also prompts you with V:) and change back to statement numbers.

```
You type:
          ps014<CR>J<CR>
You see:
          BASE C: Print C: Statement (at) A: 014
          V: J

          MA (Moderate Adventures)                1B
```

Although you are now seeing statement numbers, each statement can still be identified by its SID. Use Print Statement again, with an SID address, and move the statement number back to the left side.

```
You type:
          ps015<CR>H<CR>
You see:
          BASE C: Print C: Statement (at) A: 015
          V: H

             1B1 Definition
```

10. Another way of printing a statement is by using the Linefeed command, which prints the next statement. You execute this command by typing the Linefeed key on your terminal or <CTRL-J>. The notation for the Linefeed key is <LF>.

In this step you will change your viewspecs so that you can see all lines, then print the statement following statement 015.

.....

You type:

<SP>sevs<CR><LF>

You see:

```
BASE C:  Set C: Viewspecs V: s
BASE C:

      1B1A Usually limited to one-two...
      a budget of one-two weeks salary....
      frequently in the form of a check ...
      encourage visiting.
```

.....

11. The command Reset Viewspecs allows you to change the viewspecs back to your initial set for the session in which you are working. Typing Reset Viewspecs now will bring the default viewspecs back into force.

.....

You type:

<SP>resv<CR>

You see:

```
BASE C:  Reset C: Viewspecs OK:
```

.....

If you would like to see your viewspecs, use the Show Viewspecs command. The list that prints out will be identical to that which you saw in step 2.

12. You will use a viewspec with which you are already familiar to begin this step.

```
.....................................................
You type:
          pb1c<CR>x<CR>
You see:
          BASE C: Print C: Branch (at) A: 1c
          V: x

          1C PP (Penniless Pilgrimages)
.....................................................
```

To change the effects of viewspec x back to all lines and all levels, use the default viewspec w. Viewspec n will turn off the statement numbers. Also try viewspec y, which turns on blank lines between statements.

```
.....................................................
You type:
          pb1c<CR>wny<CR>
You see:
          BASE C: Print C: Branch (at) A: 1c
          V: wny
.....................................................
```

Branch 1C then follows, with all lines and levels showing and blank lines between the statements. Now try the default viewspec z, which turns off lines between statements.

```
.....................................................
You type:
          pb1c<CR>z<CR>
You see:
          BASE C: Branch (at) A: 1c
          V: z
.....................................................
```

You will now see the same Branch 1C without blank lines between statements.

13. The rest of this sample session will explore some ways to move among files. First try the Jump to File Return command. This will take you back to the file you had loaded before this one. If you had just logged in, you will be back at your initial file.

.....

You type:

jfr<CR><CR>

You see:

BASE C: Jump (to) C: File C: Return OK:
"< DIRECTORY, FILENAME >" Y/N: OK:
< DIRECTORY, FILENAME >

.....

Notice that you are prompted by Y/N in the command. This gives you the option to return to other files you have had loaded in this work session. If you had typed "n" rather than a <CR> (or "y"), the name of the file you had been working in previous to the one you have now returned to would have appeared, and so on.

14. There is another way to go from the file you have loaded back to the previous one. You can use the Print STRUCTURE command and give ",fr" (file return) for your address. Try it here with the Print Statement command. This will take you back to statement 1C in USERGUIDES, VACATION, since that's where you were last working in that file.

.....

You type:

ps,fr<CR><CR>

You see:

BASE C: Print C: Statement (at) A: ,fr
V:

PP (Penniless Pilgrimages)

.....

Note that you were given the opportunity to change your viewspecs in this command. By typing a <CR> you left them as they were when you were last in that file.

15. The Print Branch command also allows you to move to another file when your address allows for it. In this step you will go to your initial file and print your first Journal citation. (If you have not yet received any Journal mail, come back to complete this sample session when you have. If you do not read your Journal mail, you can stop here.)

For your address, use the FILENAME (IDENT,) followed by the word "journal", and the notation ",n" to indicate the next branch. This will print the first citation in your Journal. Set viewspec m to see the statement number.

.....

You type:

pbIDENT,journal<SP>,n<CR>m<CR>

You see:

```
BASE C: Print C: Branch A: IDENT,journal ,n
V: m

   1A IDENT DATE TIME JOURNAL FILE NUMBER
   TITLE
   Location: (Journal, FILE, STATEMENT NUMBER:
             VIEWSPECS)
   *****Note: [ACTION]*****

   < FILENAME >
```

.....

Your journal citation will not look like the one shown here, but the two should be very similar.

16. This step will explore the Jump to Address command using an address element called a link to read your mail.

A link is a string of characters in a statement that names the address of any location in any NLS file (optionally with any viewspecs). Links are surrounded by delimiters: parentheses () or angle-brackets <>. The link you will be using here is in your journal citation, shown above. Notice that there is a string of characters in parentheses following the word "Location:". This is the link to this journal item. When typing in the link, spaces are optional.

.....

You type:

jaDIRECTORY,FILE,STATEMENT NUMBER:VIEWSPEC<CR>

You see:

```
BASE C: Jump (to) C: Address A: DIRECTORY,
FILE,STATEMENT NUMBER:VIEWSPEC

< DIRECTORY, FILE >
```

.....

To see this journal item you can use the Print File command unless the journal item is less than 1000 characters. (If this is the case, the item will already have appeared when you printed the branch in your initial file.) You may do this now if you wish. Then return to your initial file with the Jump File Return command (see step 11) and you can practice a faster way of using links with the Jump to Address command.

17. This time you will use a different ADDRESS when prompted by A:. You cite the location of the link (in this case statement 1A), followed by ",1". This indicates that you want to jump to the link shown in that statement.

.....

You type:

jaia,1<CR>

You see:

BASE C: Jump (to) C: Address A: 1a,1

< DIRECTORY, FILE >

.....

You may log out at this point, Jump File Return back to your initial file, Load another file, or Goto a subsystem.

FILE-VIEWING SAMPLE SESSION

SAMPLE SESSION SUMMARY

Viewspecs:

Single letter codes that control the appearance or "view" of your files. When viewspecs are allowed in a command you are prompted by a "V:". You may type a string of any of the viewspec codes, terminated by OK. Type just an OK if you don't want to change the viewspecs. Uppercase viewspecs do different things than lowercase viewspecs.

Viewspecs used in this sample session:

x - show one line and one level only
w - show all lines and all levels (default)
c - show all levels
d - show first level only
s - show all lines
t - show first lines only
a - show one level less
b - show one level more
r - show one line more
q - show one line less
m - statement numbers/SIDs on (default)
n - statement numbers/SIDs off
I - show SIDs not statement numbers
J - show statement numbers, not SIDs (default)
G - statement numbers/SIDs right
H - statement numbers/SIDs left (default)
y - blank lines between statement on
z - blank lines between statements off (default)

Show Viewspecs:

Lists the viewspecs in force in the current work session.

Set Viewspecs:

Allows you to change the viewspecs at anytime for the current work session.

Reset Viewspecs:

Sets the viewspecs back to your initial set for the current work session.

FILE-VIEWING SAMPLE SESSION

Print Statement:

Prints at your terminal the statement you specify when prompted for an ADDRESS.

Jump File Return:

Moves you to a file where you were before. ("<PAST FILEADDRESS>") is the name of the file you will go to if you answer Yes or type <CR>. If you answer No, the FILEADDRESS before that will appear.

Jump Address:

Moves you to the statement you specify (optionally with viewspecs).

Link:

A string of characters in a statement that names the ADDRESS of any location in any NLS file (optionally with any view). Links are surrounded by delimiters in the order and format: < ADDRESS : VIEWSPECS >. Spaces are optional. Delimiters may be parentheses () or angle-brackets <>.

INFILEADDRESS - POSITION:

A character preceded by a period. Addresses a POSITION in and among files. Moves you in relation to your current location in the direction that corresponds to the character you type.

.fr - file return
.n - next
.l - link

FILE-VIEWING SAMPLE SESSION

(J26424) 9-SEP-75 20:15;;;; Title: Author(s): Beverly Boli/BEV;
Distribution: /SRI-ARC([INFO-ONLY]) DIRT([INFO-ONLY]) ;
Sub-Collections: SRI-ARC DIRT; Clerk: BEV; Origin: < BOLI,
VIEWING,NLS;14, >, 26-AUG-75 19:44 BEV ;;;;
####;

26424 Distribution

Douglas C. Engelbart, Martin E. Hardy, J. D. Hopper, Charles H. Irby, Harvey G. Lehtman, James C. Norton, Jeffrey C. Peters, Dirk H. Van Nouhuys, Kenneth E. (Ken) Victor, Richard W. Watson, Don I. Andrews, Jonathan B. Postel, Priscilla A. Wold, Rita Hysmith, Pamela K. Allen, Delorse M. Brooks, Elizabeth F. Finney, Beverly Boli, Lawrence A. Crain, Kirk Sattley, Susan Gail Roetter, Robert N. Lieberman, Ann Weinberg, Kenneth E. (Ken) Victor, Douglas C. Engelbart, James H. Bair, Elizabeth K. Michael, Richard W. Watson, Elizabeth J. Feinler, Harvey G. Lehtman, Kirk E. Kelley, Laura E. Gould, Jeanne M. Beck, Dirk H. Van Nouhuys, James C. Norton,
David C. Smith, Mary Ann Kellan, Buddie J. Pine, Andy Poggio, David L. Retz, Laura J. Metzger, Karolyn J. Martin, Jan A. Cornish, Larry L. Garlick, Priscilla A. Wold, Pamela K. Allen, Delorse M. Brooks, Beverly Boli, Rita Hysmith, Log Augmentation, Joseph L. Ehardt, Raymond R. Panko, Susan Gail Roetter, Robert Louis Belleville, Rene C. Ochoa, Ann Weinberg, Adrian C. McGinnis, Robert S. Ratner, David S. Maynard, Robert N. Lieberman, Sandy L. Johnson, James H. Bair, Jeanne M. Leavitt, Rodney A. Bondurant, Jeanne M. Beck, Marcia L. Keeney, Elizabeth K. Michael, Jonathan B. Postel, Elizabeth J. Feinler, Kirk E. Kelley, N. Dean Meyer, James E. (Jim) White