Journal Accession Number: 33874

THE BASIC TNLS-8 COURSE OUTLINE

SRI-ARC

18 FEB 76

Applications Development

Augmentation Research Center
Stanford Research Institute
Menlo Park, California 94025

THE BASIC TNLS-8 COURSE OUTLINE

INTRODUCTION TO NLS

NLS = oN Line System

TNLS = Typewriter Version

CAPABILITIES OF SYSTEM:

Composing

Editing

Studying

Structuring

Browsing - viewing

Printing

Publishing

Communicating -

Sending and receiving mail, messages, documents; teleconferencing; etc.

Storing and retrieving -

Record keeping, library services, data bases, searching, etc.

Calculating

SOME NOTES

This is designed for use when terminals are available for all participants to use/view easily. It is intended to be the first course a person receives on NLS. The commands are shown as they would appear with partial prompting.

COURSE ORGANIZATION

The course is organized by concepts of what a user can do with TNLS at this level. The seven concepts [listed below] are ordered as one would need them to use the system. Under each concept are the exact commands that instruct the computer to perform the function that goes with the concept. There is a command summary at the end of the course outline that lists the same commands alphabetically for easy reference.

The commands which are included in this first course have been selected to let a user write, edit, store, and communicate typewritten information [text]. Those items labeled [Optional] can be covered on the second day of the course.

TNLS CONCEPTS: [Things you can do as covered in this course:]

1. FILES FOR STORAGE
2. TYPING IN INFORMATION
3. TYPING OUT INFORMATION
4. EDITING
5. COMMUNICATING
6. ADDRESSING
7. TROUBLE SHOOTING AND HELP

DEFINITIONS FOR THE COURSE OUTLINE

COMMANDS: You typing some characters to tell the computer what to do.

<SP> means strike the SPACE bar.

Upper case and underlined characters in a TNLS command phrase are what you type.

Words or characters that are uppercase and underlined represent something appropriate that you type.

TYPEIN = a string of characters from the keyboard, ending with an OK.

OK: means you type a carriage return, indicated by a CR.

CR: means you strike the carriage return key; OK.

WHERE YOU ARE: Where the computer thinks you are pointing [to some character in some file]; you tell it by specifying an address; this is where your command will be done.

CTRL means you hold down the control [CTRL] key while typing the specified character.

BASE C: is the TNLS ready signal. It means that you can type in an editing or file-handling command [like home base...].

SEND C: is the Sendmail subsystem ready signal. It means that you can type in a Sendmail command.

GETTING TO NLS

THE TERMINAL AND USE [if dialing in]

If dialing in, turn on, dial TIP number, place receiver in cradle after hearing tone, and make sure terminal is online.

NETWORK [if used]

Net login: brief summary for dial in users [steps are numbered]

[I] Type E [to get the Network's attention]

[II] Type @ O <SP> 43 CR
[to Open a connection to Office-1, Host 43; Host 49 for BBNB]

You now should be connected to TENEX
[@ is the TENEX ready signal]

TENEX EXECUTIVE

Login procedure:

[III] Type LOG <SP> USERNAME <SP> PASSWORD <SP> CR
[the last SPACE fills in the account number automatically; you're then ready to call NLS]

[Optional] Group allocation quota:
GROUPSTAT CR [to see who else is using your slot]

Calling NLS:

[IV] Type NLS CR

[Optional] To get back to TENEX:

BASE C: Goto (subsystem) C: Tenex CR
[you may continue in NLS by typing QUIT CR]

To leave the system, logout in NLS:

BASE C: <SP> Logout OK: CR

To close the network connection:

@ C CR

BASIC TNLS:

Abort command = CTRL X [kills the command before the final CR]

1. FILES

The initials file

The initials file is automatically your first file, and is named after your initials [it is also one of your mailboxes]

The origin statement

The origin statement contains the file name and other information about the file for the system. It should not be edited. It is numbered 0, but no number will be printed.

Creating new files -- use the Create File command

```
BASE C: <SP>CReate C: File T: FILENAME CR
< DIRECTORY, FILENAME.NLS;1, >
```

[a filename can be any short "word", like a folder label]

[Optional] To see a list of all your files use the Show Directory command:

```
BASE C: <SP>SHow C: Directory (of) T/OK: CR
OK: CR OK:
```

To work in another file:

```
BASE C: Load C: File T: FILENAME CR
```

2. TYPING IN INFORMATION

Insert Statement - lets you enter sentences, headings, or paragraphs

```
BASE C: Insert C: Statement (to follow) A: ADDRESS CR
L: CR
T: TYPEIN CR
```

[ADDRESS means statement number; TYPEIN means you type whatever you want in the statement]

[Optional] Continue to insert by typing a CTRL E instead of the first CR for a final OK:

[puts you in the "Enter mode" -- you type CR
to end each statement, when you are finished
type a CTRL X after the prompt L:]

Backspace: You may backspace while typing to immediately correct errors

backspace character = CTRL A
[the deleted character will be printed after a slash]
backspace word = CTRL W
[an underline or backarrow will be printed]

[Optional] To insert text at the end of a statement:

BASE C: Insert C: Text (to follow) A: STATEMENT NUMBER +e CR
T: TYPEIN CR

[+e means the end of the statement]

3. TYPING OUT INFORMATION *To stop printing type a CTRL O
[it takes some time to get through!!]

Printing the file:

BASE C: Print C: File OK: CR

Printing a statement:

BASE C: Print C: Statement at A: STATEMENT NUMBER CR
V: CR

Printing the rest of the file:

BASE C: Print C: Rest OK: CR

[Optional] Easy print: A shortcut to print the current statement

BASE C: \ [backslash prints the statement where you are]

4. EDITING

To change information that has been typed in:

To change information within a statement:

Substitute Text in Statement [to correct most errors]:

```
BASE C: Substitute C: Text (in) C: Statement (at) A: ADDRESS
CR
(New TEXT) T: TYPEIN CR
(Old TEXT) T: TYPEIN CR Finished? S/Y/N: Y [for yes]
Substitutions made: Number
```

[replaces the old text with the new text every time it occurs in the statement]

To change whole statements:

Delete Statement

```
BASE C: Delete C: Statement (at) A: ADDRESS CR
OK: CR
```

[Optional] Move Statement:

```
BASE C: Move C: Statement (from) A: ADDRESS CR
(to follow) A: ADDRESS CR
L: CR
```

[Optional] Copy Statement:

```
BASE C: Copy C: Statement (from) A: ADDRESS CR
(to follow) A: ADDRESS CR
L: CR
```

To Delete a File

```
BASE C: Delete C: File T: FILENAME CR
OK: CR
```

[Careful, this removes all versions of the file. You can Undelete a File anytime before Logout with the Undelete command.]

To Update a File: [should be done periodically to save changes for backup, not imperative]

BASE C: Update C: File OK/C: CR
<DIRECTORY, FILENAME.NLS;2,>

[Optional] Formatting Technique: use carriage returns to control line length

To type a carriage return within a TYPEIN, type CTRL V CR

5. [Optional] COMMUNICATING

SENDMAIL SYSTEM: How to Send Journal Mail

Example of submitting a message using idents [or .lastname] and Interrogate [where the system prompts you]:

BASE C: Goto (subsystem) C: Sendmail OK: CR

SEND C: Interrogate OK: CR

(distribute for action to:) T: JMB,FEEDBACK,SGR CR

(distribute for information-only to:) T: JCN CR

(title:) T: Your Example CR

(type of source:) C: Message T: TYPEIN CR

(show status?) Y/N: Y [the status typed by the system:]

TITLE: Your Example
AUTHOR(S): jhb
DISTRIBUTE FOR ACTION TO: jmb feedback sgr
DISTRIBUTE FOR INFO-ONLY TO: jcn
MESSAGE: [Typein of message will be repeated]

(send the mail now?) Y/N: Y [for yes]

Completed

SEND C: Quit OK/C: CR

To send a Statement that's already stored online, use the following instead of Message for the type of source in the Interrogate command [see the Command Summary for an example]

(type of source:) C: Statement A: ADDRESS CR

To send a File use the following instead of Message [see the Command Summary for an example]

(type of source:) C: File A: FILENAME CR

How to read journal mail you've received

The mail box is in your initials file under a statement called "(Journal)"

Use the Print Journal command

BASE C: Print C: Journal (mail) OK: CR

To empty your mail box: substitute (read) for (journal)

[this will empty your mailbox by having all journal mail that is already read put in a different place from incoming mail. A new journal mailbox will be created whenever there is mail to be delivered. This is a temporary method for the present course level]

TENEX ways of communicating: SNDMSG and LINK

To send a Message in TENEX

BASE C: Goto (subsystem) C: Tenex OK: CR
[must QUIT]:

@SNDMSG CR [the system will prompt you:]
(To (? for help):) TYPEIN CR [lastnames separated by comma]
(cc (? for help):) TYPEIN CR [lastnames separated by comma]
(Subject:) TYPEIN CR [subject of your message]
(Message ? for help): TYPEIN
CRTL Z [to terminate and send the message]
(Q, S, ?, carriage return:) CR [to send the message]

@QUIT CR [to go back to where you were in TNLS]

To read a Message in TENEX

MESS CR

In some systems you must use the readmail command:

READMAIL CR CR

Linking [in TENEX] [first ask where the person is:]

BASE C: Quit OK/C: Nls OK: CR
WHERE <SP> USERNAME CR [do not link when user is in SNDMSG, OUTPRC, NOUTPRC, or XLIST]

LINK <SP> USERNAME CR [precede comment with a ";" and end with a CR; repeat every 3 lines]
BYE CR [to break the link]
CON CR [returns you to NLS]

6. ADDRESSING

Where the control marker is -- type a / [see the questionmark key on some terminals]

This will show an arrow pointing to the character that you are at: ==>x

To move the control marker within your current file use the Jump to Address command

BASE C: Jump (to) C: Address A: ADDRESS CR

An address can be one of the following anytime you see the prompt A:

STATEMENT NUMBER [NOTE: TNLS automatically renumbers statements when appropriate]

BASE C: Jump (to) C: Address A: STATEMENT NUMBER CR

[Optional] "TYPEIN" To find some word or text, enclose whatever word or series of characters you want to find in quotes:

BASE C: Jump (to) C: Address A: "TYPEIN" CR

[takes you to the first occurrence of the TYPEIN to the right and down in your file]

[Optional] STATEMENT NUMBER and "TYPEIN"; To find some word or text starting in a particular statement use:

BASE C: Jump (to) C: Address A: STATEMENT NUMBER "TYPEIN" CR

[Optional] STATEMENT NUMBER .t ("tail"); To find the last statement in the file. Use this address to add statements to the end of your file.

BASE C: Jump (to) C: Address A: STATEMENT NUMBER .t CR

Addressing across files and directories

To address another file:

BASE C: Jump (to) C: Address A: FILENAME, CR

To address another person's file:

BASE C: Jump (to) C: Address A: DIRECTORY,FILENAME, CR

To address a particular statement in another person's file:

BASE C: Jump (to) C: Address A: DIRECTORY,FILENAME,STATEMENT NUMBER CR

7. TROUBLE SHOOTING AND HELP

Immediate assistance from the system:

Type ? for a list of all the possible command words.

[Optional] Status commands

Two CTRL T's [note words RUNNING or WAIT -- system should be either running or waiting for you]

Call SRI-ARC, [415 326-6200, ext. 3630]
Or link to Feedback, Bair, Roetter, or Beck at Office-1

[Optional] FEEDBACK mechanism: [any complaints, questions, problems, suggestions]

SNDMSG or Sendmail to FEEDBACK
Response should be no later than 1 working day

[Optional] Remedies

CTRL C, RESET CR, NLS CR
[use CTRL C only in emergencies to get to Tenex]

If your connection is broken:

Repeat Step 2 of the Net login procedure on page 4
To check if you are detached, use the where command:
WHERE <SP> USERNAME CR
If you are detached, instead of logging in, type:
ATT <SP> USERNAME <SP> PASSWORD <SP> CR
CTRL O [to wake up NLS if that's where you were, or:]
CTRL C NLS CR [to start over again]

If you accidently delete a file:

BASE C:<sp>Undelete C: File T: FILENAME CR
Undeleted files are:
(FILENAME)

PRACTICE

In addition to trying each command, there is an instructional document called "TNLS-8 Primer" which was written to be used for practice.

TNLS COMMAND SUMMARY FOR THIS COURSE: [alphabetical] you type that part of the command that is capitalized and underlined. CR = Carriage Return.

BACKSPACE CHARACTER = CTRL A; BACKSPACE WORD = CTRL W

CARRIAGE RETURN [formatting] = CTRL V CR

CONTINUE TO INSERT = CTRL E instead of the CR ending your typin
[CTRL X to stop inserting]

COPY STATEMENT:

Copy C: Statement (from) A: ADDRESS CR
(to follow) A: ADDRESS CR
L: CR

CREATE FILE:

<SP>CReate C: File T: FILENAME CR

DELETE STATEMENT:

Delete C: Statement (at) A: ADDRESS CR
OK: CR

DELETE FILE:

Delete C: File T: FILENAME CR
OK: CR

GOTO TENEX:

Goto (subsystem) C: Tenex OK: CR

INSERT STATEMENT:

Insert C: Statement (to follow) A: ADDRESS L: CR
T: TYPEIN CR

INSERT TEXT at the end of a statement

Insert C: Text (to follow) A: STATEMENT NUMBER +e CR
T: TYPEIN CR

JUMP TO ADDRESS:

Jump (to) C: Address A: ADDRESS CR

Jump (to) C: Address A: "TYPEIN" CR

Jump (to) C: Address A: STATEMENT NUMBER "TYPEIN" CR

Jump (to) C: Address A: FILENAME, CR

Jump (to) C: Address A: DIRECTORY,FILENAME, CR

Jump (to) C: Address A: DIRECTORY,FILENAME,STATEMENT NUMBER CR

LOAD FILE:

Load C: File T: FILENAME CR

LOGOUT:

<SP>Logout OK: CR

MOVE STATEMENT:

Move C: Statement (from) A: ADDRESS CR
(to follow) A: ADDRESS CR
L: CR

PRINT FILE:

Print C: File OK: CR

PRINT JOURNAL:

Print C: Journal (mail) OK: CR

PRINT REST:

Print C: Rest OK: CR

Stop printing = CTRL O

PRINT STATEMENT:

Print C: Statement at A: ADDRESS CR
V: CR

Easy print = \

QUIT IN TENEX:

QUIT CR

SHOW DIRECTORY:

<SP>SHow C: Directory (of) OK/T: CR
OK: CR OK:

SUBSTITUTE TEXT IN STATEMENT:

Substitute C: Text (in) C: Statement (at) A: ADDRESS CR
(New TEXT) T: TYPEIN CR
(Old TEXT) T: TYPEIN CR Finished? S/Y/N: ANSWER
Substitutions made: Number

TAIL = 1 .t for ADDRESS

[the last statement in the file -- when single level]

UPDATE A FILE:

Update C: File OK/C: CR

SENDMAIL SYSTEM:

Submit Message or Statement or File, idents [or .lastname], and Interrogate:

BASE C: Goto (subsystem) C: Sendmail OK: CR

SEND C: Interrogate OK: CR

(distribute for action to:) T: JMB,FEED,JCN CR

(distribute for information-only to:) T: RWW CR

(title:) T: Your Example CR

(type of source:) C: Message T: TYPEIN CR

OR..type of source:) C: Statement A: ADDRESS CR

OR..type of source:) C: File A: FILENAME, CR

```
(show status?) Y/N: Y
TITLE: Example
AUTHOR(S): JHB
DISTRIBUTE FOR ACTION TO: jmb feed jcn
DISTRIBUTE FOR INFO-ONLY TO: rww
MESSAGE: TYPEIN of message.
(Send the mail now?) Y/N:  Y

Completed

Quit OK/C: CR
```

Journal Accession Number: 33874

THE BASIC TNLS-8 COURSE OUTLINE

SRI-ARC

18 FEB 76

Applications Development

Augmentation Research Center
Stanford Research Institute
Menlo Park, California 94025

THE BASIC TNLS-8 COURSE OUTLINE

INTRODUCTION TO NLS

NLS = oN Line System

TNLS = Typewriter Version

CAPABILITIES OF SYSTEM:

- Composing
- Editing
- Studying
  - Structuring
  - Browsing - viewing
- Printing
- Publishing
- Communicating -
  - Sending and receiving mail, messages, documents; teleconferencing; etc.
- Storing and retrieving -
  - Record keeping, library services, data bases, searching, etc.
- Calculating

SOME NOTES

This is designed for use when terminals are available for all participants to use/view easily. It is intended to be the first course a person receives on NLS. The commands are shown as they would appear with partial prompting.

COURSE ORGANIZATION

The course is organized by concepts of what a user can do with TNLS at this level. The seven concepts [listed below] are ordered as one would need them to use the system. Under each concept are the exact commands that instruct the computer to perform the function that goes with the concept. There is a command summary at the end of the course outline that lists the same commands alphabetically for easy reference.

The commands which are included in this first course have been selected to let a user write, edit, store, and communicate typewritten information [text]. Those items labeled [Optional] can be covered on the second day of the course.

TNLS CONCEPTS: [Things you can do as covered in this course:]

1. FILES FOR STORAGE
2. TYPING IN INFORMATION
3. TYPING OUT INFORMATION
4. EDITING
5. COMMUNICATING
6. ADDRESSING
7. TROUBLE SHOOTING AND HELP

DEFINITIONS FOR THE COURSE OUTLINE

COMMANDS: You typing some characters to tell the computer what to do.

<SP> means strike the SPACE bar.

Upper case and underlined characters in a TNLS command phrase are what you type.

Words or characters that are uppercase and underlined represent something appropriate that you type.

TYPEIN = a string of characters from the keyboard, ending with an OK.

OK: means you type a carriage return, indicated by a CR.

CR: means you strike the carriage return key; OK.

WHERE YOU ARE: Where the computer thinks you are pointing [to some character in some file]; you tell it by specifying an address; this is where your command will be done.

CTRL means you hold down the control [CTRL] key while typing the specified character.

BASE C: is the TNLS ready signal. It means that you can type in an editing or file-handling command [like home base...].

SEND C: is the Sendmail subsystem ready signal. It means that you can type in a Sendmail command.

GETTING TO NLS

THE TERMINAL AND USE [if dialing in]

If dialing in, turn on, dial TIP number, place receiver in cradle after hearing tone, and make sure terminal is online.

NETWORK [if used]

Net login: brief summary for dial in users [steps are numbered]

[I] Type E [to get the Network's attention]

[II] Type @ O <SP> 43 CR
[to Open a connection to Office-1, Host 43; Host 49 for BBNB]

You now should be connected to TENEX
[@ is the TENEX ready signal]

TENEX EXECUTIVE

Login procedure:

[III] Type LOG <SP> USERNAME <SP> PASSWORD <SP> CR
[the last SPACE fills in the account number automatically; you're then ready to call NLS]

[Optional] Group allocation quota:
GROUPSTAT CR [to see who else is using your slot]

Calling NLS:

[IV] Type NLS CR

[Optional] To get back to TENEX:

BASE C: Goto (subsystem) C: Tenex CR
[you may continue in NLS by typing QUIT CR]

To leave the system, logout in NLS:

BASE C: <SP> Logout OK: CR

To close the network connection:

@ C CR

BASIC TNLS:

Abort command = CTRL X [kills the command before the final CR]

1. FILES

The initials file

The initials file is automatically your first file, and is named after your initials [it is also one of your mailboxes]

The origin statement

The origin statement contains the file name and other information about the file for the system. It should not be edited. It is numbered 0, but no number will be printed.

Creating new files -- use the Create File command

```
BASE C: <SP>CReate C: File T: FILENAME CR
< DIRECTORY, FILENAME.NLS;1, >
```

[a filename can be any short "word", like a folder label]

[Optional] To see a list of all your files use the Show Directory command:

```
BASE C: <SP>SHow C: Directory (of) T/OK: CR
OK: CR OK:
```

To work in another file:

```
BASE C: Load C: File T: FILENAME CR
```

2. TYPING IN INFORMATION

Insert Statement - lets you enter sentences, headings, or paragraphs

```
BASE C: Insert C: Statement (to follow) A: ADDRESS CR
L: CR
T: TYPEIN CR
```

[ADDRESS means statement number; TYPEIN means you type whatever you want in the statement]

[Optional] Continue to insert by typing a CTRL E instead of the first CR for a final OK:

[puts you in the "Enter mode" -- you type CR
to end each statement, when you are finished
type a CTRL X after the prompt L:]

Backspace: You may backspace while typing to immediately correct errors

backspace character = CTRL A
[the deleted character will be printed after a slash]
backspace word = CTRL W
[an underline or backarrow will be printed]

[Optional] To insert text at the end of a statement:

BASE C: Insert C: Text (to follow) A: STATEMENT NUMBER +e CR
T: TYPEIN CR

[+e means the end of the statement]

3. TYPING OUT INFORMATION *To stop printing type a CTRL O
[it takes some time to get through!!]

Printing the file:

BASE C: Print C: File OK: CR

Printing a statement:

BASE C: Print C: Statement at A: STATEMENT NUMBER CR
V: CR

Printing the rest of the file:

BASE C: Print C: Rest OK: CR

[Optional] Easy print: A shortcut to print the current statement

BASE C: \ [backslash prints the statement where you are]

4. EDITING

To change information that has been typed in:

To change information within a statement:

Substitute Text in Statement [to correct most errors]:

```
BASE C: Substitute C: Text (in) C: Statement (at) A: ADDRESS
CR
(New TEXT) T: TYPEIN CR
(Old TEXT) T: TYPEIN CR Finished? S/Y/N: Y [for yes]
Substitutions made: Number
```

[replaces the old text with the new text every time it occurs in the statement]

To change whole statements:

Delete Statement

```
BASE C: Delete C: Statement (at) A: ADDRESS CR
OK: CR
```

[Optional] Move Statement:

```
BASE C: Move C: Statement (from) A: ADDRESS CR
(to follow) A: ADDRESS CR
L: CR
```

[Optional] Copy Statement:

```
BASE C: Copy C: Statement (from) A: ADDRESS CR
(to follow) A: ADDRESS CR
L: CR
```

To Delete a File

```
BASE C: Delete C: File T: FILENAME CR
OK: CR
```

[Careful, this removes all versions of the file. You can Undelete a File anytime before Logout with the Undelete command.]

To Update a File: [should be done periodically to save changes for backup, not imperative]

BASE C: Update C: File OK/C: CR
<DIRECTORY, FILENAME.NLS;2,>

[Optional] Formatting Technique: use carriage returns to control line length

To type a carriage return within a TYPEIN, type CTRL V CR

5. [Optional] COMMUNICATING

SENDMAIL SYSTEM: How to Send Journal Mail

Example of submitting a message using idents [or .lastname] and Interrogate [where the system prompts you]:

BASE C: Goto (subsystem) C: Sendmail OK: CR

SEND C: Interrogate OK: CR

(distribute for action to:) T: JMB,FEEDBACK,SGR CR

(distribute for information-only to:) T: JCN CR

(title:) T: Your Example CR

(type of source:) C: Message T: TYPEIN CR

(show status?) Y/N: Y [the status typed by the system:]

TITLE: Your Example
AUTHOR(S): jnb
DISTRIBUTE FOR ACTION TO: jmb feedback sgr
DISTRIBUTE FOR INFO-ONLY TO: jcn
MESSAGE: [Typein of message will be repeated]

(send the mail now?) Y/N: Y [for yes]

Completed

SEND C: Quit OK/C: CR

To send a Statement that's already stored online, use the following instead of Message for the type of source in the Interrogate command [see the Command Summary for an example]

(type of source:) C: Statement A: ADDRESS CR

To send a File use the following instead of Message [see the Command Summary for an example]

(type of source:) C: File A: FILENAME,CR

How to read journal mail you've received

The mail box is in your initials file under a statement called "(Journal)"

Use the Print Journal command

BASE C: Print C: Journal (mail) OK: CR

To empty your mail box: substitute (read) for (journal)

[this will empty your mailbox by having all journal mail that is already read put in a different place from incoming mail. A new journal mailbox will be created whenever there is mail to be delivered. This is a temporary method for the present course level]

TENEX ways of communicating: SNDMSG and LINK

To send a Message in TENEX

BASE C: Goto (subsystem) C: Tenex OK: CR
[must QUIT]:

@SNDMSG CR [the system will prompt you:]
(To (? for help):) TYPEIN CR [lastnames separated by comma]
(cc (? for help):) TYPEIN CR [lastnames separated by comma]
(Subject:) TYPEIN CR [subject of your message]
(Message ? for help): TYPEIN
CRTL Z [to terminate and send the message]
(Q, S, ?, carriage return:) CR [to send the message]

@QUIT CR [to go back to where you were in TNLS]

To read a Message in TENEX

MESS CR

In some systems you must use the readmail command:

READMAIL CR CR

Linking [in TENEX] [first ask where the person is:]

BASE C: Quit OK/C: Nls OK: CR
WHERE <SP> USERNAME CR [do not link when user is in SNDMSG, OUTPRC, NOUTPRC, or XLIST]

LINK <SP> USERNAME CR [precede comment with a ";" and end with a CR; repeat every 3 lines]
BYE CR [to break the link]
CON CR [returns you to NLS]

6. ADDRESSING

Where the control marker is -- type a / [see the questionmark key on some terminals]

This will show an arrow pointing to the character that you are at: ==>x

To move the control marker within your current file use the Jump to Address command

BASE C: Jump (to) C: Address A: ADDRESS CR

An address can be one of the following anytime you see the prompt A:

STATEMENT NUMBER [NOTE: TNLS automatically renumbers statements when appropriate]

BASE C: Jump (to) C: Address A: STATEMENT NUMBER CR

[Optional] "TYPEIN" To find some word or text, enclose whatever word or series of characters you want to find in quotes:

BASE C: Jump (to) C: Address A: "TYPEIN" CR

[takes you to the first occurrence of the TYPEIN to the right and down in your file]

[Optional] STATEMENT NUMBER and "TYPEIN"; To find some word or text starting in a particular statement use:

BASE C: Jump (to) C: Address A: STATEMENT NUMBER "TYPEIN" CR

[Optional] STATEMENT NUMBER .t ("tail"); To find the last statement in the file. Use this address to add statements to the end of your file.

BASE C: Jump (to) C: Address A: STATEMENT NUMBER .t CR

Addressing across files and directories

To address another file:

BASE C: Jump (to) C: Address A: FILENAME, CR

To address another person's file:

BASE C: Jump (to) C: Address A: DIRECTORY,FILENAME, CR

To address a particular statement in another person's file:

BASE C: Jump (to) C: Address A: DIRECTORY,FILENAME,STATEMENT NUMBER CR

7. TROUBLE SHOOTING AND HELP

Immediate assistance from the system:

Type ? for a list of all the possible command words.

[Optional] Status commands

Two CTRL T's [note words RUNNING or WAIT -- system should be either running or waiting for you]

Call SRI-ARC, [415 326-6200, ext. 3630]
Or link to Feedback, Bair, Roetter, or Beck at Office-1

[Optional] FEEDBACK mechanism: [any complaints, questions, problems, suggestions]

SNDMSG or Sendmail to FEEDBACK
Response should be no later than 1 working day

[Optional] Remedies

CTRL C, RESET CR, NLS CR
[use CTRL C only in emergencies to get to Tenex]

If your connection is broken:

Repeat Step 2 of the Net login procedure on page 4
To check if you are detached, use the where command:
WHERE <SP> USERNAME CR
If you are detached, instead of logging in, type:
ATT <SP> USERNAME <SP> PASSWORD <SP> CR
CTRL O [to wake up NLS if that's where you were, or:]
CTRL C NLS CR [to start over again]

If you accidently delete a file:

BASE C: Undelete C: File T: FILENAME CR
Undeleted files are:
(FILENAME)

PRACTICE

In addition to trying each command, there is an instructional document called "TNLS-8 Primer" which was written to be used for practice.

TNLS COMMAND SUMMARY FOR THIS COURSE: [alphabetical] you type that part of the command that is capitalized and underlined. CR = Carriage Return.

BACKSPACE CHARACTER = CTRL A; BACKSPACE WORD = CTRL W

CARRIAGE RETURN [formatting] = CTRL V CR

CONTINUE TO INSERT = CTRL E instead of the CR ending your typin
[CTRL X to stop inserting]

COPY STATEMENT:

Copy C: Statement (from) A: ADDRESS CR
(to follow) A: ADDRESS CR
L: CR

CREATE FILE:

<SP>CReate C: File T: FILENAME CR

DELETE STATEMENT:

Delete C: Statement (at) A: ADDRESS CR
OK: CR

DELETE FILE:

Delete C: File T: FILENAME CR
OK: CR

GOTO TENEX:

Goto (subsystem) C: Tenex OK: CR

INSERT STATEMENT:

Insert C: Statement (to follow) A: ADDRESS L: CR
T: TYPEIN CR

INSERT TEXT at the end of a statement

Insert C: Text (to follow) A: STATEMENT NUMBER +e CR
T: TYPEIN CR

JUMP TO ADDRESS:

Jump (to) C: Address A: ADDRESS CR

Jump (to) C: Address A: "TYPEIN" CR

Jump (to) C: Address A: STATEMENT NUMBER "TYPEIN" CR

Jump (to) C: Address A: FILENAME, CR

Jump (to) C: Address A: DIRECTORY,FILENAME, CR

Jump (to) C: Address A: DIRECTORY,FILENAME,STATEMENT NUMBER CR

LOAD FILE:

Load C: File T: FILENAME CR

LOGOUT:

<SP>Logout OK: CR

MOVE STATEMENT:

Move C: Statement (from) A: ADDRESS CR
(to follow) A: ADDRESS CR
L: CR

PRINT FILE:

Print C: File OK: CR

PRINT JOURNAL:

Print C: Journal (mail) OK: CR

PRINT REST:

Print C: Rest OK: CR

Stop printing = CTRL O

PRINT STATEMENT:

Print C: Statement at A: ADDRESS CR
V: CR

Easy print = \

QUIT IN TENEX:

QUIT CR

SHOW DIRECTORY:

<SP>SHow C: Directory (of) OK/T: CR
OK: CR OK:

SUBSTITUTE TEXT IN STATEMENT:

Substitute C: Text (in) C: Statement (at) A: ADDRESS CR
(New TEXT) T: TYPEIN CR
(Old TEXT) T: TYPEIN CR Finished? S/Y/N: ANSWER
Substitutions made: Number

TAIL = 1 .t for ADDRESS

[the last statement in the file -- when single level]

UPDATE A FILE:

Update C: File OK/C: CR

SENDMAIL SYSTEM:

Submit Message or Statement or File, idents [or .lastname], and Interrogate:

BASE C: Goto (subsystem) C: Sendmail OK: CR

SEND C: Interrogate OK: CR

(distribute for action to:) T: JMB,FEED,JCN CR

(distribute for information-only to:) T: RWW CR

(title:) T: Your Example CR

(type of source:) C: Message T: TYPEIN CR

OR..type of source:) C: Statement A: ADDRESS CR

OR..type of source:) C: File A: FILENAME, CR

```
(show status?) Y/N: Y
TITLE: Example
AUTHOR(S): JHB
DISTRIBUTE FOR ACTION TO: jmb feed jcn
DISTRIBUTE FOR INFO-ONLY TO: rww
MESSAGE: TYPEIN of message.
(Send the mail now?) Y/N: Y

Completed

Quit OK/C: CR
```

To change autho

n for show status
n for send mail

Author:
SEND C: ~~sh~~ow status
SEND C: Send

COURSE OUTLINE:
INTRODUCTION TO THE DISPLAY ON-LINE SYSTEM (DNLS)

ARC-ADG

12 FEB 76

Applications Development

Augmentation Research Center
Stanford Research Institute
Menlo Park, California 94025

INTRODUCTION TO NLS

AKW = Augmented Knowledge Workshop

PURPOSE OF SYSTEM: Augmentation of Knowledge Work

GOAL: To provide computer based tools to accomplish all aspects of knowledge work with an emphasis on collaboration.

OVERVIEW of system

NLS = oN Line System

DNLS = Display Version

CAPABILITIES OF SYSTEM:

- Composing
- Editing
- Studying
  - Structuring
  - Browsing - viewing
- Printing
- Publishing
- Communicating -

  sending and receiving mail, messages, documents; teleconferencing; etc.
- Storing and retrieving -

  record keeping, library services, data bases, searching, etc.
- Calculating

Organization of NLS Courses

COURSE LEVELS:

NLS training is divided into course levels for ease of learning. Three are concered with TNLS (typewriter version) and one treats DNLS. Each level corresponds to what can be covered at one time. The information introduced at each level is determined by difficulty, usefulness, complexity, and quantity (i.e., so that there is not an excessive amount to cover at any one time and to provide an opportunity for practice between courses).

> Each level in the series of NLS courses contains most of the commands from the previous level for review in addition to the commands to be introduced (which are marked by an *).

BASIC TNLS:
This is the first course level which covers those commands necessary to enter, edit, and "mail" typewritten information. It has a special organization.

INTRODUCTION TO TNLS STRUCTURE AND VIEWING:
This is the second course which introduces NLS structure (hierarchical) and special tools for viewing structured information ("viewspecs") in addition to expanding upon the Basic course.

*INTRODUCTION TO DNLS:
This is the first course dealing with the display verison of NLS. It introduces the equipment and its use, and deals with the commands necessary to edit, view, address and communicate in DNLS. This course introduces commands which are new to users who have completed TNLS Courses I and II. It is important to complete these prerequisite courses and to have had sufficient time to practice the commands that have been introduced before taking this course.

This course is divided into sections with the headings shown below. The commands under each heading can be used to perform the general operation denoted by the heading, e.g., "editing" includes commands that allow text to be changed in various ways.

COURSE HEADINGS:

1. INTERFACING TO DNLS
2. STRUCTURE
3. ADDRESSING AND VIEWING
4. EDITING
5. COMMUNICATING
6. LEAVING NLS
7. TROUBLE SHOOTING AND HELP

*1. INTERFACING TO DNLS

THE WORKSTATION

*The Lineprocessor: (See The Lineprocessor User's Guide;)

A microcomputer which processes coordinates of the location of each character on the screen.

Allows the display to run in DNLS.

The four silver toggles on the front of the lineprocessor should all be down.

*The Keyboard:

Letters and numbers are arranged as on a standard typewriter.

On the Datamedia terminals, there are special keys which control the display such as DPLX, BREAK, MSTR CLEAR, ERASE, ROLL, TAPE and UNLOCK. These keys change the display function and should not be used once they are properly set.

OK or CA (Command Accept): Used to terminate TYPIN and viewspecs, in the pointing process, or to give a confirmation.

CMD DEL (Command Delete): Used to abort a command sentence.

BACK SPACE (Backspace Character): Used to delete one character from input.

BACK SPACE WORD: Used to delete one word from input.

RETURN (Carriage Return): Used to enter a carriage return into text and to terminate Tenex commands.

ESC (Escape): Used to complete command words and filenames in Tenex and filenames in NLS.

*The Keyset (OPTIONAL):

An alternative to the keyboard; used in conjunction with the mouse.

See the viewspec card for key codes
[Key codes follow a logical order according to the alphabet.]

*The Mouse:

[See Figure 2;]

*One Button Alone:

Right button alone - OK, CA or bugging

Center button alone - CD (<CTRL-X>)

Left button alone - BC (<CTRL-A>)

*Buttons Used in Combination:

Right and center buttons - OK/REPEAT (<CTRL-B>)

Left and center buttons - BW (<CTRL-W>)

Right and left buttons - ESCAPE or ALTMODE

*Holding Down Buttons While Typing Characters:

Left and center buttons while typing characters
- lowercase viewspec
[Type a viewspec "f" to recreate the screen or the view will not change until after the next command is completed.]

All three buttons while typing characters
- capital viewspec
[Type a viewspec "f" to recreate the screen or the view will not change until after the next command is completed.]

Center button while typing a letter - capital letter

Left button while typing a character
- number or non-alphabetic character

*The Display Terminal:

Ensure that the display is set to online mode, full duplex.

CONNECTING TO NLS [See the Lineprocessor User's Guide for detailed instructions.]

*Using a Display Terminal Connected to a TIP:

@I <SP> 25 CR

[Changes the TIP intercept character from @ to <CTRL-Y> (unless it has already been changed).]

<CTRL-Y> O <SP> 43 CR [For Office-1;]
[<CTRL-Y> O <SP> 49 CR [For BBNB;]

LOG USERNAME <SP> PASSWORD <SP> CR

NO RAISE CR [For BBNB only;]

TERMINAL LINEPROCESSOR CR [Without this command you will get TNLS.]

NLS CR

*Using A Display Terminal Connected To An ELF System:

<CTRL-C> [To get ELF's attention;]

LOG <SP> USERNAME <SP> PASSWORD CR

TELNET CR

ESCAPE CHARACTER = <CTRL-Y> CR
[Changes the Telnet intercept character from <CTRL-Z> to <CTRL-Y>; in some later versions of ELF this is not necessary.]

OFFICE-1 CR [Or BBNB CR;]

LOG USERNAME <SP> PASSWORD <SP> CR

NO RAISE CR [For BBNB only;]

TERMINAL LINEPROCESSOR CR

NLS CR

THE DISPLAY

See Figure 3

*Feedback Area: [A minimum of 6 lines at the top of the screen, divided into 5 windows.]

*1. Viewspec Window:

Indicates the current status of certain viewspecs; this section of the screen will flash or be underlined when the viewspecs can be changed.

Displayed viewspecs: [The default settings indicate that nothing is filtered or left out.]

Number of levels; number of lines
[The first two numbers or the words "ALL ALL";]
h - show all branches
j - don't filter statements
u - recreate display after each change
C - show statement names
P - user sequence generator off

*2. Typewriter Simulation Window:

This area provides feedback, and shows interaction with Tenex. It is used for error messages, system messages and the name of the file being loaded. It will remain empty if there is no information. If Tenex is called with a <CTRL-C>, interaction is shown here.

*3. Subsystem Name: Displays the name of the subsystem being used.

*4. Command Feedback Window:

Displays the current command phrase with noise words and prompts:

A: Address
C: Command word
T: Typin
L: Level
V: Viewspecs
OK: Command Accept
Y/N: Yes or No
B: Bug [point with the mouse and press OK]
...> Executing command

*5. Type In Feedback Window:

Displays the typein or address as it will be accepted after an OK

*File Window:

Contains files or parts of files which can be pointed to with the mouse. Differs from the feedback windows in that you can point to it.

*Moving Bug or Cursor:

A traveling mark on the screen which can be controlled with the mouse. [It is controlled by the Lineprocessor independently from the host.]

2. ORGANIZATION OF THE SYSTEM

FILES & DIRECTORIES

Information in the origin ("parent") statement of a file:
The origin statement contains the file name, version number, the date and time of last modification, the ident of the last person to modify the file, and 4 semicolons. The statement should not be edited (except to add output processor directives or status information).

File names

Types of files [indicated by filename extensions;]

TXT = sequential file which can be copied into NLS
COPY = a temporary sequential file, usually a message

User creation of files:

<SP>CReate File FILENAME OK

To see a list of all your files:

<SP>SHow Directory (of) OK OK

*Copy Directory (of) BUG/TYPEIN (to follow) BUG/ADDRESS LEVELADJUST OK

[Provides a useful list of links to your files - you can easily Jump to each file by pointing to the links.]

Load File FILENAME OK

[Use the Jump to Link command instead; an <ESC> will finish any filename when enough characters are typed to be uniquely recognized.]

3. ADDRESSING and VIEWING: [To control the view of stored information.]

*BUGGING

Instead of typing in an address as you would in TNLS, the mouse can be used to point to a specific position on the screen by depressing the right-most mouse button in response to the prompt B:. A "bugmark" will mark the character pointed to with either a circle, an underline, a blot-out, or a character video inversion. This indicates that the host computer knows the position you selected.

*JUMP COMMANDS (changing the information that is being viewed):

MOST COMMONLY USED JUMP COMMANDS:

*Jump (to) BUG VIEWSPECS OK

This command positions the statement pointed at to the top of the screen.

*Jump (to) Item BUG/ADDRESS VIEWSPECS OK

This command positions the statement addressed or pointed at to the top of the screen.

*Jump (to) Back BUG/ADDRESS VIEWSPECS OK

This command positions the statement back one from the statement pointed at to the top of the screen.

BACK refers to the statement immediately preceding the statement pointed at, regardless of level or source.

*Jump (to) <SP> Next BUG/ADDRESS VIEWSPECS OK

This command positions the statement immediately following the statement pointed at, regardless of level or source, to the top of the screen.

NEXT refers to the statement immediately following the statement pointed at, regardless of level or source.

Jump (to) Origin BUG/ADDRESS VIEWSPECS OK

This command positions statement 0 at the top of the screen.

Jump (to) Return OK ANSWER OK

This command flashes back the last line within the current file that was at the top of the screen. If "n" is typed for ANSWER, the line that was at the top of the screen before that will be flashed back. When "y" is typed for ANSWER, the view will be returned, beginning with the chosen flash-backed line, to the display.

VIEWSPECS: To specify how the information is displayed, use the characters below when prompted with a V:

w = Default, all lines & levels (show all of the text)
m/n = numbers on/off
y/z = blank lines on/off

To clip levels and lines, use lower case viewspecs including:

a/b - show one level less/more
c/d - show all levels/show first level
e - show referenced statement level
*f - recreates display
g/h - show branch only/show all branches
*o/p - frozen statements on/frozen statements off
q/r - show one line less/more
s/t - show all lines/show first lines only
w/x - show all lines, all levels/show one line, one level
*F - forces the display to recreate when there is some problem preventing it

*If viewspecs are set using the mouse an "f" (recreate display) must be typed before releasing the mouse buttons. [If for some reason the display does not recreate with viewspec "f", try viewspec "F".]

In DNLS it is more efficient, in most cases, to bug a location rather than address it, permitting you to work with viewspec "n" rather than "m". Viespec "m" will cause the screen to refresh after each structural change made to the file. [This is necessary to ensure that statement number changes due to editing are displayed.]

LESS USED JUMP COMMANDS: (OPTIONAL)

*Jump (to) Successor BUG/ADDRESS VIEWSPECS OK

This command positions the successor of the statement pointed at to the top of the screen.

SUCCESSOR refers to the statement immediately following the current statement at the same level and with the same source.

*Jump (to) Predecessor BUG/ADDRESS VIEWSPECS OK

This command positions the predecessor of the statement pointed at to the top of the screen.

PREDECESSOR refers to the statement immediately preceding the current statement at the same level and with the same source.

*Jump (to) Head BUG/ADDRESS VIEWSPECS OK

This command positions the head of a specified plex to the top of the screen.

HEAD refers to the first statement of a plex.

*Jump (to) Tail BUG/ADDRESS VIEWSPECS OK

This command positions the last statement pointed at to the top of the screen.

TAIL refers to the last statement of a plex at the level pointed to.

Jump (to) End (of branch) BUG/ADDRESS VIEWSPECS OK

This command positions the end of the branch pointed at to the top of the screen.

END refers to the last statement of a branch.

*Jump (to) Up BUG/ADDRESS VIEWSPECS OK

This command positions the statement one level up from the statement pointed at to the top of the screen.

UP refers to the statement one level up from the current statement.

*Jump (to) Down BUG/ADDRESS VIEWSPECS OK

This command positions the statement one level down from the statement pointed at to the top of the screen.

DOWN refers to the statement one level down from the current statement, if there is one.

*Jump (to) Address (relative to) BUG/ADDRESS
ADDRESS VIEWSPECS OK

The second ADDRESS is evaluated starting at the first BUG/ADDRESS. The statement pointed to by the last address given moves to the top of the screen.

TO FIND A WORD OR STRING OF CHARACTERS [no quotes]:

Jump (to) Word First BUG/TYPEIN VIEWSPECS OK

Jump (to) Word Next BUG/TYPEIN VIEWSPECS OK

Jump (to) Content First BUG/TYPEIN VIEWSPECS OK

Jump (to) Content Next BUG/TYPEIN VIEWSPECS OK
[Type a <CTRL-B> instead of BUG/TYPEIN to continue searching for the same content.]

TO JUMP USING A LINK:

*Jump (to) Link BUG/TYPEIN OK

[Note: The Jump to Link command uses the prompt B/T: which means that you may alternatively type in a link rather than pointing to it in the text of a statement.]

Links: Special forms of text that may be used for addressing and other purposes.

Characteristics of Links:

-- it is text in a statement rather than typed in after the A:

-- must be surrounded by angle brackets < > (or parentheses).

-- may contain any logical Address.

-- it may include viewspecs that will take effect at the address in the link.

-- must be in one of the following forms:

<DIRECTORY,FILENAME,IN-FILE-ADDRESS:VIEWSPECS>

[Without optional Viewspecs:]
<DIRECTORY,FILENAME,IN-FILE-ADDRESS>

[or in current directory:]
<FILENAME,IN-FILE-ADDRESS>

[or in current file:]
<IN-FILE-ADDRESS>

[or:] <:VIEWSPECS> [Only the viewspecs will be changed.]

Note that the different fields default to the current value if not specified (the same as addresses).

-- may include things other than addresses and/or viewspecs [which will be covered in more advanced courses;]

JUMPING BETWEEN FILES AND DIRECTORIES:

To address another file in your directory you need to add the FILENAME to the address within a file. To address a file in another directory, you need to add the DIRECTORY name as well as the filename. FILENAME and DIRECTORY must be followed by commas.

To address another file:

FILENAME, IN-FILE-ADDRESS OK

[If the IN-FILE ADDRESS is not specified it will be statement 0.]

To address another user's file:

DIRECTORY, FILENAME, IN-FILE-ADDRESS OK

[e.g., Copy Branch (from BAIR,JHB,1 OK (to) 3a OK]

TO GO BACK TO PREVIOUS FILES:

Jump (to) File Return OK ANSWER OK

[Flashes back the name of the last file you were in. If "n" is typed for ANSWER, the name of the file you were in before that will be flashed back. When "y" is typed for ANSWER, you will return to the last view of that file.]

4. EDITING
[Changes occur on the screen as you execute commands.]

Syntax: VERB NOUN B/A: BUG/ADDRESS(ES) (V:VIEWSPECS) (L:LEVEL)
(B/T: BUG/TYPEIN) OK (OK? OK)

[Consistently, commands will have a VERB then a NOUN. Then optionally, ADDRESSES, VEIWSPECS and LEVEL precede the final OK. See the specific command for complete syntax.]

STRING and STRUCTURE = "nouns":

STRING [One of the following command words which refer to part of a statement]:

Character

Word [Note that the system readjusts spaces;]

Text [Two bug marks/addresses necessary;]

STRUCTURE [One of the following command words that refers to one or more statements]:

Statement - the basic element of structure in a file.

Branch - a statement plus substructure (if any).

Group - a set of contiguous branches at the same level and with the same source.
[Two bug marks/addresses necessary;]

*Plex - a complete list of branches at the same level with the same source, i.e., all the branches in the source branch. Especially useful when a list does not fit on the screen and you want to move, copy or delete it.

EDITING COMMANDS = "verbs":

[Note that in DNLS you usually "bug" items instead of typing in an address. The commands are the same as in the Second TNLS course.]

LEVEL-ADJUST determines the level of a statement at a new location -- it must be ended by an OK

Just an OK = same level

u (position up a level from referenced statement)

d (position down a level from referenced statement)

INSERT

Insert Statement (to follow) BUG/ADDRESS LEVEL-ADJUST BUG/TYPEIN OK

Insert STRING (to follow) BUG/ADDRESS BUG/TYPEIN OK

Continue to insert: <CTRL-E> instead of a final OK puts you in the Enter statement mode, type a CD to get out. Some terminals have an insert key which can be used.

[Text will be echoed in the typin feedback window as it is typed in, before being added to the file.]

DELETE

Delete File BUG/TYPEIN OK

[The deleted filename(s) will be displayed. Type an OK to continue. When your screen refreshes after an OK, the deleted file will still appear, but should not be edited - the file will be deleted after you leave it.]

Delete STRUCTURE (at) BUG/ADDRESS OK

Delete STRING (at) BUG/ADDRESS OK

MOVE

Move STRUCTURE (from) BUG/ADDRESS (to follow) BUG/ADDRESS LEVEL-ADJUST OK

Move STRING (from) BUG/ADDRESS (to follow) BUG/ADDRESS OK

COPY

Copy STRUCTURE (from) BUG/ADDRESS (to follow) BUG/ADDRESS
LEVEL-ADJUST OK

Copy STRING (from) BUG/ADDRESS (to follow) BUG/ADDRESS OK

*Copy Directory (of) BUG/TYPEIN OK (to follow) BUG/ADDRESS
LEVEL-ADJUST OK

[Provides a useful list of links to your files - you can easily Jump to each by pointing to the links.]

REPLACE

Replace STRUCTURE (at) BUG/ADDRESS (by) BUG/TYPEIN OK

*Replace STRING (at) BUG/ADDRESS (by) BUG/TYPEIN OK

TRANSPOSE

Transpose STRUCTURE (at) BUG/ADDRESS (and) BUG/ADDRESS OK

*Transpose STRING (at) BUG/ADDRESS (and) BUG/ADDRESS OK

BREAK

Break Statement (at) BUG/ADDRESS LEVEL-ADJUST OK

[Used to break a statement into two statements after the character pointed to; it is often used if a statement is longer than fits on the screen.]

APPEND

Append Statement (at) BUG/ADDRESS (to) BUG/ADDRESS
(join with) BUG/TYPEIN OK

[Used to join two statements together to form one statement; BUG/TYPEIN is text that will be added where the old and new statements join. Use <CTRL-N> if no BUG/TYPEIN is desired.]

SUBSTITUTE

Substitute STRING (in) STRUCTURE (at) BUG/ADDRESS OK
(New STRING) T: BUG/TYPEIN OK
(Old STRING) T: BUG/TYPEIN OK (Finished?) S/Y/N: Y [for yes]
(Substitutions made: Number)

[Will replace the old STRING with new STRING every time it finds it in the STRUCTURE. If you type S for Show, the screen will recreate to show the substitutions to be made. A CA will return you to your previous view.]

UPDATE FILE [Not imperative, but good practice;]

Update File Compact OK

Update File OK

*EDGES (OPTIONAL)

*Insert Edge (perpendicular to) MARGIN EDGE OK

[Inserts imaginary lines dividing the screen into as many as 8 parts or "windows" so different information can be viewed or edited in each window.]

*Move Edge (from) BUG/ADDRESS (to) BUG/ADDRESS OK

*Delete Edge (at) BUG/ADDRESS OK

*FROZEN STATEMENTS (OPTIONAL)

*Freeze Statement (at) BUG/ADDRESS VIEWSPECS OK

[Keeps the specified (bugged) statement(s) at the top of the screen when viewspec o is on; viewspec p will make the frozen statements invisible - similar to split screen in capability.]

*Release Frozen (statement at) BUG/ADDRESS OK

["Unfreezes" a specified statement.]

*Release All (frozen statements) OK

5. COMMUNICATING with other users

SENDMAIL SUBSYSTEM and the Journal

*The Initial file is automatically loaded when NLS is entered; the Journal branch of this file acts as a mailbox for incoming Journal mail. When NLS is called the phrase "You Have New Journal Mail" is flashed in the typewriter window if a new delivery has been made.

*You can load a file where information is contained; Goto Sendmail and bug the message, title, etc., in response to the questions below [up to 6 bug/marks maximum]. The currently loaded file will stay on the screen while in Sendmail.

Goto (subsystem) Sendmail OK

Interrogate Command

```
Interrogate OK
(distribute for action to:) IDENT(s) or .LASTNAME
(distribute for information-only to:) IDENT(s) or .LASTNAME
(title:) BUG/TYPEIN
(type of source:) Message BUG/TYPEIN  or
                  STRUCTURE (at) BUG/ADDRESS  or
                  File (at) BUG/ADDRESS
(show status?) ANSWER
   [Will recreate the screen, type OK to continue with
    this command;]
(distribute the mail now?) ANSWER
```

Individual commands: instead of Interrogate, specify by using the following:

Title BUG/TYPEIN OK

Distribute (for) Information (Only) (to) IDENT(s) or .LASTNAME OK

Distribute (for) Action (to) IDENT(s) or .LASTNAME OK

Comment BUG/TYPEIN OK

To send a message or statement:

Message BUG/TYPEIN OK

<SP>STatement (at) BUG/ADDRESS OK

To send a structure or file:

<SP>Group (at) BUG/ADDRESS OK

Branch (at) BUG/ADDRESS OK

File BUG/ADDRESS OK

<SP> Plex (at) BUG/ADDRESS OK

<SP>SHow Status OK

Send (the mail) OK

To identify a user by lastname or ident:

<SP>SHow Record (for ident) .LASTNAME OK
[precede the lastname by a period;]

<SP>SHow Record (for ident) IDENT OK

[The entire display area of the screen
is used to show the information.]

To leave the Sendmail subsystem when you are done:

Quit OK [Returns you to Base;]

CALLING TENEX

*1. Goto (Subsystem) Tenex OK
[Clears the screen and simulates a typewriter terminal - Use for SNDMSG.]

QUIT CR
[Returns you to DNLS.]

*2. Quit Nls OK
[Clears the screen and simulates a typewriter terminal - as when you logged in]

CONTINUE CR
[Returns you to DNLS]

*3. <CTRL-C>
[Gives a 2-line typewriter simulation window at the top of the screen which may be used for interaction with Tenex.]

CONTINUE CR
[Returns to DNLS, no prompt is given.]

SEND MESSAGE (Tenex)

Goto (subsystem) Tenex OK

```
SNDMSG CR      [The system will prompt you:]
(To (? for help):) TYPEIN CR  [Lastnames separated by comma;]
(cc (? for help):) TYPEIN CR  [Lastnames separated by comma;]
(subject:) TYPEIN CR  [Subject of your message;]
(message:) TYPEIN
*<CTRL-S>     [OPTIONAL] [Allows you to see a message before
                          sending it;]
<CRTL-Z> CR      [To terminate and send the message;]
QU CR  [To return to NLS;]
```

Linking (Tenex)

First: Goto (subsystem) Tenex OK, Quit OK or <CTRL-C>
[You can use either the entire screen or the typewriter simulation window (no three line limit in the typewriter window when linking.) You do not have to go to Tenex to respond to a link, merely type a ";".]

WHE<ESC>re (is user) USERNAME CR [Do not link when user is in SNDMSG, OUTPRC, NOUTPRC, or XLIST.]

LIN<ESC>k (to) USERNAME CR [Precede each comment with a ; end with a CR, repeat every 3 lines.]

BR CR [To break the link;]

QU CR [Returns you to NLS;]

6. LEAVING NLS

For TIP Users:

<SP>Logout OK

<CTRL-Y> c
[Closes the TIP connection;]

For ELF Users:

<SP>Logout OK

<CTRL-Y>
[Recalls Telnet;]

QUIT CR or DISCONNECT CR

LOGO CR

7. TROUBLE SHOOTING AND HELP

See the Lineprocessor User's Guide for detailed instructions concerning the workstation.

*The Error Light on the lineprocessor indicates hardware problems. Push the left button to reset, and continue working.

Immediate assistance from the system:

Type ? for commands or needed information after any prompt.

HELP:

Type <CTRL-Q> for help concerning what you are doing or type H for the Help command (after typing H you can type any word in NLS you wish to know about). The screen will recreate to show help. <CTRL-X> gets you out of help and back to where you were.

Help TYPEIN OK

Help OK

System Status:

Two <CTRL-T>'s

[Note the words RUNNING or WAIT -- WAIT means the computer is waiting for you to do something. The message will appear in the typewriter simulation window.]

<SP>SHow <SP>Disk (space status) OK

Send a message or sendmail item to: FEEDBACK

Call SRI/ARC, (415 326-6200, ext.3630)

Link to FEEDBACK

Remedies:

<CTRL-C>
RESET CR
NLS CR

If over allocation:

<SP>Trim Directory (no. of versions to keep) BUG/TYPEIN OK (really?)
OK

<SP>Expunge Directory OK

Update File Compact OK [Re-stores file more efficiently
in the computer.]

Delete Modifications (to file) OK (Really?) OK
[Destroys all changes since the last update!]

If your connection is broken:

Repeat login procedure to get to your host

To check if you are detached, use the "where" command:
WHERE <SP> USERNAME CR

If you are detached, instead of logging in, type:
ATT <SP> USERNAME <SP> PASSWORD <SP> CR
<CTRL-C>
NO RAISE [BBNB Only]
TERMINAL LINEPROCESSOR
NLS CR [to start over again]

Line Processor Trouble:

*Lineprocessor Halts:

Shows as flashing status lights and indicates transmission or program error.

If you are in DNLS when the lights flash, wait for the LPR light on the EP connection to stop flashing, then push in the System Reset button. If only 1 light is on after you have reset, you are not in DNLS.

*Host Crashes:

TIP sends the message HOST NOT RESPONDING.

Type <CTRL-C> perodically

[The TIP may maintain the connection but it doesn't tell you when the system is back.]

When you get an @ from Tenex, repeat the Login to TENEX and NLS

OTHER AVAILABLE COURSES:

INTERMEDIATE TNLS

This is the third course and level of expertise, and represents significant experience with the system. The Programs and Useroptions subsystems are introduced as well as Output Processing for printer formating. Topics introduced in the Third TNLS Course are useful in DNLS.

EXAMPLE OF STRUCTURE:

```
< BAIR, MENU.NLS;1, >, 28-JAN-75 17:29 JHB ;;;;
1 SOUP
   1A VEGETABLE
   1B CREAM OF MUSHROOM
2 ENTREE
   2A  FRIED CHICKEN
   2B  SALMON
      2B1 WITH CREAM SAUCE
   2C  PRIME RIBS
3 DESSERT
   3A  PIE
      3A1  APPLE
         3A1A A LA MODE
      3A2  BLUEBERRY
   3B  ICE CREAM
      3B1  VANILLA
      3B2  PEPPERMINT
      3B3  MAPLENUT
      3B4  CHOCOLATE
4  BEVERAGE
   4A  TEA
   4B  COFFEE
```

DEFINITIONS FOR THE COURSE OUTLINE

COMMANDS = You type some characters to tell the computer what to do. The characters you type are represented by the uppercase letters in each "command word".

<SP> = You type a space.

Uppercase words = You type in the appropriate information for that command phrase, e.g., BUG/TYPEIN.

CTRL = hold down the control (CTRL) key WHILE typing the specified character.

OK, B, or CA = you type a Command Accept.

<esc> = the ESC or escape key on your terminal (sometimes labeled "alt mode").

TYPEIN and BUG/TYPEIN = a string of characters from the keyboard, or keyset ending with an OK, prompted by T:. [TYPEIN has a special form when a FILE ADDRESS or Link or Ident is called for (You can tell from the noise words)].

LEVELADJUST: specifies level relative to addressed statement -- type any number of u's [for up], d's [for down], or an OK for the same level, prompted by L:.

EDGE: Imaginary lines which can divide the display into as many as 8 parts, allowing a different view in each part.

VIEWSPECS: a string of one or more viewspec characters followed by OK, prompted by V: [type OK if no viewspecs are to be entered]

STRING: Character or Word or Text, prompted by C:

STRUCTURE: Statement or Branch or Group or plex, prompted by C:

BUG: hit OK on mouse when cursor is positioned. Prompted by B:

CURSOR: The moving dash on display when in DNLS

BUG/TYPEIN: A string of characters from the keyboard, keyset, or bugged with the mouse, ended by an OK, prompted by B/T.

<CTRL-N>: Used in DNLS to specify null or no CONTENT

0 < REPORT, OUTLINE.NLS; 1, > 1-FEB-74 08:30

BRANCH 1

1 INTRODUCTION

1a (P)REFACE:

1b [BACKGROUND]:

1b1 PREVIO[US REPOR]T

1b2 REPORT ON(O)PREVIOUS INTRODUCTION

1b3 "BACKGROUND [ON] PREVIOUS REPORT"

1c [REQUIREMENTS:] } BRANCH 1c

2 PERSPECTIVE:

2a DEFINITION:

2b ASPIRATIONS: } BRANCH 2b

2b1 FUTURE }

3 ENVIRONMENT:

3a ENVIRONMENT DEFINED:

3b RELEVANT FACTORS IN ENVIRONMENT:

3b1 DEFINITIONS OF RELEVANT ENVIRONMENTS

3b2 REPORTS ON OTHER RELEVANT DEFINITIONS

3b3 FACTORS IN RELEVANCE

3c REQUIREMENTS:

KEY:

CHARACTERS ARE ENCLOSED IN CIRCLES

WORDS ARE ENCLOSED IN RECTANGLES

TEXT IS ENCLOSED IN SQUARE BRACKETS

STATEMENTS ARE ENCLOSED IN QUOTES

FIGURE 1

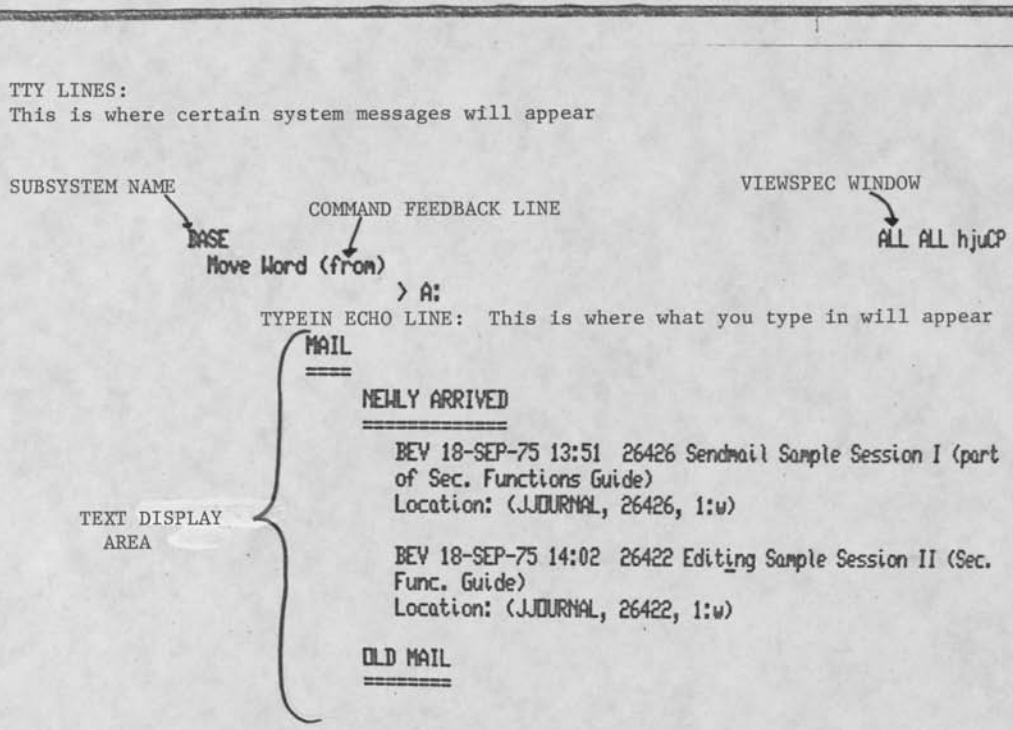


FIGURE 3

DNLS DISPLAY

FGB 7-JAN-76 14:20 34206

KWAC Action Items

Please read and comment as I will be using this to formulate a KWAC position paper towards the end of January.

KWAC Action Items

People, 1

The following is a list of the action items we covered at the last meeting. ARC (in particular Bud Pine) has agreed to address them. Volunteers are needed to follow up on some of them for the KWAC community. These items will form the basis of a KWAC positon paper. 1a

Please let me know if: 1b

(1) Any of you would care to track some of the items and 1b1

(2) If I've left out or misinterpreted any items of interest. 1b2

Regards, 1b2a

Frank 1b2b

I. COSTS & RESOURCES 2

1. SRI's Pricing Policies 2a

These policies were adequately covered during the meeting. Any departures will be negotiated by the individual organizations. 2a1

2. Obtaining Additional Resources on a Short Term Basis 2b

(a) It will be difficult to provide a "temporary slot" (or equivalent) in the near future. 2b1

(b) At Office-1, windfall will be distributed equally among the subscribers. 2b2

(c) Priority for demos can be obtained by arrangement with Bud Pine. 2b3

II. PERFORMANCE 3

1. Tymshare 3a

(a) A written statement of duties & responsibilities will be obtained from Tymshare by SRI and distributed to KWAC. 3a1

(b) Retreival of archived files will be speeded up. The range of acceptble retrieval times is betweem 30 mins. & 3 hours (My notes don't show if a target retrieval time was ever decided upon. Can anybody help on this?). 3a2

(c) The Tymshare operators are being trained in Tenex & some elements of NLS. 3a3

(d) SRI will try to insure more consistent notification to network users of expected recovery time when Office-1 is unavailable (via the message that can be put out thru the NCC). 3a4

2. Reliability of Office-1 3b

(a) Monthly reports of system reliability (using currently available data) will be sent to KWAC. 3b1

3. Upgrade of Office-1 3c

(a) Duane Stone will put an unfunded option in his contract calling for doubling the core at Office-1. 3c1

III. NLS/OFFICE-1 4

1. Help 4a

(a) No action was taken with respect to the on-line Help subsystem. The consensus seemed to be that, if the system remains, it should not be expanded. 4a1

(b) Any additional help facilities (e.g., an on-line terminal) were vetoed due to the additional funds involved. 4a2

2. Batch 4b

(a) The Batch subsystem is now available at Office-1 on a "use at your own" risk basis. It is expected to become fully operational shortly. 4b1

3. DEX 4c

DEX works over a direct line but, as yet, no NLS user has tried it over the ARPANET. 4c1

4. Screen Refresh Problem 4d

(a) Jim Norton is having someone look into the algorithm and write it up for our benefit. 4d1

(b) Dave Hopper & Karolyn Martin would be the ones to redo this algorithm but there are other bugs which might be more profitable to fix. 4d2

5. Ctrl T 4e

(a) This command has been changed to give, instead of load averages, the % CPU entitled, % CPU received , etc. 4e1

6. Detatched Jobs 4f

(a) The Autologout job will be reinstituted at Office-1 so that detatched jobs will be automatically logged out. 4f1

7. Compatability with other systems 4g

(a) Discussion of this issue was deferred to a later date. 4g1

IV. SERVICES 5

No action taken on the following: 5a

- user statistics 5a1

- applications programmer 5a2

- training 5a3

V. DOCUMENTATION 6

No action taken on this topic 6a

VI. AID DIRECTORY 7

This will be a directory used to exchange information among architects and others. It was conceived by Glenn Sherwood of SRI who has taken responsibility for setting up the initial implementation. 7a

< GJOURNAL, 34206.NLS;1, >, 7-JAN-76 18:35 XXX ;;;; Title: Author(s):
Frank G. Bridnoll7FGB; Distribution: /AID([ACTION]) ;
Sub-Collections: NIC AID; Clerk: FGB;

This memo records some miscellaneous information about the Army Materiel Command (AMC) pertaining to software activities that potentially could participate with us in "Software Engineering" application and development activities.

During a visit I made to AMC Headquarters on 8 Sep 75, I brought up this topic. Ron Uhlig gave me the folowing information about two particular "Field Activities" reporting to John Gilbert (Ron's boss), which also are two of the initial seven AMC sites aimed at Office-1 use:

Automated Field Logistics Management Systems Agency (ALMSA), in St. Louis.

Develops and maintains standard business systems for all of the AMC commodity commands. About 800 people involved in designing, writing and installing software.

Two 360/65s, 16 2314s (about 500 megabytes), 40 to 60 mag-tape drives -- whole configuration replicated at each of six commodity commands.

Apparently one large "standard software system" cost $100 million to develop.

Logistics Systems Support Agency (LSSA), in Pennsylvania (I believe)

Automating standard business system for sizteen AMC depots, involving like 15 CDC 3300s. Something like $20 million in development here.

Note: See (IJOURNAL, 27291,2:ebtz) for list of eight software people from two AMC locations to whom Pam Allen taught NLS courses for last month -- six of them were from LSSA, two from Ft. Belvoir.

Dave Grobstein and Picattiny Arsenal

On 22 Jul 75, Ron Uhlig and Dave Grobstein visited ARC. Dave is Director of Computer Center at Picattiny Arsenal, Dover, N.J. an R&D Center under the Armament Command within the AMC. His line responsibility comes down this channel (through the ARMCOM MIS Directorate), but there is a staff linkup to John Gilbert (Ron's boss) who coordinates all of the data processing within the whole AMC.

Dave has worked with Ron in the past, co-authoring papers on network economics, rationing schemes, etc. He is sharp, knowledgeable, and very much interested in network resouce sharing.

I had some sndmsg interchanges with Dave, and then about half an hour on the phone on 20 Oct 75 (when I was in Washington). Says the Arsenal has about 5000 people, and is an R&D center for the Armament Command. Their work is scientific, as opposed to businss systems -- Dave pointed out that there were four major scientiic-like activities:

Avionics Systems Command, St. Louis (? may have this wrong);

Edgewood Arsenal;

White Sands Missile ...;

Picattiny Arsenal.

Some of thes places have computer facilitis serving more than their internal needs (Picattiny is such, I gathered).

Data processing is divided formally under Gilbert into Scientific and Logistic. Grobstein says that he serves on a Scientific Computer Steering Committee -- and also represents his people (?the Armament Command) on the logistics-related community.

I described in general our interest in getting exploratory application going in the areas of technical documentation and software engineering, both of which could be tied into his own, local activity; was he interested? "Yes, but can't do anything about it for perhaps six months -- undergoing giant re-organization that will keep me heavily absorbed." (Six months -- April 75)

He volunteered that he had been wanting to expose his programmers to structured programming, saying that they have sort of been out of the main stream of modern programming. Talked as though he wanted to budget and plan for doing something to upgrade his staff's programming methods -- for instance, he'd like to get a good survey course imported. Said that for the labor costs involved, it was smuch more effective to import the teachers than export the students.

We talked of NSW, of RADC's increased focus on software productivity, structured programming, etc.

I brought back a useful descriptive document from my September visit with Ron Uhlig. It is stored in the XDOC collection:

A Guide to the ARMY MATERIEL COMMAND Organization & Missions, July 1974 (Out of date now) (XDOC -- 34225,)
(Includes a large Organization Chart)

< MJOURNAL, 34252.NLS;1, >, 14-JAN-76 12:41 XXX ;;;; .HJOURNAL="JHB 13-JAN-76 10:59 34252"; Title: .H1="Quick Overview of ARC-ADG Near Term Efforts"; Author(s): James H. Bair/JHB; Distribution: /LJM([ACTION]) JMB([ACTION]) RLL([ACTION]) POOH([ACTION]) ARC-APP([INFO-ONLY]) BEV([INFO-ONLY]) JBP([INFO-ONLY]) ; Sub-Collections: SRI-ARC ARC-APP; Clerk: JHB; .IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PES; Origin: < BAIR, ADG-DOC-STAT.NLS;3, >, 13-JAN-76 10:39 JHB ;;;; LJM JMB RLL POOH ARC-APP BEV JBP####;

Quick Overview of ARC-ADG Near Term Efforts

- Write and release list of CHANGES IN NLS 8.5 --LJM
  - 2 VERSIONS: 1) announcing trial usage by architects,
  - 2) announcing general release for all clients
- Rewrite GRPSTAT description -- JMB
  - To reflect changes and make more readable/understandable by non-Tenex experts
- Revise and release BATCH Facility Userguide --JMB
- Review and release TI ASR DEX Userguide --JHB --JMB
- Update and print AKW Seminar Agenda --LJM
  - and additional mailings to attendees
- Expand & release general Description of Utility Client Applications --JCN & JHB
- Finish MIT Application Description --JHB
  - Includes Tutorial on Process Commands which will issued separately.
- Help POOH write GRAPHICS AND PROOF Initial Userguide and release --JHB --JMB
  - A fast but comprensive (quick and dirty) description based on Help writeup
- Write Applications Descriptions of AMC and RADC --JHB
- Write Application Descriptions of ARPA and NSRDC --RLL
- Revise and release IR & D Technology Transfer Report (to KWAC) --JHB
- Revise and release Third Course --JHB & LJM
- Revise, revise (and improve?) AKW Seminar Presentations --JHB RLL (PAW2 JMB LJM DCE JCN JBP)
  - More visual aids, demos, etc.
- Work with Bev et.al. on ARC Long Term Documentation Issues --JHB
- ** Trip reports and similar items (e.g. demos) not included; more later....

< AJOURNAL, 34276.NLS;1, >, 18-JAN-76 16:27 XXX ;;;; Title:
Author(s): Gwen C. Edwards/GCE; Distribution: /DAP([ACTION]) RWH([
ACTION]) RA3Y([ACTION]) SKC([ACTION]) JHB([ACTION]) ;
Sub-Collections: NIC; Clerk: GCE; Origin: < GEDWARDS,
QUESTIONNAIRE.NLS;4, >, 16-JAN-76 13:02 GCE ;;;;####;

GCE 16-JAN-76 13:13 34276

Here iit is...in all its entirety. Would appreciate any comments / criticisms you might have within the next week. THanks... Gwen

SECTION 1 1

3. Do you yourself type the material into the NLS system, does someone type it for you, or do both occur? 2

TYPE IN MYSELF ... Answer Qn 4
HAVE TYPED IN ... Answer Qn 5
BOTH OCCUR ... Answer Qn 6 2a

QUESTION 4 3

4. a) Is the information that you input your own, someone elses or both? 3a

Own only ... Go to Section 2
Someone elses only ... Skip 3b, go to 3C
Both ... Continue with 3B 3a1

When you input information for yourself, what sort of information do you input? 3b

unwritten thoughts / ideas, e.g. composing on-line 3b1

hand written rough draft 3b2

editing changes to text 3b3

messages/mail 3b4

file searches or other information retreival requests 3b5

other (please describe) 3b6

What type of information do you input for other people? 3c

unwritten thoughts (via dictation, etc.) 3c1

hand written rough draft 3c2

editing changes to text 3c3

messages/mail 3c4

file searches or other information retreival requests 3c5

other (please describe) 3c6

How many other peopl do you input material for? 3d

what is the relationship of this (these) person(s) to you (eg. boss, other professional, etc.) 3d1

Go to Section 2 3e

Question 5 4

5. a) Do you know how to imput material yourself? 4a

YES ...
NO ... Go to Section 2 4a1

b) If YES, what are the main reasons why you have chosen to have someone else input your material for you. (Please check all that apply. If one choice is predominant, check twice.) 4b

I don't have time to use the system myself. 4b1

Using the system directly, i.e., typing at a terminal, is not compatable with my professional role 4b2

I dislike working on-line. (Please explain) 4b3

other (please explain) 4b4

Go to Section 2 4b5

Question 6 5

What type of material do you input yourself? 5a

unwritten thoughts ,ideas, e.g. composing on-line 5a1

hand written rough draft 5a2

editing changes to text 5a3

messages/mail 5a4

file searches or other information retreival requests 5a5

other (please describe) 5a6

What type of material do you ask someone else to input for you? 5b

unwritten thoughts (via dictation, etc.) 5b1

hand written rough draft 5b2

editing changes to text 5b3

messages/mail 5b4

file searches or other information retreival requests 5b5

other (please describe) 5b6

When you choose to use a support person to input material for you, upon which of the following factors is your decision dependent? (If more than one is applicable, please rank accordingly: 1= most applicable; 2= next most applicable; etc.) 5c

I use support staff for all tasks checked above as much as support staff is available 5c1

I use support staff mostly when my own time is constrained 5c2

I use support staff mostly when the job is urgent or "rush". 5c3

whenever performing the task does not fit with my professional role 5c4

Comments: 5c5

SECTION 2 - OVERALL EVALUATION OF THE NLS SYSTEM 6

6a

PART A: GENERAL IMPRESSION OF N.L.S. 6b

Which statement best describes the incentive (or lack thereof) within your group for your use of N.L.S.? (Please check only one) 6b1

required to use it 6b1a

requested to use it 6b1b

...
Am free to use it as I choose ...
Would rather I didn't use it ...
Other (specify)..... 6b1c

Were you actively involved in the decision to subscribe to NLS? 6b2

yes 6b2a

no 6b2b

Which statement best describes your initial reaction when N.L.S. was first introduced? (Please check only one) 6b3

I thought it would be useless ... 6b3a

I thought it might be useful for others but not really for me personally ... 6b3b

I was skeptical about it but willing to give it a try ... 6b3c

I was basically indifferent or neutral 6b3d

I thought it would have certain limited but worthwhile uses for meSplit;... 6b3e

I thought it would be very useful in many respects. 6b3f

I thought it would revolutionize my work/communications processes. 6b3g

When NLS was first introduced to you/your organization, I was: (Please check only one) 6b4

1) anxious to learn the commands - to use it yourself ... 6b4a

2) willing to learn to use it yourself ... 6b4b

3) not willing to learn to use it yourself ... 6b4c

4) other (please explain) 6b4d

Did your attitude re: above change over time? 6b5

yes 6b5a

no 6b5b

If yes, please describe: 6b5c

Which statement best describes your present reaction to N.L.S.? (Please check only one) 6b6

I think it is useless and should be discontinued ... 6b6a

I think it has its uses for others but not for me personally ... 6b6b

I am skeptical but am giving it a try ... 6b6c

I am basically indifferent or neutral ... 6b6d

I think that it has certain worthwhile uses for me. 6b6e

I think it is very useful in many respects. 6b6f

I think it is revolutionizing my work/communications processes. 6b6g

Are there any aspects of the NLS system that you particularly like? (Please describe) 6b7

Are there any aspects of NLS that you particularly dislike ? (Please describe) 6b8

What other capabilities can you imagine you would like to have, given an ideal system? 6b9

If you had to choose one, which of the following would you say best describes the NLS package? (Please check only one) 6b10

1) a communications medium ... 6b10a

2) a text editor ... 6b10b

An information retreival system 6b10c

3) a tool augmenting a variety of your every day tasks ... 6b10d

4) a tool augmenting your overall intellectual, thought, and/or organizational processes ... 6b10e

PART B: USAGE and RELATED ATTITUDES 6c

How long have you been using NLS? 6c1

.....years and .....months 6c1a

In an average week, how many hours do you spend using NLS? 6c2

.....hours per week 6c2a

Over the last year, has your usage of NLS decreased, increased or stayed about the same? 6c3

increased 6c3a

decreased 6c3b

stayed about the same 6c3c

What is the approximate size of your directory in terms of the number of files? .....files 6c4

in terms of the number of computer pages?
.....pages 6c4a

Please describe the types of purposes for which you create NLS files, (e.g., you may create files for reports, letters, reminders, lists, budgets, etc.) Try to be as specific as possible . 6c5

Do you ever have difficulty accessing the NLS system? 6c6

yes 6c6a

no go to #.... 6c6b

If yes, approximately how many tmes a week? 6c6b1

...../week 6c6b1a

What are the main reasons: (Please rank in order of frequency, if more than one is applicable) 6c6b2

"NO NEW LOGINS ALLOWED" 6c6b2a

communications lines/ network down 6c6b2b

computer down 6c6b2c

no available terminal 6c6b2d

other 6c6b2e

Would you like to increase your usage of NLS, decrease your usage, or leave it about the same? 6c7

increase my usage (go to 6c7d) 6c7a

decrease my usage (go to 6c7e) 6c7b

leave it about the same (go to 6c8) 6c7c

Is there anything that prevents you from increasing your usage? 6c7d

yes 6c7d1

no (go to 6c8) 6c7d2

If yes, what? 6c7d2a

access problems 6c7d2a1

slot limitations 6c7d2a2

dependent on clerical staff and they are too busy 6c7d2a3

other (please explain) 6c7d2a4

Is there anything that prevents you from decreasing your usage? 6c7e

yes 6c7e1

no 6c7e2

If yes, what? (Please check the ONE most applicable) 6c7e2a

general organizational ("political") pressures 6c7e2a1

current usage is part of job description/definiton 6c7e2a2

I need the system to meet current work load requirements 6c7e2a3

other (please explain) 6c7e2a4

In general, what sort of training in using NLS did you receive? (If more than one applies, place a 1 beside the most applicable statement, a 2 besides the next most applicable, etc.) 6c8

Formal program ...
From other employee in charge of training ...
By other user of NLS ...
Picked it up on my own ...
Not applicable - was not interested in learning NLS for my direct use ...
Other (specify)..... 6c8a

Which of the following statements best describes the ways in which you use NLS? If you use the system in more than one way, place a 1 beside the most frequent method, a 2 beside the next most frequent, ect. 6c9

I give handwritten copy to someone who gets hard-copy for me to work with. ... 6c9a

I give handwritten copy to someone who inputs it into the system and from there I work with the material on-line via a terminal. ... 6c9b

I input the material myself, get a hard copy and work with that. ... 6c9c

I input the information, get a hard copy, and then work on-line. 6c9d

I input the material myself and work on it through a CRT terminal, rarely, if ever using hard copy in the process.... 6c9e

I dictate the information, someone else types it in and I get a hard-copy to work from 6c9f

I "sign on" primarily to send and receive messages. 6c9g

OTHER (please describe) ... 6c9h

If NLS were removed from your work environment and/or NLS were no longer available for your use, what would be your reaction? 6c10

PART C: IMPACTS AND RELATED ATTITUDES 6d

Now consider how NLS has affected your workstyle. The following questions will probe for possible impacts NLS has had on 1) you as an individual 2) on the group as an organizational unit 3) on other group members. 6d1

On a scale of 1-5, how would you say the system has affected your workstyles in the following areas; how do you feel about the changes? 6d1a

1) flexibility in hours 6d1a1

decreased...1...2...3...4...5...increased 6d1a1a

Pleased...1...2...3...4...5...Dipleased 6d1a1b

2) compatability of system - how compatable is the system to your normal working/writing/thinking/organizing style? 6d1a2

compatable...1...2...3...4...5...incompatable 6d1a2a

Pleased...1...2...3...4...5...Dipleased 6d1a2b

2) flexibility of system - how flexible is the system to

meeting your normal working/writing/thinking/organizing style? 6d1a3

flexible...1...2...3...4...5...inflexible 6d1a3a

pleased..1...2...3...4...5...displeased 6d1a3b

flexibility of self - how flexible do you feel you have had to be to permit a blending of your working/writing/organizing/(managing) with the capabilities/requirements of the system? 6d1a4

flexible...1...2...3...4...5...inflexible 6d1a4a

pleased..1...2...3...4...5...displeased 6d1a4b

3) overall work efficiency / productivity 6d1a5

increased...1...2...3...4...5...decreaseddecreased 6d1a5a

Pleased...1...2...3...4...5...Dipleased 6d1a5b

overall quality of your work 6d1a6

increased...1...2...3...4...5...decreased 6d1a6a

Pleased...1...2...3...4...5...Dipleased 6d1a6b

accessobo;oty of your work to others 6d1a7

increased...1...2...3...4...5...decreased 6d1a7a

Pleased...1...2...3...4...5...Dipleased 6d1a7b

your professional image 6d1a8

increased...1...2...3...4...5...decreased 6d1a8a

Pleased...1...2...3...4...5...Dipleased 6d1a8b

amount of time spent communicating face to face with your associates 6d1a9

increased...1...2...3...4...5...decreased 6d1a9a

Pleased...1...2...3...4...5...Displeased 6d1a9b

On a scale of 1-5, how would you rate the system's reliability? (do not include here problems of access, but

system reliability once you are on-line).

6d1b

reliable...1...2...3...4...5...unreliable 6d1b1

annoying...1...2...3...4...5...not annoying 6d1b2

How does system reliability affect your work mode on NLS? 6d1c

I am concerned, take precautions such as updating, files, etc. 6d1c1

b) I keep hard copies of everything important - that I wouldn't want lost. 6d1c2

I simply don't use the system fo critical items that I must be able to access on-demand. 6d1c3

I do not let it affect my work mode. 6d1c4

other (please explain) 6d1c5

COMMENTS: 6d1c5a

On a scale of 1-5, how would you rate the privacy your work via NLS? 6d1d

private...1...2...3...4...5...public 6d1d1

annoying...1...2...3...4...5...not annoying 6d1d2

How does the privacy aspect of NLS affect you use of the system? 6d1e

I am concerned, take precautions such as changing my password when advisable, using privacy codes, etc. 6d1e1

I simply don't use the system for work of a very confidential nature. 6d1e2

I do not let the privacy aspect affect my use of the system 6d1e3

Not Applicalbe - I don't work with material of a confidential nature. 6d1e4

other (please explain) 6d1e5

COMMENTS: 6d1e5a

Please describe the most significant new task(s) you perform or work modes you have as a result of NLS: 6d1f

Are there any tasks or work modes that have been eliminated as a result of your use of NLS? 6d1f1

Can you think of any other ways in which your use of NLS has affected your work habits or behavior? For example, are there other ways in which you have adapted yourself to meet the system's requirements, or other ways in which you have made or make the system adapt to you? 6d1g

Do you have any particular reaction to such change? (Please describe) 6d1g1

SECTION III : EVALUATION OF SPECIFIC CAPABILITIES 7

In the following section, we are going to describe 6 specific capabilities of NLS. Would you please read the description of each , identify yourself as a user or non user of each capability, and answer the appropriate questions? 7a

NLS FOR COMMUNICATION PURPOSES 8

NLS can be used to communicate with other users of NLS (or anyone with access to Office 1). Do you use NLS in this capacity - to exchange messages, files, or to link with other users? 8a

Yes , I use NLS exclusively for my computer -based communications..... move to question 8E 8a1

Yes - and I also use other computer system(s) for communications purposes.....go to question 8B. 8a2

No - but use other computer systems to do the same thing..... go to question 8C. 8a3

No - and use no computer system to do this.....go to question 8D. 8a4

What are these other computer systems? 8b

Wddhat factors determine which system you use? 8b1

(Go on to 8E) 8b1a

What other systems do you use? 8c

Why did you choose not to use NLS for this capability? 8c1

was not aware the capability existed 8c1a

no need 8c1b

too hard to use 8c1c

other: please describe 8c1d

(Go to question number 11) 8c2

How do you usually accomplish this task? 8d

Why did you choose not to use NLS for this capability? 8d1

was not aware the capability existed 8d1a

no need 8d1b

too hard to use 8d1c

other: please describe 8d1d

(Go to question number 11) 8d2

There are basically three types of interpersonal communications capabilities in NLS (or in conjunction with): exchanging of messages, the exchanging of files or documents, and linking in real time. Keep in mind that each of these three capabilities can be executed in a number of ways, and in the case of the first two, more that one program or subroutine can be selected to accomplish the particular objective. Which of the three CAPABILITIES do you make use of? Please answer the appropriate questions followng each capability that you use. 8e

The exchange of messages ...please answer questions .....at 8E4 8e1

the exchange of documents, reports, or files ....please answer questions.....at 8E5 8e2

linking in real time to another user ....please answer questions .....at 8E6 8e3

THE EXCHANGE OF MESSAGES 8e4

what methods do you predominately use to exchange messages. (e.g., do you use "send message" , "msg", the journal mail

system, etc.) Be sure to keep in mind the distinction between sending messages and sending NLS files. (If you are uncertain of the name of the message system, please indicate instead the commands you use to execute a message.) 8e4a

For what purposes do you exchange messages? Please check all that apply, ranking them according to applicability 1=most frequently applicable;3=least frequently applicable, etc. 8e4b

administrative purposes, e.g. notices of meetings, agenda etc. 8e4b1

making requests, project control 8e4b2

personnel management purposes, e.g. requests to subordinate 8e4b3

information exchange or problem solving re a particular work related topic 8e4b4

generally keeping in touch with professional contacts or friends. 8e4b5

other: please describe 8e4b6

Do you send/receive messages primarily to/from users within your own organization or external to your organization? 8e4c

mostly internal 8e4c1

mostly external 8e4c2

neither is predominant 8e4c3

Which outside organizations do you exchange messages? (Please list) 8e4d

How often do you use this capability in an average week? 8e4e

.....times each week 8e4e1

Would you say this usage is/will be increasing, decreasing, or staying about the same? 8e4e2

increasing 8e4e2a

decreasing 8e4e2b

staying about the same 8e4e2c

How easy was it to learn how to use this capability? 8e4f

Very easy...1...2...3...4...5...Very hard 8e4f1

Any comments related to training or ease of use? 8e4f2

What has been the single most significant impact of your using this capability on your work routine 8e4g

How would you say the use of the message sending capabilities has affected your communications patterns; how do you feel about the change? 8e4h

amount of communication with people outside of your group/organization? 8e4h1

increased...1...2...3...4...5...decreased 8e4h1a

Pleased...1...2...3...4...5...Displeased 8e4h1b

within your organization: 8e4h2

5) overall amount of communication with your peers 8e4h2a

increased...1...2...3...4...5...decreased 8e4h2a1

Pleased...1...2...3...4...5...Displeased 8e4h2a2

7) amount of communication with your superiors 8e4h2b

increased...1...2...3...4...5...decreased 8e4h2b1

Pleased...1...2...3...4...5...Displeased 8e4h2b2

8) amount of communication with subordinates 8e4h2c

decreased...1...2...3...4...5...increased 8e4h2c1

Pleased...1...2...3...4...5...Displeased 8e4h2c2

What was the primary mode/ medium you used to accomplish an analogous task before you started using NLS? 8e4i

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts, or suggestions, for change? 8e4j

THE EXCHANGE OF DOCUMENTS, REPORTS, OR FILES 8e5

What methods to you use to execute this capability, e.g., do you use the "send mail" subsystem, and/or other methods? Again, if you can not recall the name of the particular routine, please indicate the commands you use. 8e5a

Do you send/receive files primarily to/from users within your own organization or external to your organization? 8e5b

mostly internal 8e5b1

mostly external 8e5b2

neither is predominant 8e5b3

With which organizations other than your own do you exchange files? (Please list) 8e5c

How often do you use this capability in an average month? 8e5d

.....times each month 8e5d1

Would you say this usage is/will be increasing, decreasing, or staying about the same? 8e5d2

increasing 8e5d2a

decreasing 8e5d2b

staying about the same 8e5d2c

How easy was it to learn how to use this capability? 8e5e

Very easy...1...2...3...4...5...very hard 8e5e1

Any comments related to training or ease of use? 8e5e2

What has been the single most significant impact of your using this capability on your work routine 8e5f

communications with other people in your group 8e5f1

communications with people outside of your group 8e5f2

How would you say the use of the file sending capabilities has affected your communications patterns; how do you feel about the change? 8e5g

amount of communication with people outside of your group/organization? 8e5g1

within your organization: 8e5g2

5) overall amount of communication with your peers 8e5g2a

decreased...1...2...3...4...5...increased 8e5g2a1

Pleased...1...2...3...4...5...Dipleased 8e5g2a2

7) amount of communication with your superiors 8e5g2b

decreased...1...2...3...4...5...increased 8e5g2b1

Pleased...1...2...3...4...5...Displeased 8e5g2b2

8) amount of communication with subordinates 8e5g2c

decreased...1...2...3...4...5...increased 8e5g2c1

Pleased...1...2...3...4...5...Displeased 8e5g2c2

what was the primary mode/ medium you used to accomplish an analogous task before you started using NLS? 8e5h

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts, or suggestions, for change? 8e5i

LINKING IN REAL TIME TO ANOTHER USER 8e6

Describe the most typical situation in which you use this capability: 8e6a

For what other purposes do you link to another user? 8e6a1

How often do you use this capability in an average week? 8e6b

.....times each week 8e6b1

Would you say this usage is/will be increasing, decreasing, or staying about the same? 8e6b2

increasing 8e6b2a

decreasing 8e6b2b

staying about the same 8e6b2c

while linked, do you ever share files? 8e6c

yes 8e6c1

if yes, how often, with which organizations, for what purposes? 8e6c1a

.....times per month 8e6c1a1

organizations: 8e6c1a2

purposes / types of functions: 8e6c1a3

go to 8e6d 8e6c1a3a

no 8e6c2

if no, why? 8e6c2a

Why did you choose not to use NLS for this capability? 8e6c2a1

was not aware the capability existed 8e6c2a1a

no need 8e6c2a1b

too hard to use 8e6c2a1c

other: please describe 8e6c2a1d

Do you split the screen to share more than one file or part of a file(s)? 8e6d

yes 8e6d1

if yes, how often? 8e6d1a

no 8e6d2

if no, why? 8e6d2a

Why did you choose not to use NLS for this capability? 8e6d2a1

was not aware the capability existed 8e6d2a1a

no need 8e6d2a1b

too hard to use 8e6d2a1c

other: please describe 8e6d2a1d

What has been the single most significant impact of your using this capability on your:work routine 8e6e

What has been the single most significant impact of your using this capability on your communications with others? 8e6f

What was the primary mode/ medium you used to accomplish an analogous task before you started using NLS? 8e6g

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts, suggestions, etc.? 8e6h

Are there any other ways to communicate through NLS that you know of (or have invented) and use? Please describe: 8e7

NLS FOR INFORMATION RETREIVAL 9

NLS can be used in many ways as an information retreival tool, to retreive old messages, journal items, copies of letters, archived material, etc. Do you use this capability either directly or indirectly in NLS? 9a

Yes I use NLS exclusively for my computer-based information retreival purposes..... move to question 9B 9a1

yes, and also retrieve information through other computer-based systems. .....go to 9A5 9a2

No - but use other computer systems to do the same thing..... go to question 9A3. 9a3

No - use no computer system to do this.....go to question 17A4. 9a4

What are these other computer systems? 9a5

What factors determine which system you use? 9a5a

(go on to question #17B) 9a5a1

What other systems do you use? 9a6

were you aware that this capability existed in NLS? 9a6a

Yes 9a6a1

No...go to next capability description, question #18 9a6a2

Are there any specific reasons for your choosing not to use NLS? 9a6b

(go to next capability description, question #18) 9a6b1

How do you usually accomplish this task? 9a7

Why did you choose not to use NLS? 9a7a

was not aware of the capability 9a7a1

no need 9a7a2

too hard to use 9a7a3

too poorly documented 9a7a4

other: please describe 9a7a5

There are several types of information and correspondingly several different ways in which one can retreive information in NLS. Which of the following do you use? 9b

Retreiving previously read messages .... answer questions ... 9b1

retreiving previously read journal items 9b2

retrieving files from the on-line library of journal documents 9b3

archived material 9b4

RETRIEVING PREVIOUSLY READ MESSAGES: 9b5

How often do you use this capability? .....times per month 9b5a

For what purposes / in what types of situations? 9b5b

When you use this capability, who inputs the information, i.e. who executes the NLS commands? 9b5c

myself 9b5c1

someone else .... (go to question #17B2) 9b5c2

both 9b5c3

How easy was it to learn how to use this capability? 9b5d

Very easy...1...2...3...4...5...Very hard 9b5d1

Any comments? 9b5d2

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts, suggestions, etc.? 9b5e

RETREIVING PREVIOUSLY READ JOURNAL ITEMS 9b6

How often do you use this capability? .....times per month 9b6a

For what purposes / in what types of situations? 9b6b

When you use this capability, who inputs the information, i.e. who executes the NLS commands? 9b6c

myself 9b6c1

someone else .... (go to question #17B2) 9b6c2

both 9b6c3

How easy was it to learn how to use this capability? 9b6d

Very easy...1...2...3...4...5...6...7...Very hard 9b6d1

Any comments? 9b6d2

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts, suggestions, etc.? 9b6e

RETREIVING FILES FROM THE PUBLIC LIBRARY OF JOURNAL DOCUMENTS 9b7

How often do you use this capability? .....times per month 9b7a

For what purposes / in what types of situations? 9b7b

When you use this capability, who inputs the information, i.e. who executes the NLS commands? 9b7c

myself 9b7c1

someone else... 9b7c2

both 9b7c3

How easy was it to learn how to use this capability? 9b7d

Very easy...1...2...3...4...5...Very hard 9b7d1

Don't know 9b7d2

Any comments? 9b7d3

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts, suggestions, etc.? 9b7e

RETRIEVING ARCHIVED MATERIAL 9b8

How often do you use this capability? .....times per month 9b8a

For what purposes / in what types of situations? 9b8b

What is the average turn around time between your issuing

the request and receiving the document back on-line?
_____hours 9b8c

When you use this capability, who inputs the information, i.e. who executes the NLS commands? 9b8d

myself 9b8d1

someone else.... 9b8d2

both 9b8d3

How easy was it to learn how to use this capability? 9b8e

Very easy...1...2...3...4...5...Very hard 9b8e1

Don't know 9b8e2

Any comments? 9b8e3

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts, suggestions, etc.? 9b8f

NLS AS A SHARED WORK SPACE 10

The capability to access other people's files and share, in a sense, the same work space, again has several dimensions. Do you ever access other people's files? 10a

yes 10a1

no 10a2

if no, why? 10a2a

go to next capability, question #11 10a2b

For which functions /tasks do you access other people's files? (If more than one applies, please rank as to frequency of use:
1 = most frequent
2 = next most frequent, etc.) 10b

for reading/reviewing other's work 10b1

to write on other people's files 10b2

to copy statements, sections of files, or perhaps entire files from one person's directory into your own 10b3

other (please specify): 10b4

What would you say are the purposes of your accessing other peoples files? (If more than one applies, please rank as to frequency of use:
1 = most frequently used
2 = next most frequently used, etc.) 10c

for management or supervisory purposes 10c1

for general interest / a way of keeping informed 10c2

for purposes of joint or collaborative authorship 10c3

for purposes of transfering information from someone else's file into your own, perhaps later to quote or reference in your own document. 10c4

other (please specify): 10c5

Do you access files of those individuals: 10d

within your own organization ?.... 10d1

yes 10d1a

with approximately how many different individuals? 10d1a1

no 10d1b

external to your organization?... 10d2

yes 10d2a

which organizations? (please list) 10d2a1

no 10d2b

How often do you use this capability in an average month? 10e

.....times each month 10e1

Would you say this usage is/will be increasing, decreasing, or staying about the same? 10e2

increasing 10e2a

decreasing 10e2b

staying about the same 10e2c

What has been the single most significant impact of your using this capability on your work routine 10f

What has been the single most significant impact of your using this capability on your communications with others 10g

What was the primary mode/ medium you used to accomplish an analogous task before you started using NLS? 10h

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts, or suggestions for change? 10i

NLS AS A WORD PROCESSOR: ON-LINE EDITING, CUTTING, AND PASTING 11

NLS has the capability to allow the user to electronically cut and paste a document on-line, and to perfect the "print" through numerous editing features. 11a

Do you use this capability either dirtcty or indirectly in NLS? 11b

1. Yes ..... move to question 21F 11b1

2. Yes - but also do same thing with other computer system(s).....go to question 21C. 11b2

3. No - but use other computer systems to do the same thing..... go to question 21D. 11b3

4. No - use no computer system to do this.....go to question 21E. 11b4

What other computer systems do you use? (Please list) 11c

How many times in the last month have you used a text-editing system other than NLS? .....times/month 11c1

What factors determine which system you use? 11c2

go on to question #21F 11c2a

What other systems do you use? 11d

How many times in the last month have you used a text-editing system other than NLS? .....times/month 11d1

Why did you choose not to use NLS? 11d2

was not aware of the capability 11d2a

no need 11d2b

too hard to use 11d2c

too poorly documented 11d2d

other: please describe 11d2e

How do you usually accomplish this task? 11e

Why did you choose not to use NLS? 11e1

was not aware of the capability 11e1a

no need 11e1b

too hard to use 11e1c

other: 11e1d

go on to next capability description, question #12. 11e1d1

Would you say your usage of NLS as a text editor / word processor is/will be increasing, decreasing, or staying about the same? 11f

increasing 11f1

decreasing 11f2

staying about the same 11f3

What has been the single most significant impact of your using this capability on your work routine 11g

What was the primary mode/ medium you used to accomplish an analogous task before you started using NLS? 11h

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts, suggestions for change, etc.? 11i

In those situations where SOMEONE ELSE inputs the information, what is the average turnaround for: 11j

a one page document .....hours.....days 11j1

a 10 page document .....hours.....days 11j2

a document of more than ten pages .....hours.....days 11j3

How many revisions do you go through before you get a satisfactory product? 11j4

.....revisions (If you ONLY use this capability indirectly, that is through another user, go on to question # 12). 11j4a

How easy was it to learn how to use this capability? 11k

Very easy...1...2...3...4...5...Very hard 11k1

Any comments on training in this capability or on ease of use? 11K2

Which system do you use for editing, cutting and pasting? 111

DNLS 1111

TNLS.... 1112

NLS AS A PRINTER 12

NLS also has the capability to provide the user with hard copy in a number of ways. Do you ever use NLS for printing? If you ONLY use a terminet terminal or similar terminal with paper roll, check here ..... and go on to next capability, question 13. 12a

yes 12a1

no (go to next capability description, question #24) 12a2

If yes, in which of the following ways: 12b

I press THE "PRINT" BUTTON, and obtain hard copy from a terminal printer located nearby. 12b1

yes (go to question #23B1D) 12b1a

no 12b1b

If no, why? 12b1b1

was not aware of the capability 12b1b1a

was aware but don't know how to use it 12b1b1b

we don't have the necessary hardware 12b1b1c

no need for it 12b1b1d

other 12b1b1e

comments: 12b1b2

(go on to question #23B3) 12b1b2a

How often do you use this capability in an average month? 12b1c

.....times each month 12b1c1

Would you say this usage is/will be increasing, decreasing, or staying about the same? 12b1c2

increasing 12b1c2a

decreasing 12b1c2b

staying about the same 12b1c2c

In what types of situations, or under what conditions do you use this capability? 12b1d

What has been the single most significant impact of your using this capability on your:work routine 12b1e

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts, or suggestions for change? 12b1f

I go through THE OUTPUT PROCESSING COMMAND: 12b2

yes (go to question #23B2D) 12b2a

yes, but use it indirectly (go to question #23B2E) 12b2b

no 12b2c

If no, why? 12b2c1

was not aware of the capability 12b2c1a

was aware but don't know how to use it 12b2c1b

we don't have the necessary hardware 12b2c1c

no need for it 12b2c1d

other 12b2c1e

comments: 12b2c2

(go to question #23B3) 12b2c2a

How easy was it to learn how to use this capability? 12b2d

Very easy...1...2...3...4...5...Very hard 12b2d1

Any comments? 12b2d2

How often do you use this capability in an average month? 12b2e

.....times each month 12b2e1

Would you say this usage is/will be increasing, decreasing, or staying about the same? 12b2e2

increasing 12b2e2a

decreasing 12b2e2b

staying about the same 12b2e2c

In what types of situations, or under what conditions do you use this capability? 12b2f

What has been the single most significant impact of your using this capability on your:work routine 12b2g

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts, suggestions, etc.? 12b2h

I use THE "COM" SYSTEM 12b3

yes (go to question #23B3D) 12b3a

yes, but use it indirectly (go to question #23B3E) 12b3b

no 12b3c

If no, why? 12b3c1

was not aware of the capability 12b3c1a

was aware but don't know how to use it 12b3c1b

no need for it 12b3c1c

other 12b3c1d

comments: 12b3c2

(go to next capability description, question #24) 12b3c2a

How easy was it to learn how to use this capability? 12b3d

Very easy...1...2...3...4...5...Very hard 12b3d1

Any comments? 12b3d2

How many times have you used this capability? 12b3e

Over what time frame? 12b3e1

Would you say this usage is/will be increasing, decreasing, or staying about the same? 12b3f

increasing 12b3f1

decreasing 12b3f2

staying about the same 12b3f3

In what types of situations, or under what conditions do you use this capability? 12b3g

What would you estimate is the average preparation time for a COM document, that is how long does it take to format the paper with the necessary directives, after the content of the paper has been drafted. 12b3h

approximately.....time for approximately .....number of pages. 12b3h1

What is the average turnaround time for your COM documents, that is how long does it take from the time your document is ready for delivery to COM (with directives)and the time a satisfactory version is returned to you? 12b3i

.....days 12b3i1

What has been the single most significant impact of your using this capability on your: work routine? 12b3j

Do you have any additional comments you would like to make regarding the pros and cons of this capability, your usage, its impacts,or suggestions for change? 12b3k

USEROPTIONS 13

NLS also contains a number of useroptions subprograms. Do you make use of any of these? 13a

yes (go to question #25B) 13a1

no 13a2

If no, why? (check all that apply) 13a2a

was not aware of the capability 13a2a1

was aware of it but don't know how to use it 13a2a2

no need for the capability 13a2a3

too hard to use 13a2a4

other 13a2a5

If yes, which subprograms do you use? Please list and describe how you use each and for what purposes / types of tasks: 13b

Were these generally easy or hard to learn to use? 13b1

very easy..1..2..3..4..5..very hard 13b1a

How frequently do you use each of the above? 13b2

Do you have any additional comments you would like to make regarding the pros and cons of the useroptions, your usage, its impacts, or suggestions for change? 13c

FINAL COMMENTS: 14

Are there any final comments you would like to make regarding your usage of NLS, the system itself, the effects of your usage on yourself, others, or your organization, or perhaps comments regardestionnaire? 14a

BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BB
BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BB
BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BBN-TENEXB BB

FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER
FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER
FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER

(JAKE)34324 (JAKE)34324 (JAKE)34324 (JAKE)34324 (JAKE)34324 (JAKE)3432
(JAKE)34324 (JAKE)34324 (JAKE)34324 (JAKE)34324 (JAKE)34324 (JAKE)3432
(JAKE)34324 (JAKE)34324 (JAKE)34324 (JAKE)34324 (JAKE)34324 (JAKE)3432

TUESDAY, JANUARY 27, 1976 19:14:08-EST TUESDAY, JANUARY 27, 1976 19:14:08-EST
TUESDAY, JANUARY 27, 1976 19:14:08-EST TUESDAY, JANUARY 27, 1976 19:14:08-EST
TUESDAY, JANUARY 27, 1976 19:14:08-EST TUESDAY, JANUARY 27, 1976 19:14:08-EST

< AJOURNAL, 34324.NLS;1, >, 26-JAN-76 11:49 XXX ;;;; .HJOURNAL="DLS 26-JAN-76 06:43 34324"; Title: .H1="Mail Systems, Journal & Internal Company Mail"; Author(s): Duane L. Stone/DLS; Distribution: /RA3Y([ACTION]) DPCS([INFO-ONLY]) RAM4([INFO-ONLY]) JPC([INFO-ONLY]) JLM([INFO-ONLY]) RJC([INFO-ONLY]) ; Sub-Collections: RADC DPCS; Clerk: DLS; .IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PES;

.PEL; .PN=PN-1; .GCR;Contains comments on (34281,)
Ray, I have been following your work with the MSGROUP and the Journal. I have been meaning to interject a few thoughts earlier, but as everyone, have been busy with everday concerns. In reading your latest version of the ICC paper, I came across a few things that I'd like to comment on, especially the relationship of Journal to "internal mail". References are to (34281,) statement numbers.

(2c) ...in most moderate size offices...This tends to imply that offices are the only place that can accommodate the terminal. Anywhere there is a conventional phone handset, including the nearest phone booth is a candidate for plugging in a terminal.

(4a) Although you qualify your statements by cautioning the reader that "this analysis is not rigorous", I would be interested in a little more rigor.

(4b) ...$1.50 for a 100-150 word message...I'd be interested in see how you arrived at this figure. It might not be appropriate to put it in the paper itself, but just for my info. Does this $ figure apply in a linear manner to documents of 100-150 pages? Does the number include transmission of copies to other hosts participating in the Journal and multiple addressees? Are background costs like maintenance of the ident system included, or is this trivial?

(4b) TWX is terminal-to-terminal message service isn't it? This does not account for the additional cost/time/effort in getting the message from the originator to the message delivery system and then from the output machine to the addressee. A typical delay in a proposal that Norton might send me is a week, a day in the SRI internal mails, 3-4 cross-country and a couple of days in the RADC internal mails. Only the cross country delay would be saved by something like TWX.

(4b) ...million terminals...but potentially 50M? considering TV sets (if they can be made into terminals for a few hundred bucks).

(4i) ...first six months...are you also waiting for the Journal indices to be updated? Other important parameters that should be mentioned, and that one can get fairly easily by sampling, are the average length of the "messages" and the average number of addressees. This data would give a little more complete picture of the volume of traffic to be expected. The number of private messages should also be analyzed, since this would give more complete numbers and an indication about the default setting of the public/private switch.

Is it difficult to trap the number of times SNDMSG is called? If so could Jeff or someone do this for a month and collect some data to support the 3X SNDMSG factor.

(4g) The COTCO study is an attempt to describe one type of "internal company" mail.

One of the biggest problems with NLS, is that it does not support internal mail. I tend to do all the work I can on NLS; memos, statements of work, sole sources, trip reports, etc. Most of

these must be printed out at the lowest level (me) and entered into the "office" system. There they are edited, hacked up, returned, sent off to sister offices for coordination, commented, initialed, signed, etc, until they either emerge a finished product or die a slow death. It does not seem to me to be such a big thing to accomodate all this activity using NLS...especially on a system called OFFICE-1. A few basic capabilities are needed, sketched out below.

The ident system (or something built on top of it) needs to be able to model a tree structured organization, which encompasses most of the military. From my view, most correspondence flows up the chain-of-command, from office-to-office, not from person-to-person. Routing is controlled by a list of Office symbols terminated by IN TURN, meaning that the top person on the list looks at it first, the bottom person is the intended recipient. The person next in line reviews, comments, edits and then either initials/signs and forwards, returns to sender or sends to an office not on the original routing list for coordination. In any event, the paper is routed to offices, the people in these offices may vary from day to day depending on travel, leave etc. When they do, another person is authorized to sign for the normal office chief. This implies that the Office ident system needs to be dynamic in nature, where Office idents can be readily changed and mapped into NLS idents.

Another capability that is needed is to be able to comment on a piece of a memo (like Insert <>Comment) and have the comments visible or not depending on perhaps a viewspec. From my meger understanding of the new file structure, an attribute of a statement might be that it is commented. These comments should contain the same signature as regular NLS statements so that persons up-the-line can see easiy who said what when...perhaps with last name instead of ident.

Persons up-the-line should also be able to perform any edits that they might wish, while still retaining the original author's version.

When the memo is finally initialed/signed by everyone, signature blocks should be automatically inserted based on the type of memo and the memo should be automaticaly submitted to the journal with copies back to anyone who has initialed or edited the memo.

There should be a background process that collects statistics on the number of memos, source and length of time spent in each stage. This would give us some real data on the volume of internal mail and an idea where the bottle necks exists. One should also be able to query the office system, to determine the progress (or lack thereof) of a piece of mail.

I think we know enough about the military internal mail process to specify a beginning system. A system like that outlined above, though not as broad in scope as that proposed by COTOCO, would be extremely useful. I feel that it could be implemented with a manyear of effort. The problem is getting funds to do it. RADC has all of its $ tied up in the NSW and Utility. If SRI is interested in doing this, perhaps there are internal funds that can be made available? There are other DOD orgaanizations that would benefit from this I'm sure. RADC would be more than happy

to act as a test case in the IS division, if someone can come up with the $/manpower.

Trivia

(3d) "marked" _ market

(3e) C3...The order around here is Command, Control and Communication. It is quite often spoken as "C cubed", written C with a superscript of 3 or typed CCC if you can't roll back the platen a half line.

(4k) "usabe" _ usage

(5c5) "dwarved" _ dwarfed

(6a) "then" _ the

"then" - the

OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFF
OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFF
OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFFICE-1 OFF

GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF
GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF
GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF GEOFF

SPONSORS-MEETING SPONSORS-MEETING SPONSORS-MEETING SPONSORS-MEETING SPON
SPONSORS-MEETING SPONSORS-MEETING SPONSORS-MEETING SPONSORS-MEETING SPON
SPONSORS-MEETING SPONSORS-MEETING SPONSORS-MEETING SPONSORS-MEETING SPON

TUESDAY, JANUARY 27, 1976 16:27:22-PST TUESDAY, JANUARY 27, 1976 16:27:22-PST
TUESDAY, JANUARY 27, 1976 16:27:22-PST TUESDAY, JANUARY 27, 1976 16:27:22-PST
TUESDAY, JANUARY 27, 1976 16:27:22-PST TUESDAY, JANUARY 27, 1976 16:27:22-PST

INTRODUCTION 1

In a companion paper [1], the author proposes a simple protocol and software framework that would facilitate the construction of distributed systems within a resource-sharing computer network by enabling distant processes to communicate with one another at the procedure call level. Although of great utility even in its present form, this rudimentary "distributed programming system (DPS)" supports only the most fundamental aspects of remote procedure calling. In particular, it permits the caller to identify the remote procedure to be called, supply the necessary arguments, determine the outcome of the procedure, and recover its results. The present paper extends this simple procedure call model and standardizes other common forms of process interaction to provide a more powerful and comprehensive distributed programming system. The particular extensions proposed in this paper serve hopefully to reveal the DPS concept's potential, and are offered not as dogma but rather as stimulus for further research. 1a

The first section of this paper summarizes the basic distributed programming system derived in [1]. The second section describes the general strategy to be followed in extending it. The third and longest section identifies and explores some of the aspects of process interaction that are sufficiently common to warrant standardization, and suggests methods for incorporating them in the DPS model. 1b

REVIEWING THE BASIC SYSTEM 2

The distributed programming system derived in [1] assumes the existence of and is built upon a network-wide "inter-process communication (IPC)" facility. As depicted in Figure 1, DPS consists of a high-level model of computer processes and a simple, application-independent "procedure call protocol (PCP)" that implements the model by regulating the dialog between two processes interconnected by means of an IPC communication "channel". DPS is implemented by an installation-provided "run-time environment (RTE)", which is link loaded with (or otherwise made available to) each applications program. 2a

The Model 2b

The procedure call model (hereafter termed the Model) views a process as a collection of remotely callable subroutines or "procedures". Each procedure is invoked by name, can be supplied a list of arguments, and returns to its caller both a boolean outcome, indicating whether it succeeded or failed, and a list of results. The Model permits the process at either end of the IPC channel to invoke procedures in its neighbor, and further permits a process to accept two or more procedure calls for concurrent execution. 2b1

The arguments and results of procedures are modeled from a small set of primitive "data types", listed below: 2b2

LIST: A list is an ordered sequence of N data objects called "elements" (here and throughout these descriptions, N is confined to the range [0, 2**15-1]). A LIST may contain other LISTs as elements, and can therefore be employed to construct arbitrarily complex, composite arguments or results.

CHARSTR: A character string is an ordered sequence of N ASCII characters, and conveniently models a variety of textual entities, from short user names to whole paragraphs of text.

BITSTR: A bit string is an ordered sequence of N bits and, therefore, provides a means for representing arbitrary binary data (for example, the contents of a word of memory).

INTEGER: An integer is a fixed-point number in the range [-2**31, 2**31-1], and conveniently models various kinds of numerical data, including time intervals, distances, and so on.

INDEX: An index is an integer in the range [1, 2**15-1]. As its name and value range suggest, an INDEX can be used to address a particular bit or character within a string, or element within a list. Furthermore, many of the protocol extensions to be proposed in this paper will employ INDEXes as handles for objects within the DPS environment (for example, processes and channels).

BOOLEAN: A boolean represents a single bit of information, and has either the value true or false.

EMPTY: An empty is a valueless place holder within a LIST or parameter list.

The Protocol 2c

The procedure call protocol (hereafter termed the Protocol), which implements the Model, defines a "transmission format" (like those suggested in Appendix A) for each of the seven data types listed above, and requires that parameters be encoded in that format whenever they are transported between processes. 2c1

The Protocol also specifies the inter-process messages by which remote procedures are invoked. These messages can be described symbolically as follows: 2c2

```
message-type=CALL   [tid] procedure-name arguments
message-type=RETURN  tid  outcome        results
```

The first message invokes the procedure whose NAME is specified using the ARGUMENTS provided. The second is returned in eventual response to the first and reports the OUTCOME and RESULTS of the completed procedure. Whenever OUTCOME indicates that a procedure has failed, the procedure's RESULTS are required to be an error number and diagnostic message, the former to help the invoking program determine what to do next, the latter for possible presentation to the user. The presence of an optional "transaction identifier (TID)" in the CALL message constitutes a request by the caller for an acknowledging RETURN message echoing the identifier. 2c3

Although data types and their transmission formats serve primarily as vehicles for representing the arguments and results of remote procedures, they can just as readily and effectively be employed to represent the messages by which those parameters are transmitted. The Protocol, therefore, represents each of the two messages described above as a PCP data object, namely, a LIST whose first element is an INDEX message type. The following concise statement of the Protocol results: 2c4

```
LIST (CALL,   tid,    procedure, arguments)
      INDEX=1 [INDEX] CHARSTR    LIST
LIST (RETURN, tid,    outcome,   results)
      INDEX=2 INDEX   BOOLEAN    LIST
```

Here and in subsequent protocol descriptions, elements enclosed in square brackets are optional (that is, may be EMPTY). The RESULTS of an unsuccessful procedure would be represented as follows: 2c5

```
LIST (error, diagnostic)
      INDEX  CHARSTR
```

The Run-Time Environment 2d

The run-time environment (hereafter termed the environment) interfaces the applications program to a remote process via an IPC channel. In doing so, it provides the applications program with a collection of "primitives", implemented either as subroutines or system calls, that the applications program can employ to manipulate the remote process to which the channel connects it. The environment implements these primitives by sending and receiving various protocol messages via the channel. 2d1

In its present rudimentary form, the Protocol enables the environment to make a single, remote procedure calling primitive like the following available to the applications program: 2d2

CALLPROCEDURE (procedure, arguments -> outcome, results)
CHARSTR LIST BOOLEAN LIST

This primitive invokes the indicated remote PROCEDURE using the ARGUMENTS provided and returns its OUTCOME and RESULTS. While this primitive blocks the invoking applications program until the remote procedure returns, a variant that simply initiates the call and allows the applications program to collect the outcome and results in a second operation can also be provided. 2d3

Since the interface between the environment and the applications program is machine- and possibly even language-dependent, environment-provided primitives can only be described in this paper symbolically. Although PCP data types provide a convenient vehicle for describing their arguments and results and are therefore used for that purpose above and throughout the paper, such parameters will normally be transmitted between the environment and the applications program in some internal format. 2d4

BOOTSTRAPPING THE NEW PROTOCOL FUNCTIONS 3

Since the Protocol already provides a mechanism for invoking arbitrary remote procedures, the Model extensions to be proposed in this paper will be implemented whenever possible as procedures, rather than as additional messages. Unlike applications procedures, these special "system procedures" will be called and implemented by run-time environments, rather than by the applications programs they serve. Although inaccessible to the remote applications program via the normal environment-provided remote procedure calling primitive, system procedures will enable the environment to implement and offer new primitives to its applications program. 3a

The calling sequences of many of these new primitives will closely correspond to those of the remote system procedures by which they are implemented. Other primitives will be more complex and require for their implementation calls to several system procedures, possibly in different processes. Besides describing the Protocol additions required by the various Model extensions proposed, the author will, throughout this paper, suggest calling sequences for the new primitives that become available to the applications program. 3b

SOME POSSIBLE EXTENSIONS TO THE MODEL 4

1. Creating Remote Processes 4a

Before a program in one machine can use resources in another, it must either create a new process in the remote machine, or gain access to an existing one. In either case, the local process must establish an IPC channel to a resident dispatching process within the remote system, specify the program to be started or contacted, and identify itself so that its access to the program can be established and billing carried out. After these preliminary steps have been accomplished, the requested process assumes responsibility for the IPC channel and substantive communication begins. 4a1

The manner in which the environment carries out the above scenario is largely dictated by the IPC facility upon which the distributed system is based. If the IPC facility itself provides a single primitive that accomplishes the entire task, then the environment need only invoke that primitive. If, on the other hand, it only provides a mechanism by which the environment can establish a channel to the remote dispatcher, as is the case within the ARPA Computer Network (the ARPANET), then the Protocol itself must contain provisions for naming the program to be run and presenting the required credentials. 4a2

Adding to the Protocol the following system procedure enables the local environment to provide the remote dispatcher with the necessary information in this latter case: 4a3

```
INIPROCESS (program, credentials)
            CHARSTR  LIST (user,   password, account)
                          CHARSTR CHARSTR   CHARSTR
```

Its arguments include the name of the applications PROGRAM to be run; and the USER name, PASSWORD, and ACCOUNT of the local user to whom its use is to be billed. 4a4

This new procedure effectively adds to the Model the notion of "process creation", and enables the environment to offer the following primitives to its applications program: 4a5

```
CRTPROCESS (computer, program, credentials -> ph)
            CHARSTR   CHARSTR  (as above)     INDEX
DELPROCESS (ph)
            INDEX
```

The first primitive creates a new process or establishes contact with an existing one by first creating a channel to the dispatcher within the indicated COMPUTER and then invoking the remote system procedure INIPROCESS with the specified PROGRAM name and CREDENTIALS as arguments. The primitive returns a "process handle PH" by which the applications program can refer to the newly created process in subsequent dialog with the local environment. The process handle may be a token returned to the environment by the IPC facility, an index into a table within the environment, or anything else the environment's implementer may find convenient. 4a6

The second primitive "deletes" the previously created process whose handle PH is specified by simply deleting the IPC channel to the remote process and reclaiming any internal table space that may have been allocated to the process. 4a7

2. Introducing Processes to One Anotner 4b

The simplest distributed systems begin with a single process that creates, via the CRTPROCESS primitive described above, one or more "inferior" processes whose resources it requires. Some or all of these inferiors may in turn require other remote resources and so create inferiors of their own. This creative activity can proceed, in principle, to arbitrary depth. The distributed system is thus a tree structure whose nodes are processes and whose branches are IPC channels. 4b1

Although a distributed system can include an arbitrarily large number of processes, each process is cognizant of only the process that created it and those it itself creates, that is, its parent and sons. The radius within which a process can access the resources of the tree is thus artificially small. This limited sharing range, which prevents the convenient implementation of many distributed systems, can be overcome by extending the Model to permit an arbitrarily complex network of communication paths to be superimposed upon the process tree. 4b2

One of the many ways by which the Protocol can provide for such communication paths is to permit one process to "introduce" and thereby make known to one another any two processes it itself knows (for example, two of its sons, or its parent and son). Once introduced, the two processes would be able to invoke one another's procedures with the same freedom the introducing process enjoys. They could also introduce one another to other processes, and so create even longer communication paths. 4b3

2.1 Introductions Within a Homogeneous Environment 4b4

Provided one remains within a "homogeneous environment" (that is, the domain of a single IPC facility), the introduction of two processes requires little more than the formation of an IPC channel between them. Adding to the Protocol the following system procedures, which manipulate IPC "ports", enables the run-time environment of the process performing the introduction to negotiate such a channel:

```
ALOPORT (-> ph,   computer, port)
            INDEX CHARSTR   any
CNNPORT (ph,   computer, port)
         INDEX CHARSTR   any
DCNPORT (ph)
         INDEX
```

The detailed calling sequences for these procedures are dictated by the IPC facility that underlies the distributed system. Those above are therefore only representative of what may be required within any particular network, but are only slightly less complicated than those required, for example, within the ARPANET.

To create the channel, the introducing process' run-time environment allocates a PORT in each target process via ALOPORT, and then instructs each process via CNNPORT to connect its port to the other's via the IPC facility. The process handle PH returned by ALOPORT serves as a handle both initially for the allocated port, and then later for the process to which the attached channel provides access. To "separate" the two processes, the introducing process' environment need only invoke the DCNPORT procedure in each process, thereby dissolving the channel, releasing the associated ports, and deallocating the process handles.

Armed with these three new system procedures, the environment can provide the following new primitives to its applications program:

```
ITDPROCESS (ph1,  ph2 -> ph12, ph21, ih)
            INDEX INDEX  INDEX INDEX INDEX
SEPPROCESS (ih)
            INDEX
```

The first primitive introduces the two processes whose handles PH1 and PH2 are specified. Each handle may designate either a son, in which case the handle is one returned by CRTPROCESS; the parent process, for which a special handle (for example, 1) must always be defined; or a previously introduced process, in which case the handle is one obtained in a previous invocation of ITDPROCESS.

ITDPROCESS returns handles PH12 and PH21 by which the two processes will know one another, as well as an "introduction handle IH" that the applications program can later employ to separate the two processes via SEPPROCESS. The applications program initiating the introduction assumes responsibility for communicating to each introduced applications program its handle for the other.

2.2 Introductions Within a Heterogeneous Environment 4b5

While their interconnection via an IPC channel is sufficient to introduce two processes to one another, in a heterogeneous environment the creation of such a channel is impossible. Suppose, as depicted in Figure 2, that processes P1 and P2 (in computers C1 and C2, respectively) are interconnected within a distributed system by means of a network IPC facility. Assume further that P2 attaches to the system another process P3 in a minicomputer M that although attached to C2 is not formally a part of the network. With this configuration, it is impossible for P2 to introduce processes P1 and P3 to one another by simply establishing an IPC channel between them, since they are not within the domain of a single IPC facility.

One way of overcoming this problem is to extend the Model to embrace the notion of a composite or "logical channel" composed of two or more physical (that is, IPC) channels. A message transmitted by process P1 via the logical channel to Pn (n=3 in the example above) would be relayed over successive physical channels by the environments of intermediate processes P2 through Pn-1. Although more expensive than physical channels, since each message must traverse at least two physical channels and be handled by all the environments along the way, logical channels would nevertheless enable processes that could not otherwise do so to access one another's resources. Since the relaying of messages is a responsibility of the environment, the applications program need never be aware of it.

As depicted in Figure 3, a logical channel would consist of table entries maintained by the environment of each process P1 through Pn, plus the committment of each environment to forward messages that arrive with a "routing code" addressing the local table entry. Each table entry would contain process handles for the two adjacent processes, as well as the routing code recognized by each. To communicate a message to its distant neighbor, the source process (say P1) would transmit it via its IPC channel to P2, with a routing code addressing the appropriate table entry within P2. Upon receipt of the message, P2 would locate its table entry via the routing code, update the message with the routing code recognized by P3, and forward the message to P3. Eventually the message would reach its final destination, Pn.

Adding to the Protocol the following system procedures enables the environment to construct a logical channel like that described above:

```
CRTROUTE (mycode, oldcode -> code, ph)
          INDEX   [INDEX]    INDEX INDEX
DELROUTE (yourcode)
          INDEX
```

The simplest logical channel (n=3) is created by P2, which invokes CRTROUTE in both P1 and P3, specifying in each case the routing code MYCODE it has assigned to its segment of the logical channel, and receiving in return the routing CODEs and process handles PHs assigned by the two processes. OLDCODE is not required in this simple case and is therefore EMPTY.

More complicated logical channels (n>3) are required when one or both of the processes to be introduced is already linked, by a logical channel, to the process performing the introduction. In such cases, a portion of the new channel to be constructed must replicate the existing channel, and hence the routing code OLDCODE for the table entry that represents that channel within the target process is specified as an additional argument of the system procedure. The target process must call CRTROUTE recursively in the adjacent process to replicate the rest of the model channel.

The process Pi that creates a logical channel assumes responsibility for insuring that it is eventually dismantled. It deletes the logical channel by invoking DELROUTE in Pi-1 and Pi+1, each of which propagates the call toward its end of the channel.

3. Controlling Access to Local Resources 4c

The process introduction primitive proposed above effectively permits access to a process to be transmitted from one process to another. Any process P2 that already possesses a handle to a process P1 can obtain a handle for use by a third process P3. Once P1 and P3 have been introduced, P3 can freely call procedures in P1 (and vice versa). 4c1

Although a process can, by aborting the ALOPORT system procedure, prevent its introduction to another process and so restrict the set of processes that gain access to it, finer access controls may sometimes be required. A process may, for example, house two separate resources, one of which is to be made available only to its parent (for example), and the other to any process to which the parent introduces it. Before such a strategy can be conveniently implemented, the Model must be extended to permit access controls to be independently applied to individual resources within a single process. 4c2

Although a single procedure can be considered a resource, it is usually more practical and convenient to conceive of larger, composite resources consisting of a number of related procedures. A simple data base management module containing procedures for creating, deleting, assigning values to, reading, and searching for data objects exemplifies such composite resources. Although each procedure is useless in isolation, the whole family of procedures provides a meaningful service. Such "packages" of logically related procedures might thus be the most reasonable object of the finer access controls to be defined. 4c3

Access controls can be applied to packages by requiring that a process first "open" and obtain a handle for a remote package before it may call any of the procedures it contains. When the process attempts to open the package, its right to do so can be verified and the attempt aborted if necessary. Challenging the open attempt would, of course, be less expensive than challenging every procedure call. The opening of a package would also provide a convenient time for package-dependent state information to be initialized. 4c4

Adding to the Protocol the following pair of system procedures enables the environment to open and close packages within another process. For efficiency, these procedures manipulate an arbitrary number of packages in a single transaction: 4c5

```
OPNPACKAGE (packages ->    pkhs)
           LISTofCHARSTRs LISTofINDEXs
CLSPACKAGE (pkhs)
           (as above)
```

The first procedure opens and returns "package handles PKHS" for the specified PACKAGES; the second closes one or more packages and releases the handles PKHS previously obtained for them. 4c6

Besides incorporating these two new system procedures, the Protocol must further require that a package handle accompany the procedure name in every CALL message (an EMPTY handle perhaps designating a system procedure). Note that this requirement has the side effect of making the package the domain within which procedure names must be unique. 4c7

The system procedures described above enable the environment to make available to its applications program, primitives that have calling sequences similar to those of the corresponding system procedures but which accept the process handle of the target process as an additional argument. Their implementation requires only that the environment identify the remote process from its internal tables and invoke OPNPACKAGE or CLSPACKAGE in that process. 4c8

4. Standardizing Access to Global Variables 4d

Conventional systems often maintain global "variables" that can be accessed by modules throughout the system. Such variables are typically manipulated using primitives of the form: 4d1

(1) Return the current value of V.
(2) Replace the current contents of V with a new value.

These primitives are either provided as language constructs or implemented by specialized procedures. The former approach encourages uniform treatment of all variables within the system. 4d2

Those distributed systems that maintain remotely-accessible variables must also select a strategy for implementing the required access primitives. While such primitives can, of course, be

implemented as specialized applications procedures, adding to the Protocol the following new system procedures insures a uniform run-time access mechanism: 4d3

```
RDVARIABLE (pkh,  variable -> value)
            INDEX CHARSTR     any
WRVARIABLE (pkh,  variable, value)
            INDEX CHARSTR   any
```

These procedures effectively define variables as named data objects modeled from PCP data types, and suggest that they be clustered in packages with related procedures. The system procedures return and specify, respectively, the VALUE of the VARIABLE whose name and package handle PKH are specified. 4d4

These new procedures enable the environment to make available to its applications program, primitives that have calling sequences similar to those of the corresponding system procedures but which accept the process handle of the target process as an additional argument. These primitives provide a basis upon which a suitably modified compiler can reestablish the compile-time uniformity that characterizes the manipulation of variables in conventional programming environments. Their implementation requires only that the local environment identify the remote process from its internal tables and invoke RDVARIABLE or WRVARIABLE in that process. 4d5

Most variables will restrict the range of data types and values that may be assigned to them; some may even be read-only. But because they are modeled using PCP data types, their values can, in principle, be arbitrarily complex (for example, a LIST of LISTs) and the programmer may sometimes wish to manipulate only a single element of the variable (or, if that element is itself a LIST, just one of its elements; and so on, to arbitrary depth). 4d6

Adding the following argument to their calling sequences extends the system procedures proposed above to optionally manipulate a single element of a variable's composite value: 4d7

```
substructure
[LISTofINDEXs]
```

At successive levels of the value's tree structure, the INDEX of the desired element is identified; the resulting list of indices identifies the SUBSTRUCTURE whose value is to be returned or replaced. 4d8

5. Routing Parameters Between Procedures 4e

In conventional programming systems, the results of procedures are used in a variety of ways, depending upon the context of the calls made upon them. A result may, for example: 4e1

(1) Provide the basis for a branch decision within the calling program.
(2) Become an argument to a subsequent procedure call.
(3) Be ignored and thus effectively discarded.

At run-time, the knowledge of a result's intended use usually lies solely within the calling program, which examines the result, passes it to a second procedure, or ignores it as it chooses. 4e2

In a distributed system, the transportation of results from callee to caller, carried out by means of one or more inter-process messages, can be an expensive operation, especially when the results are large. Data movement can be reduced in Cases 2 and 3 above by extending the Model to permit the intended disposition of each procedure result to be made known in advance to the callee's environment. In Case 2, provided both callees reside within the same process, the result can be held at its source and later locally supplied to the next procedure. In Case 3, the result can be discarded at its source (perhaps not even computed), rather than sent and discarded at its destination. 4e3

5.1 Specifying Parameters Indirectly 4e4

Variables offer potential for eliminating the inefficiencies involved in Case 2 above by providing a place within the callees' process where results generated by one procedure can be held until required by another. The Protocol can be extended to permit variables to be used in this way by allowing the caller of any procedure to include optional "argument- and result-list masks" like the following as additional parameters of the CALL message:

```
parameter list mask
[LIST (variable, ...)]
      [CHARSTR]
```

A parameter list mask would permit each parameter to be transmitted either directly, via the parameter list, or indirectly via a VARIABLE within the callee's process. Thus each

element of the mask specifies how the callee's environment is to obtain or dispose of the corresponding parameter. To supply the result of one procedure as an argument to another, the caller need only then appropriately set corresponding elements of the result and argument list masks in the first and second calls, respectively. The result list mask should be ignored if the procedure fails, and the error number and diagnostic message returned directly to the caller.

5.2 Providing Scratch Variables For Parameter Routing 4e5

Although each applications program could provide variables for use as described above, a more economical approach is to extend the Model to permit special "scratch variables", maintained by the environment without assistance from its applications program, to be created and deleted as necessary at run-time. Adding to the Protocol the following pair of system procedures enables the local environment to create and delete such variables in a remote process:

```
CRTVARIABLE (variable, value)
             CHARSTR   any
DELVARIABLE (variable)
             CHARSTR
```

These procedures create and delete the specified VARIABLE, respectively. CRTVARIABLE also assigns an initial VALUE to the newly-created variable.

These new procedures enable the environment to make available to its applications program, primitives that have calling sequences similar to those of the corresponding system procedures but which accept the process handle of the target process as an additional argument. Their implementation requires only that the environment identify the remote process from its internal tables and invoke CRTVARIABLE or DELVARIABLE in that process.

5.3 Discarding Results 4e6

The inefficiencies that result in Case 3 above are conveniently eliminated by allowing the caller to identify via the result list mask (for example, via a zero-length CHARSTR) that a result will be ignored and therefore need not be returned to the caller.

6. Supporting a Richer Spectrum of Control Transfers 4f

As currently defined by the Model, a procedure call is a simple, two-stage dialog in which the caller first describes the operation it wishes performed and the callee, after performing the operation, reports its outcome. Although this simple dialog form is sufficient to conveniently implement a large class of distributed systems, more complex forms are sometimes required. The Model can be extended to admit a variety of more powerful dialog forms, of which the four described below are examples. 4f1

6.1 Transferring Control Between Caller and Callee 4f2

Many conventional programming systems permit caller and callee to exchange control any number of times before the callee returns. Such "coroutine linkages" provide a means, for example, by which the callee can obtain help with a problem that it has encountered or deliver the results of one sub-operation and obtain the arguments for the next.

Adding to the Protocol the following system procedure, whose invocation relinquishes control of another, previously initiated procedure, enables the environment to effect a coroutine linkage between caller and callee:

```
TAKEPROCEDURE (tid,  yourtid, parameters)
                INDEX BOOLEAN  LIST
```

Its arguments include the identifier TID of the affected transaction, an indication YOURTID of from whose name space the identifier was assigned (that is, whether the process relinquishing control is the caller or callee), and PARAMETERS provided by the procedure surrendering control. By exploiting an existing provision of the Protocol (that is, by declining acknowledgment of its calls to TAKEPROCEDURE,) the invoking environment can effect the control transfer with a single inter-process message.

The addition of this new procedure to the Protocol enables the environment to provide the following new primitive to its applications program:

```
LINKPROCEDURE (tid,  arguments -> outcome,  results)
               INDEX LIST          [BOOLEAN] LIST
```

This primitive assumes that the CALLPROCEDURE primitive is also modified to return the pertinent transaction identifier should the callee initiate a coroutine linkage rather than return. Invocation of LINKPROCEDURE then continues the dialog by supplying ARGUMENTS and returning control to the remote procedure, and then awaiting the next transfer of control and the RESULTS that accompany it. If the remote procedure then returns, rather than initiating another coroutine linkage, the primitive reports its OUTCOME and invalidates the transaction identifier.

While this primitive blocks the applications program until the remote procedure relinquishes control, a variant that simply initiates the coroutine linkage and allows the applications program to collect the outcome and results in a second operation can also be provided.

6.2 Signalling the Caller/Callee 413

A monolog is often more appropriate than the dialog initiated by a coroutine linkage. The caller or callee might wish, for example, to report an event it has detected or send large parameters piecemeal to minimize buffering requirements. Since no return parameters are required in such cases, the initiating procedure need only "signal" its partner, while retaining control of the call.

Adding to the Protocol the following system procedure extends the Model to support signals and enables the environment to transmit parameters to or from another, previously initiated procedure without relinquishing control of the call:

```
SGNLPROCEDURE (tid,  yourtid, parameters)
                INDEX BOOLEAN  LIST
```

Like the TAKEPROCEDURE procedure already described, its arguments include the identifier TID of the affected transaction, an indication YOURTID of from whose name space the identifier was assigned, and the PARAMETERS themselves.

This new procedure enables the environment to make available to its applications program a primitive that has a calling sequence similar to that of the system procedure but which does not require YOURTID as an argument. Its implementation requires only that the environment identify the remote process via its internal tables and invoke SGNLPROCEDURE in that process.

By requesting the acknowledgment of each call to SGNLPROCEDURE and, if necessary, delaying subsequents calls affecting the same transaction until the acknowledgment arrives, the invoking environment effects a crude form of flow control and so prevents the remote process' buffers from being overrun.

6.3 Soliciting Help from Superiors 4f4

As in conventional programming systems, remotely callable procedures within a distributed system will sometimes call upon others to carry out portions of their task. Each procedure along the "thread of control" resulting from such nested calls is, in a sense, responsible to not only its immediate caller but also to all those procedures that lie above it along the control thread. To properly discharge its responsibilities, a procedure must sometimes communicate with these "superiors".

Occasionally a procedure reaches a point in its execution beyond which it cannot proceed without external assistance. It might, for example, require additional resources or further direction from the human user upon whose behalf it is executing. Before reaching this impasse, the procedure may have invested considerable real and/or processing time that will be lost if it aborts.

Adding to the Protocol the following system procedure minimizes such inefficiencies by enabling the environment to solicit help from a callee's superiors:

```
HELPPROCEDURE (tid,   number,  information -> solution)
                INDEX INDEX    any             any
```

Its arguments include the identifier TID of the affected transaction (the direction of the control transfer being implicit in this case), a NUMBER identifying the problem encountered, and arbitrary supplementary INFORMATION.

The primitive that this new procedure enables the environment to provide its applications program has an identical calling sequence. Its implementation requires only that the environment identify the remote process from its internal tables and invoke HELPPROCEDURE in that process.

The search for help begins with invocation of HELPPROCEDURE in the caller's environment. If the caller understands the problem

(that is, recognizes its number) and is able to solve it, HELPPROCEDURE will simply return whatever SOLUTION information the caller provides. Otherwise, HELPPROCEDURE must give the next superior an opportunity to respond by calling itself recursively in that process. The search terminates as soon as a superior responds positively or when the end of the control thread is reached. In the latter case, each of the nested HELPPROCEDURE procedures returns unsuccessfully to indicate to its caller that the search failed.

6.4 Reporting an Event to Superiors 4f5

A procedure sometimes witnesses or causes an event of which its superiors should be made aware (for example, the start or completion of some major step in the procedure's execution). Adding to the Protocol the following system procedure enables the environment to notify a callee's superiors of an arbitrary event:

```
NOTEPROCEDURE (tid,  number, information)
                INDEX INDEX   any
```

Like HELPPROCEDURE, its arguments include the identifier TID of the transaction it affects, a NUMBER identifying the event being reported, and arbitrary supplementary INFORMATION.

The primitive that this new procedure enables the environment to provide its applications program has an identical calling sequence. Its implementation requires only that the environment identify the remote process from its internal tables and invoke NOTEPROCEDURE in that process.

By requesting acknowledgment of each call to NOTEPROCEDURE and, if necessary, delaying subsequents calls that affect that transaction until the acknowledgment arrives, the invoking environment effects a crude form of flow control and so prevents the remote process' buffers from being overrun.

Notification of the procedure's superiors begins with invocation of NOTEPROCEDURE in the caller's process and works its way recursively up the thread of control until the top is reached.

7. Aborting Executing Procedures 4g

Conventional systems that accept commands from the user sometimes permit him to cancel an executing command issued inadvertently or with erroneous parameters, or one for whose completion he cannot wait. This ability is particularly important when the command (for example, one that compiles a source file) has a significant execution time. In a distributed system, the execution of such a command may involve the invocation of one or more remote procedures. Its cancellation, therefore, requires the abortion of any outstanding remote procedure calls. 4g1

Adding to the Protocol the following system procedure provides the basis for a command cancellation facility by enabling the environment to abort another, previously invoked procedure: 4g2

```
ABRTPROCEDURE (tid)
              INDEX
```

Its sole argument is the identifier TID of the transaction it affects. 4g3

The primitive that this new procedure enables the environment to make available to the applications program has an identical calling sequence. Its implementation requires only that the local environment identify the remote process from its internal tables and invoke ABRTPROCEDURE in that process. 4g4

CONCLUSION 5

The expanded Protocol and Model that result from the extensions proposed in the present paper are summarized in Appendixes B and C, respectively. Needless to say, many additional forms and aspects of process interaction, of which Appendix D suggests a few, remain to be explored. Nevertheless, the primitives already made available by the run-time environment provide the applications programmer with a powerful and coherent set of tools for constructing distributed systems. 5a

ACKNOWLEDGMENTS 6

Many individuals within both SRI's Augmentation Research Center (ARC) and the larger ARPANET community have contributed their time and ideas to the development of the Protocol and Model described in this and its companion paper. The contributions of the following individuals are expressly acknowledged: Dick Watson, Jon Postel, Charles Irby, Ken Victor, Dave Maynard, and Larry Garlick of ARC; and Bob Thomas and Rick Schantz of Bolt, Beranek and Newman, Inc. 6a

ARC has been working toward a high-level framework for network-based distributed systems for a number of years now [2]. The particular Protocol and Model result from research begun by ARC in July of 1974. This research included developing the Model; designing and documenting the Protocol required to support it [3]; and designing, documenting, and implementing a prototype run-time environment for a particular machine [4, 5], specifically a PDP-10 running the Tenex operating system developed by Bolt, Beranek and Newman, Inc [6]. Three design iterations were carried out during a 12-month period and the resulting specification implemented for Tenex. The Tenex RTE provides a superset of the capabilities proposed in this paper. 6b

The work reported here was supported by the Advanced Research Projects Agency of the Department of Defense, and by the Rome Air Development Center of the Air Force. 6c

APPENDIX A: TRANSMISSION FORMATS FOR PCP DATA OBJECTS 7

Data objects must be encoded in a standard transmission format before they can be sent from one process to another via the Protocol. An effective strategy is to define several formats and select the most appropriate one at run-time, adding to the Protocol a mechanism for format negotiation. Format negotiation would be another responsibility of the environment and could thus be made completely invisible to the applications program. 7a

Suggested below are two transmission formats. The first is a 36-bit binary format for use between 36-bit machines, the second an 8-bit binary, "universal" format for use between dissimilar machines. Data objects are fully typed in each format to enable the environment to automatically decode and internalize incoming parameters should it be desired to provide this service to the applications program. 7b

PCPB36, For Use Between 36-Bit Machines 7c

Bits 0-13 Unused (zero) 7c1
Bits 14-17 Data type 7c2
- EMPTY =1 INTEGER=4 LIST=7
- BOOLEAN=2 BITSTR =5
- INDEX =3 CHARSTR=6

Bits 18-20 Unused (zero) 7c3
Bits 21-35 Value or length N 7c4
- EMPTY unused (zero)
- BOOLEAN 14 zero-bits + 1-bit value (TRUE=1/FALSE=0)
- INDEX unsigned value
- INTEGER unused (zero)
- BITSTR unsigned bit count N
- CHARSTR unsigned character count N
- LIST unsigned element count N

Bits 36- Value 7c5
- EMPTY unused (nonexistent)
- BOOLEAN unused (nonexistent)
- INDEX unused (nonexistent)
- INTEGER two's complement full-word value
- BITSTR bit string + zero padding to word boundary
- CHARSTR ASCII string + zero padding to word boundary
- LIST element data objects

PCPB8, For Use Between Dissimilar Machines 7d

Byte 0 Data type 7d1

EMPTY =1 INTEGER=4 LIST=7
BOOLEAN=2 BITSTR =5
INDEX =3 CHARSTR=6

Bytes 1- Value 7d2

EMPTY	unused (nonexistent)
BOOLEAN	7 zero-bits + 1-bit value (TRUE=1/FALSE=0)
INDEX	2-byte unsigned value
INTEGER	4-byte two's complement value
BITSTR	2-byte unsigned bit count N + bit string + zero padding to byte boundary
CHARSTR	2-byte unsigned character count N + ASCII string
LIST	2-byte element count N + element data objects

APPENDIX B: THE EXPANDED PROCEDURE CALL PROTOCOL 8

The Protocol that results from the extensions proposed in this paper is summarized below. The reader should note the concise syntactic description made possible by the underlying notion of PCP data types. 8a

Parameter list masks have been included not only as additional parameters of the CALL message, as proposed in the paper, but as arguments of the TAKEPROCEDURE and SGNLPROCEDURE system procedures as well. Throughout the Protocol description, "MASK" is shorthand for: 8b

[LIST (variable [CHARSTR], ...)] 8b1

Messages 8c

```
LIST (route INDEX, opcode INDEX CALL=1, tid [INDEX],
      pkh [INDEX], procedure CHARSTR, arguments LIST,
      argumentlistmask MASK, resultlistmask MASK)                8c1
LIST (route INDEX, opcode INDEX RETURN=2, tid INDEX,
      outcome BOOLEAN, results LIST)                             8c2

   If OUTCOME is FALSE,
      RESULTS is LIST (error INDEX, diagnostic CHARSTR)
```

Process-Related System Procedures 8d

```
INIPROCESS (program CHARSTR,
            credentials LIST (user CHARSTR, password CHARSTR,
                              account CHARSTR))                  8d1
ALOPORT    (-> ph INDEX, computer CHARSTR, port)                 8d2
CNNPORT    (ph INDEX, computer CHARSTR, port)                    8d3
DCNPORT    (ph INDEX)                                            8d4
CRTROUTE   (mycode INDEX, oldcode [INDEX]
            -> code INDEX, ph INDEX)                             8d5
DELROUTE   (yourcode INDEX)                                      8d6
```

Package-Related System Procedures 8e

```
OPNPACKAGE (packages LISTofCHARSTRs -> pkhs LISTofINDEXs)        8e1
CLSPACKAGE (pkhs LISTofINDEXs)                                   8e2
```

Variable-Related System Procedures 8f

CRTVARIABLE (variable CHARSTR, value) 8f1
DELVARIABLE (variable CHARSTR) 8f2
RDVARIABLE (pkh INDEX, variable CHARSTR,
substructure [LISTofINDEXs] -> value) 8f3
WRVARIABLE (pkh INDEX, variable CHARSTR,
substructure [LISTofINDEXs], value) 8f4

Procedure-Related System Procedures 8g

TAKEPROCEDURE (tid INDEX, yourtid BOOLEAN, parameters LIST,
argumentlistmask MASK, resultlistmask MASK) 8g1
SGNLPROCEDURE (tid INDEX, yourtid BOOLEAN, parameters LIST,
parameterlistmask MASK) 8g2
HELPPROCEDURE (tid INDEX, number INDEX, information
-> solution) 8g3
NOTEPROCEDURE (tid INDEX, number INDEX, information) 8g4
ABRTPROCEDURE (tid INDEX) 8g5

APPENDIX C: SUMMARY OF RTE PRIMITIVES 9

The DPS primitives made available to the applications program as a result of the Model extensions proposed in this paper are summarized below. Collectively, they provide the applications programmer with a powerful and coherent set of tools for constructing distributed systems. Some of the primitives (for example, CRTPROCESS and DELPROCESS) are necessary elements of a "network operating system (NOS)", into which DPS may itself one day evolve. 9a

Processes 9b

CRTPROCESS (computer, program, credentials -> ph) 9b1
DELPROCESS (ph) 9b2
ITDPROCESS (ph1, ph2 -> ph12, ph21, ih) 9b3
SEPPROCESS (ih) 9b4

Packages 9c

OPNPACKAGE (ph, packages -> pkhs) 9c1
CLSPACKAGE (ph, pkhs) 9c2

Variables 9d

CRTVARIABLE (ph, variable, value) 9d1
DELVARIABLE (ph, variable) 9d2
RDVARIABLE (ph, pkh, variable, substructure -> value) 9d3
WRTVARIABLE (ph, pkh, variable, substructure, value) 9d4

Procedures 9e

CALLPROCEDURE (ph, pkh, procedure, arguments, argumentlistmask,
resultlistmask, -> outcome, results, tid) 9e1
LINKPROCEDURE (tid, arguments, argumentlistmask,
resultlistmask, -> outcome, results) 9e2
SGNLPROCEDURE (tid, parameters, parameterlistmask) 9e3
HELPPROCEDURE (tid, number, information -> solution) 9e4
NOTEPROCEDURE (tid, number, information) 9e5
ABRTPROCEDURE (tid) 9e6

APPENDIX D: ADDITIONAL AREAS FOR INVESTIGATION 10

Although the expanded distributed programming system developed in this paper and summarized in the previous appendix is already very powerful, many additional aspects of process interaction remain, of course, to be explored. Among the additional facilities that the Protocol must eventually enable the environment to provide are mechanisms for: 10a

(1) Queuing procedure calls for long periods of time (for example, days). 10a1

(2) Broadcasting requests to groups of processes. 10a2

(3) Subcontracting work to other processes (without remaining a middleman). 10a3

(4) Supporting brief or infrequent inter-process exchanges with minimal startup overhead. 10a4

(5) Recovering from and restarting after system errors. 10a5

REFERENCES 11

1. White, J. E., "A High-Level Framework For Network-Based Resource Sharing," Submitted for publication in the AFIPS Conference Proceedings of the 1976 National Computer Conference. 11a

2. Watson, R. W., Some Thoughts on System Design to Facilitate Resource Sharing, ARPA Network Working Group Request for Comments 592, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, November 20, 1973 (SRI-ARC Catalog Item 20391). 11b

3. White, J. E., DPS-10 Version 2.5 Implementer's Guide, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, August 15, 1975 (SRI-ARC Catalog Item 26282). 11c

4. White, J. E., DPS-10 Version 2.5 Programmer's Guide, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, August 13, 1975 (SRI-ARC Catalog Item 26271). 11d

5. White, J. E., DPS-10 Version 2.5 Source Code, Augmentation Research Center, Stanford Research Institute, Menlo Park, California, August 13, 1975 (SRI-ARC Catalog Item 26267). 11e

6. Bobrow, D. G., Burchfiel, J. D., Murphy, D. L., Tomlinson, R. S., "TENEX, a Paged Time Sharing System for the PDP-10," Communications of the ACM, Vol. 15, No. 3, pp. 135-143, March 1972. 11f

FIGURE LIST 12

Elements of a Distributed Programming System

5-JAN-76

James E. White
Augmentation Research Center

Stanford Research Institute
Menlo Park, California 94025

(415) 326-6200 x2960

This paper suggests some extensions to the simple Procedure Call Protocol described in a previous paper (27197,). By expanding the procedure call model and standardizing other common forms of inter-process interaction, such extensions would provide the applications programmer with an even more powerful distributed programming system.

The work reported here was supported by the Advanced Research Projects Agency of the Department of Defense, and by the Rome Air Development Center of the Air Force.

This paper will be submitted for publication in the Journal of Computer Languages.

NELC - VOLPE

< AJOURNAL, 34375.NLS;1, >, 30-JAN-76 08:45 XXX ;;;; .HJOURNAL="DCE 29-JAN-76 18:17 34375"; Title: .H1="29 Jan 75 visit by Joe Volpe of NELC, assessing NLS for C3 experiments"; Author(s): Douglas C. Engelbart/DCE; Distribution: /JCN([ACTION]) SGR([ACTION]) PKA([ACTION]) SRI-ARC([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: DCE; .IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PES;

Joe Volpe of Naval Electronics Laboratories Center (NELC), San Diego, stopped by ARC this afternoon.

Net-mail: NELC3030@ISIA (with note, "For Volpe")
Phone: (714) 225-6844

He is working on R&D for Command Control Communication (C3) in association with the ARPA IPTO C3 Program being developed by Commander Floyd Hollister. Joe is following up on a course of action agreed upon by Hollister when he visited ARC on 13 Jan -- i.e. to have some of the systems people at NELC learn enough about NLS to assess its potential in their C3 Program.

Basic questions involved in assessment (as quoted by Joe): "What is NLS advertised to be able to do?" and, "What is the relevant overlap with what is applicable for the C3 experimentation?" Dr. John Schill at NELC will be the principle person to delve into NLS; they had already contacted Susan and Pamela about having Pamela come down to San Diego next week to provide some standard training, as a launching action for their assessment learning.

Joe was going to get some DNLS demonstration this afternoon, and had stated an explicit interest in seeing some of the graphics capability. Elizabeth Michael was set to povide the demo. When Joe arrived, we first sat down in my office (Joe, me, Elizabeth, and Dick Watson) to get clear about the course of action that should be followed to provide them background and experience suitable for the assessment.

We talked about an hour; Joe described some basic characteristics of the systems they were assessing (e.g. of ARPA- or Navy-supported origin, suitable for experimention in a special environment they are creating, etc). He mentioned that "C3" could just as well be called "Mangement" system; that the support required of the system by different commanders varied considerably depending upon their "management" style, and thus the system must be adaptable. He had participated in the evolution of a fairly sophisticated computer-based system, and mentioned how the system seemed to reach a peak of enthusiastic and skillful use soon after evolution stopped, and that some time later its use had decayed considerably. We agreed that the dynamics of how systems were accepted and how their use was maintained by different organizations was something needing to be learned about. I said that we hoped their C3 R&D would include pursuit of an understanding of these factors.

Joe frankly stated that he had formed a rather dim view of NLS himself, based upon a personal experience in trying to develop a document with somebody in Washington whom he understands to have been trained in NLS. We said that our business was to support the applications architects, and that we didn't mind at all dealing with such impressions, as long as we would be given what we consider a fair chance to present our case and to provide appropriate experiences for the person making the assessment. Under a concern about a hastily arranged training session, far from the nearest TIP, we pointed out that poor terminals and communications would make it extremely difficult to train, learn, do assessment exercises, etc. on a fair basis.

The end agreement is that Pamela would go down to NELC early next week (assuming that appropriate arrangements and agreements are made with Susan and Jim Norton); and that tentatively it would be expected that John Schill would spend the week of 9 Feb here at ARC, getting a private adaptation of the AKW Seminar. I tentatively put a $1500 figure as our charge for a special, one-person Seminar. We will see if the charge can be set up as a flat purchase for a standard course, so NELC can pay in a simple manner (i.e., without going through some contract account). Joe seemed to think that this approach was feasible.

Joe was then given a short demo by Elizabeth. He took the following documents away with him:

Engelbart, "Coordinated Information Services for a Discipline- or Mission-Oriented Community," (12445,)
Engelbart, Norton, Watson, "The Augmented Knowledge Workshop," (14724,)
Irby, "CML Paper" (27266,)
Lentman, "NLS File System" (27292,)
Watson, "Design of a User Interface,"
ARC brochure, "Seminar on the Augmented Knowledge Workshop"
Meyer, "Executive Information Tools" (34111,)

< AJOURNAL, 34403.NLS;1, >, 4-FEB-76 06:35 XXX ;;;; .HJOURNAL="ESV 3-FEB-76 18:56 34403"; Title: .H1="Better Use of the Statistics We're Collecting"; Author(s): E. S. VonGehren/ESV; Distribution: /AID([ACTION]) ; Sub-Collections: NIC AID; Clerk: ESV; .IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PES; Origin: < VONGEHREN, TEMP.NLS;2, >, 30-JAN-76 06:56 ESV ;;;; .DCase=0; .ybs=1; .h2=".gd;"; .snf=72; .yb1=1; .pn=0; .pes; .psnow=[1,100]; .H1="<draft>.split;vongehren, e s";####;

Fellow KWACers -

Here at AMC we are building a user community which is growing every day and which we project for the not too distant future, will exceed the community which ARC is building. We hope that we will be able to do this with a minimum of "unworkable" decisions. Part of this is knowing what's going on now and forcasting future trends.

I have recently been speaking with Bud Pine about this and asked if the user statistics which are collected could show us some user trends. In particular what we would like to know is:

what is the distribution of our usage time - i.e., what is the clustering we experience during our peak hours? (This statistic, unfortunately may be effected by system load; when the load is high at office-1 and response is slow we tend to put off our "serious" work.) We would like to see this statistic as an AMC composite and by individual.

What is the average length of time for any login session (we don't have the luxury to be connected all day)? This we would like to see as a composite and by individual.

How much time are we spending in each of the "subsystems" (don't forget to include time detached)? Composite and individual.

There may be other statistics that you can recommend; I would appreciate any suggestions you have to offer. Bud said that there is a lot of data which is being collected and that if interest is expressed the routines could be built to pull it together. I would like both your show of interest and your suggestions for better use of the data. AMC needs to know more about our usage patterns, and if you're building for the future as we are, I suspect you do too.

Ed

I think your Help working paper 27495 is an accurate representation of the decisions made Monday except for 4C1B. I would re-word two other paragraphs to clear up some ambiguity.

The paragraph at 4C1B states:

> If in outline mode when a BUG or menu number is given, HELP will show the designated topic in full view mode. If a word rather than a menu number is typed, HELP will look up the word without changing view modes.

This and the paragraph following it are interesting suggestions but I don't remember coming to agreement on them in the meeting. I can see a few problems.

> It would make "locator" type cross-file functions difficult. (his, I think, is also something that DSM mentioned the NIC would like to be able to do easily.
>
> It would make finding a particular title under other titles in the same file difficult. (The default viewspecs for outline contain "eb".)
>
> Another draw back is the user couldn't just point with the mouse to a reference link or to get an outline view.
>
> Finally, it is "inconsistent with NLS" (not that I think that in itself is always a valid argument) but it is also confusing to have two different functions for the same action depend on what viewspecs are in effect. It is more straight forward to have different functions represented by different explicit actions on the part of the user.
>
> > Part of the confusion arises from having to explain to the user and the user having to remember that the same action sometimes does one thing and sometimes does another. However, even if the user was familiar with the rules, there are times when it is difficult to tell just by looking at the text (expecially in DNLS) whether you are looking at a "FULL" view of a table or an "OUTLINE" view of a plex that is set up to look like a table. For examples, see <isic, xhelp, nls, prompts>, <isic, xhelp, nls, infileaddress>.

The obvious solution is to have ^D always mean the OUTLINE function and pick some other character to mean FULL. When using TYPEIN, this is consistent with the NLS convention of having different codes on the end of an address indicate different views. I would propose RETURN be used to always mean the FULL function. The user should also be able to define whatever key she or he wants for this function.

From the mouse we already have a function which means "recognize this word". Currently it only works for the directory and file name levels but it will eventually be extended to the statement name level. It makes sense to use this function to explicitly point to a word with the mouse.

The following are the two paragraphs I would re-word.

I would change 4A2B "Link taking" to read:

> If the user should also be able to asks to see a specific statement in an outline view, and the statement contains a link, then the link will be taken as in the Jump (to) Link command. Parenthetical comments will not be taken as links, only items in angle-brackets. The user will be able to see links unless they are hidden by line clipping viewspecs or some future comment facility.

I would change 4C6 "Last" to read:

Prints the last view shown. May have to restore the viewing mode. The previous views are remembered in the order in which they are viewed (including any file changes that may occur) and the current view is forgotten when the last view is requested.

Other comments: it seems "Outline" and "Full" commands would change the current view as well as the mode if no TYPEIN is specified.

THE TNLS SECOND COURSE OUTLINE: INTRODUCTION TO STRUCTURE AND VIEWING

ARC-ADG

2 MAR 76

Applications Development

Augmentation Research Center
Stanford Research Institute
Menlo Park, California 94025

INTRODUCTION TO TNLS

AKW = Augmented Knowledge Workshop

PURPOSE OF SYSTEM: Augmentation of Knowledge Work

GOAL: To provide computer based tools to accomplish all aspects of knowledge work with an emphasis on collaboration.

OVERVIEW of system

NLS = oN Line System

TNLS = Typewriter Version

CAPABILITIES OF SYSTEM:

- Composing
- Editing
- Studying
  - Structuring
  - Browsing - viewing
- Printing
- Publishing
- Communicating -

  sending and receiving mail, messages, documents; teleconferencing; etc.
- Storing and retrieving -

  record keeping, library services, data bases, searching, etc.
- Calculating

Course Organization

NLS COURSE LEVEL:

NLS training is divided into five courses for ease of learning. Each level corresponds to what can be covered at one time. The things introduced at each level are determined by difficulty, usefulness, complexity, and quantity (i.e., so that there is not an excessive amount to cover at any one time).

Each level in the series of NLS courses contains most of the commands from the previous level for review in addition to the commands to be introduced (which are marked by an *).

BASIC TNLS
This is the first course level (basic) which covers those commands necessary to enter, edit, and "mail" typewritten information. It has a special structure and is published in the Journal (see -- Journal, 33874,).

*INTRODUCTION TO TNLS STRUCTURE AND VIEWING
This is the second course which introduces NLS structure (hierarchical) and special tools for viewing structured information ("viewspecs").

NLS is divided under headings for the purposes of this course. The commands under each heading can be used to perform the general operation denoted by the heading, e.g., "printing" includes commands that cause the system to print in various ways. Certain headings are introduced in later courses.

COURSE HEADINGS:
1. GETTING TO NLS
2. STRUCTURE
3. PRINTING
4. ADDRESSING
5. EDITING
6. COMMUNICATING
7. TROUBLE SHOOTING AND HELP

*NOTE: TO BE EFFECTIVE, THIS COURSE MUST BE PRECEDED BY THE BASIC COURSE AND SUFFICIENT TIME TO PRACTICE AND BECOME THOROUGHLY FAMILIAR WITH THE BASIC COURSE MATERIAL.

DEFINITIONS FOR THE COURSE OUTLINE

<SP> = You type a space.

[ ] = Comments and suggestions will appear in brackets.

Uppercase words = You specify the appropriate information for that command phrase, e.g., TYPEIN.

OK or CR = you type a Carriage Return; prompted by OK:.

CTRL = hold down the control (CTRL) key WHILE typing the specified character.

<ESC> = the ESC, ALT MODE or ESCAPE key on your terminal.

BASE C: = the TNLS ready signal. It means that you can type in an editing or file handling command (like home base...).

SEND C: = the Sendmail subsystem ready signal. It means that you can type in a Sendmail command.

ADDRESS: = Specify a location in an NLS file. End it with an OK. For current location, just type OK. Prompted by A:.

COMMANDS = You type some characters to tell the computer what to do. The characters you type are represented by the uppercase letters in each "command word"; the rest are lower case.

CONTROL MARKER (CM) = WHERE YOU ARE: Where the computer thinks you are pointing to (to some character in some file); you may move it by specifying an ADDRESS; this is where your command will be done. Note: your address must be followed by an OK or a Carriage Return.

*LEVEL-ADJUST: specifies level relative to addressed statement -- type any number of u's [for up], d's [for down] followed by an OK, or just an OK for the same level, prompted by L:.

*STRUCTURE: Statement or Branch or Group, prompted by C:

*STRING: Character or Word or Text, prompted by C:

*TYPEIN = a string of characters from the keyboard, ending with an OK, prompted by T:. [TYPEIN has a special form when a FILENAME or Link or Ident is called for (You can tell from the noise words)].

*VIEWSPECS: a string of one or more viewspec characters followed by OK, prompted by V: [type just OK if no viewspecs are to be entered]

* INTRODUCTION TO TNLS STRUCTURE AND VIEWING

1. GETTING TO NLS

THE TERMINAL AND USE:
See the "Basic TNLS-8 Course" [You usually have to dial a telephone number and place the receiver in your terminal's cradle]

ARPA NETWORK [for a new connection where you dial in]

NETWORK CONNECTION:

[I] Type E [to get the Network's attention]

[II] Type @ O <SP> 43 CR [to open a connection to Office-1, Host 43; BBNB is 49]

You now should be connected to TENEX and will receive the usual notice to that effect and the @ (the TENEX ready signal).

TENEX

LOGIN PROCEDURE:

[III] Type LOG <SP> USERNAME <SP> PASSWORD <SP> CR
[The last space fills in your account number automatically; you're then ready to call NLS]

*For some systems to accept lower case characters, it may be necessary to type the TENEX command:

NO RAISE CR

CALLING NLS:

[IV] Type NLS CR [it's not necessary to call NLS more than once during one login session]

To Go to Tenex from NLS (as a subsystem):

Goto (subsystem) Tenex OK
QUIT CR [to return to where you were]

* Other ways to get to Tenex from NLS

*Quit Nls CR [to leave NLS]
CONTINUE CR [to return to where you were]

To leave the system, logout in NLS [from BASE only]:

<SP> Logout OK [you type only the L of logout]

To close the network connection:

@ C CR

2. ORGANIZATION OF THE SYSTEM

FILES & DIRECTORIES

Information in the origin ("parent") statement of a file
The origin statement contains the file name, version number, the date and time of last modification, the ident of the last person to modify the file, and 4 semicolons. The statement should not be edited. It is numbered 0, but no number will be printed.

File names

* Types of files [indicated by filename extensions]

  * TXT = sequential file which can be copied into NLS
  COPY = a temporary sequential file, usually a message
  NLS = an NLS file which you can load and read in NLS

Load File:

Load File FILENAME OK [FILENAME WILL BE ECHOED]

User creation of files:

<SP>CReate File FILENAME OK

To see a list of all your files:

<SP>SHow Directory (of) OK OK

FILE STRUCTURE

STATEMENT: The basic element of structure in a file
[each has a statement number]

*Relationship between statements:

*All statements have a "source" (may be the Origin) and may have statements as "substructure".

*STRUCTURES made up of statements:

BRANCH: a statement plus all substructure (if any)

GROUP: set of contiguous branches at the same level and with same source

3. PRINTING: to see specified view of stored information

[To see anything in TNLS you must print it]

Printing on a terminal:

Print File OK

Print Rest OK

Print Journal (mail) OK

\ [easy print, typing a \ prints the statement where you are]

* LF [line feed prints the next statement regardless of level]

* ^ [print back one statement regardless of level]

* Print STRUCTURE (at) ADDRESS VIEWSPECS

VIEWSPECS: to specify what you see, use the characters below when prompted with a V: and end with an OK.

w = Default, all lines & levels (show all of the text)
m/n = numbers on/off
y/z = blank lines on/off

[have instructor set these for your default]

* To clip levels and lines, use lower case viewspecs including:
a/b - show one level less/more
c/d - show all levels/show first level
e - show referenced statement level
g/h - show branch only/show all branches
q/r - show one line less/more
s/t - show all lines/show first lines only
w/x - show all lines, all levels/show one line, one level

* SIDS (Statement Identification Numbers)
I/J - SIDS on instead of statement numbers/statement numbers instead of SIDS (when m is on)
[can be used in place of statement numbers in NLS]
G/H - Numbers (SIDS or statement numbers) right side/left side (when viewspec m is on)

4. ADDRESSING

Control Marker concept = where you are [travels left to right]

Jump to a new address:
Jump (to) Address ADDRESS VIEWSPECS OK

* To tell where the Control Marker is:

  / slash command shows Control Marker context

  * . period command shows statement number and character number;

  * [Note that addressing can be combined with editing, you do not have to move the marker separately, i.e., you can give an ADDRESS in an editing command]

ADDRESSING WITHIN A FILE
Use the following which will be referred to as IN-FILE-ADDRESS:

STATEMENT NUMBER:
Automatically assigned to a statement, but not included in it.

TYPEIN SEARCH: "TYPEIN" [must be surrounded by quotes]
where TYPEIN = the text to be searched for.

* SID: Statement IDentifier: another number assigned to each statement, it's a permanent number (despite editing changes) [always beginning with a zero]

* IN-FILE-ADDRESSes within one statement:

  *+e skip to end (last character) of statement
  [always use a plus sign]

  *+f skip to front (first character) of statement
  [always use a plus sign]

* ADDRESSING BY JUMPING
  [Note: Use the Jump command when you do not want the STRUCTURE at the new location printed]

  TO FIND A WORD OR STRING OF CHARACTERS (TYPEIN) [no quotes]:

  * Jump (to) Word First TYPEIN VIEWSPECS OK

  * Jump (to) Word Next TYPEIN VIEWSPECS OK

  * Jump (to) Content First TYPEIN VIEWSPECS OK

  * Jump (to) Content Next TYPEIN VIEWSPECS OK
    [type a CTRL B for TYPEIN in response to RPT: to continue to search for the same thing]

  * TO JUMP BY STRUCTURE:

  * Jump (to) Origin ADDRESS VIEWSPECS OK

  * Jump (to) End (of Branch) ADDRESS VIEWSPECS OK

ADDRESSING BETWEEN FILES AND DIRECTORIES:

  * To address another file in your directory you need to add the FILENAME to the addresses within a file. To address a file in another user's directory, you need to add their DIRECTORY name as well as the filename. FILENAME and DIRECTORY must be followed by commas.
  [These may be used after A: in any command]

  * To address another file:

  * A: FILENAME,IN-FILE-ADDRESS OK

  * [If an IN-FILE-ADDRESS is not specified it will be statement 0]

  * To address another user's file:

  * A: DIRECTORY,FILENAME,IN-FILE-ADDRESS OK
  [e.g.: Copy Branch (from) BAIR,JHB,1 OK (to) 3a OK]

LINKS: special forms of text that may be used for addressing and other purposes.

* Characteristics of Links:

  * -- it is text in a statement rather than typed in after the A:

  * -- must be surrounded by angle brackets < > (or parentheses)

  * -- may contain any logical Address

  * -- it may include viewspecs that will take effect at the address in the link

  * -- the following forms are valid:

    * <DIRECTORY,FILENAME,IN-FILE-ADDRESS:VIEWSPECS>

    * [Without optional Viewspecs:]
      <DIRECTORY,FILENAME,IN-FILE-ADDRESS>

    * [or in current directory:]
      <FILENAME,IN-FILE-ADDRESS>

    * [or in current file:]
      <IN-FILE-ADDRESS>

      Note that the different fields default to the current value if not specified (the same as addresses).

  * -- may include things other than addresses and/or viewspecs [which will be covered by your trainer as appropriate to your application]

* To use a link that has been put in a statement, give the Address of the Statement that contains the link and the letter l preceded by a period after any A: , for example:

  * Jump (to) Address IN-FILE-ADDRESS .l OK

* TO GO BACK TO PREVIOUS FILES:

  * Jump (to) File Return OK ANSWER OK
    [N for ANSWER - next filename in stack will be echoed; repeat for file before that;
    Y for ANSWER selects that file]

5. EDITING

Syntax: VERB NOUN A: ADDRESS(ES) (L: LEVEL) (T: TYPEIN) OK (OK? OK)

* STRING and STRUCTURE = "nouns":

  * STRING: [one of the following command words that refers to part of a statement]

    * Character

    * Word [note that the system readjusts spaces]

    * Text [two addresses necessary]

  * STRUCTURE: [one of the following command words that refers to one or more statements]

    Statement

    * Branch

    * Group [two addresses necessary]

EDITING COMMANDS = "verbs":

INSERT

Insert Statement (to follow) ADDRESS LEVEL-ADJUST TYPEIN OK

* The LEVEL-ADJUST determines the level of a statement at a new location -- it is one of the following ended by an OK:

  Just an OK = same level

  * u [position up a level from referenced statement]

  * d [position down a level from referenced statement]

* Insert STRING (to follow) ADDRESS TYPEIN OK

Continue to insert: CTRL E instead of OK puts you in the Enter statement mode. Type a CTRL X to get out.

DELETE

Delete File TYPEIN OK

Delete STRUCTURE (at) ADDRESS OK

* Delete STRING (at) ADDRESS OK

SUBSTITUTE

Substitute STRING in STRUCTURE (at) ADDRESS OK
(New STRING) T: TYPEIN OK
(Old STRING) T: TYPEIN OK Finished? S/Y/N: Y [for yes]
Substitutions made: number

[will replace the old STRING with new STRING every time it finds it in the STRUCTURE.]

MOVE

Move STRUCTURE (from) ADDRESS (to follow) ADDRESS LEVEL-ADJUST OK

*Move STRING (from) ADDRESS (to follow) ADDRESS OK

COPY

Copy STRUCTURE (from) ADDRESS (to follow) ADDRESS LEVEL-ADJUST OK

*Copy STRING (from) ADDRESS (to follow) ADDRESS OK

*REPLACE

*Replace STRUCTURE (at) ADDRESS (by) TYPEIN OK

*TRANSPOSE

*Transpose STRUCTURE (at) ADDRESS (and) ADDRESS OK

*APPEND [joins two statements together to form one statement]

*Append Statement (at) ADDRESS (to) ADDRESS (join with) TYPEIN OK
[TYPEIN is text that will be added where the old and new statements join]

*BREAK [to break a statement into two statements after the visible you point to]

*Break Statement (at) ADDRESS LEVEL-ADJUST OK

UPDATE FILE [not imperative, but good practice]

Update File OK

* Update File Compact OK
[Note: this will ensure the efficient storage of a file that has been edited extensively. To find out the percent of efficiently used storage, use <SP>SHow File Status OK]

6. COMMUNICATING with other users

SENDMAIL SUBSYSTEM and the Journal

Goto (subsystem) Sendmail OK

Interrogate Command

Interrogate OK
(distribute for action to:) IDENT/.LASTNAME
[You may give a series of IDENTS
or .LASTNAMES separated by commas]
(distribute for information-only to:) IDENT/.LASTNAME
(title:) TYPEIN
(type of source:) Message or Statement or Branch or Group
or File (at) ADDRESS
(show status?) ANSWER
(distribute the mail now?) ANSWER

* Individual commands: instead of or in addition to Interrogate, you may use the following:

* Title TYPEIN OK

* Distribute (for) Information (Only) (to)
IDENT/.LASTNAME OK
[You may give a series of IDENTS
or .LASTNAMES separated by commas]

* Distribute (for) Action (to) IDENT/.LASTNAME OK

* Comments TYPEIN OK

* To send a message or statement:

* Message TYPEIN OK

* <SP>STatement (at) ADDRESS OK

* To send a structure or file:

* <SP>Group (at) ADDRESS OK

* Branch (at) ADDRESS OK

* File ADDRESS OK

* <SP>SHow Status OK

* Send (the mail) OK

* To identify a user by lastname or ident:

  * <SP>SHow Record (for ident) .LASTNAME OK [precede by a period]

  * <SP>SHow Record (for ident) IDENT OK

Mailbox = (journal) branch of your initial file -- sendmail automatically inserts citation

To leave the Sendmail subsystem when you are done:
Quit OK [returns you to Base]

SEND MESSAGE (Tenex)

Goto (subsystem) Tenex OK

```
SND CR      [The system will prompt you:]
(To (? for help):) TYPEIN CR  [lastnames separated by comma]
(cc (? for help):) TYPEIN CR  [lastnames separated by comma]
(subject:) TYPEIN CR     [subject of your message]
(message:) TYPEIN
CRTL Z      [to terminate the message]
(Q, S, ?, carriage return:) CR  [to send the message]
QU CR       [to return to NLS]
```

* To abort a send message before sending it type a CTRL C

Linking (Tenex)

```
first: Goto (subsystem) Tenex OK
WHE<ESC>re (is) USERNAME CR  [do not link when user is in
                              SNDMSG, OUTPRC, NOUTPRC, or XLIST]
LIN<ESC>k (to)  USERNAME  CR [precede comment with ; end with CR,
                              repeat every 3 lines]
BREAK CR   [to break the link; only one person must do this]
QU CR  [returns you to NLS]
```

7. TROUBLE SHOOTING AND HELP

Immediate assistance from the system:

Type ? for commands or needed information after any prompt.

* HELP:

*Type CTRL Q for help concerning what you are doing or type H for Help command (after typing H you can type any word in NLS you wish to know about). CTRL X gets you out of Help and back to where you were.

* Help TYPEIN OK

* Help OK

System Status:

Two CTRL T's [Note the words RUNNING or WAIT -- WAIT means the computer is waiting for you to do something]

*<SP>SHow <SP>Disk (space status) OK [each user has a certain allocation of pages]

Send a message or sendmail item to: FEEDBACK

Call SRI/ARC, (415 326-6200, ext.3630)

Link to FEEDBACK

Remedies:

CTRL C [use only in emergencies to get to TENEX]
RESET CR
NLS CR

*If over allocation:
<SP>EXpunge Directory OK
<SP>Trim Directory (no. of versions to keep) TYPEIN OK (really?) OK

Update File Compact OK [restores file more efficiently in computer]
Delete Modifications OK (really?) OK [destroys all changes since the last update!]

If your connection is broken:

Type @o <SP> 43 CR [Step 2 of the net login, p. 4]
To check if you are detached, use the where command:
WHERE <SP> USERNAME CR
If you are detached, instead of logging in, type:
ATT <SP> USERNAME <SP> PASSWORD <SP> CR
CTRL O [to wake up NLS if that's where you were, or:]
CTRL C NLS CR
If you are not logged in repeat STEP III of the Login Procedure, p. 4

PRACTICE

Primer ("TNLS-8 Primer," Journal Accession Number -- 32954,)

* Introductory TNLS Sample Sessions for TNLS Course Level 2
(Journal Reference Number -- 33405,)

* Use Strategies
If there is time available while your trainer is present, ask her to explain how to use the system to accomplish specific tasks, from daily routine tasks (such as message handling) to online compostion with multiple authors.

OTHER AVAILABLE COURSES:

* 3. INTERMEDIATE TNLS
This is the third formal course or level of expertise, and represents significant experience with the system. The Programs and Useroptions subsystems are introduced as well as Output Processing for printer formating.

* INTRODUCTION TO DNLS
This is an introduction to the display version of NLS designed to follow the second TNLS course (it requires an understanding of structure and viewing). It covers the use of the special workstation required for DNLS as well as the special ways of pointing and displaying information that are available.

* EXAMPLE OF STRUCTURE:

```
< BAIR, MENU.NLS;1, >, 28-JAN-75 17:29 JHB ;;;;
1 SOUP
   1A VEGETABLE
   1B CREAM OF MUSHROOM
2 ENTREE
   2A  FRIED CHICKEN
   2B  SALMON
      2B1 WITH CREAM SAUCE
   2C  PRIME RIBS
3 DESSERT
   3A  PIE
      3A1  APPLE
         3A1A A LA MODE
      3A2  BLUEBERRY
   3B  ICE CREAM
      3B1  VANILLA
      3B2  PEPPERMINT
      3B3  MAPLENUT
      3B4  CHOCOLATE
4  BEVERAGE
   4A  TEA
   4B  COFFEE
```

Page 19

is 3-Color

insert provided

Leave NLS:	Goto(subsystem)Tenex <CR>	Quit Nls <CR>	<CTRL-C>
Back to NLS:	QUIT <CR>	CONTINUE <CR> <CTRL-X>	CONTINUE <CR> <CTRL-X>
Comments: When allowed	Can use Goto command after BASE C: or SEND C:	Can use Quit command after BASE C: or SEND C:	Can do <CTRL-C> anytime, in emergencies
Use with SNDMSG.	Goto command creates an Inferior TENEX. Do not type NLS again. Use QUIT after SNDMSG.	Can not CONTINUE after SNDMSG; so use Goto command instead.	Can not CONTINUE after SNDMSG; so use the Goto command instead when possible.

DIFFERENT WAYS TO GET BETWEEN NLS AND TENEX (EXEC)

EDITING COMMANDS

VERBS	NOUNS
Insert	Structure
Substitute*	Statement
Delete	Group (needs two addresses)
Move	Plex
Copy	Branch
Replace	String
Transpose	Text (needs two addresses)
Append	Word
Break	Character
Force (case)	Visible
Sort	Invisible
	Link

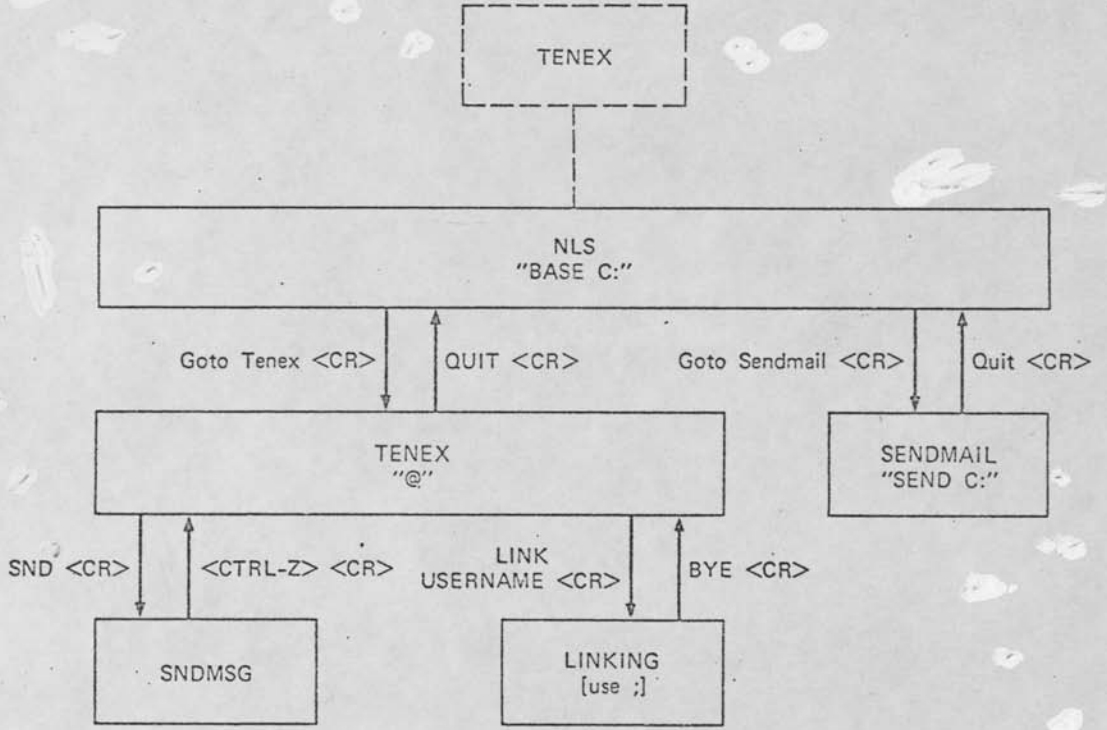
*Substitute command requires three command words.

QUICK REFERENCE SUMMARY OF SECOND COURSE LEVEL
TNLS EDITING COMMAND WORDS

SENDMESSAGE	SENDMAIL
SENT IN TENEX	SENT IN NLS
SENT TO USERNAMES (DIRECTORIES)	SENT TO IDENTS (INDIVISUALS)
DELIVERED IMMEDIATELY	DELIVERED SEVERAL TIMES DAILY
USED TO SEND IMPROMPTU MESSAGES	USED TO SEND ALREADY EDITED AND IMPROMPTU MESSAGES
NOT CATALOGUED	AUTOMATICALLY STORED AND CATALOGUED
NO AUTHOR COPY	AUTOMATIC AUTHOR COPY
READ IN TENEX (MESS)	READ IN NLS (PRINT JOURNAL)

SOME OF THE DIFFERENCES BETWEEN SENDMESSAGE AND SENDMAIL

COMMUNICATING



Journal Accession Number: 34668

The TNLS Third Course Outline: Intermediate NLS

ARC-ADG

5 MAR 76

Applications Development

Augmentation Research Center
Stanford Research Institute
Menlo Park, California 94025

*

NLS COURSE LEVEL:

Each level in the series of NLS courses contains most of the commands from the previous level for review in addition to the commands to be introduced (which are marked by an *).

*Most of the commands from Course I are not included.

*THREE -- INTERMEDIATE TNLS
This is the third session or level, and is intended for users who have completed courses I and II and have had experience with the system at level II. The Programs and Useroptions subsystems are introduced, Output processing is covered, and the Base and Sendmail subsystems are expanded.

* COURSE HEADINGS:
1. GETTING TO NLS
2. STRUCTURE
3. PRINTING
4. ADDRESSING
5. EDITING
6. COMMUNICATING
7. TROUBLE SHOOTING AND HELP
* 8. OUTPUT PRINTING
* 9. PROGRAMS
* 10. USEROPTIONS - CUSTOMIZATION OF NLS

*NOTE: TO BE EFFECTIVE, THIS COURSE MUST BE PRECEDED BY THE BASIC AND INTRODUCTORY COURSES AND SUFFICIENT TIME TO PRACTICE AND BECOME FAMILIAR WITH THE MATERIAL THEY COVER.

DEFINITIONS FOR THE COURSE OUTLINE

<SP> = You type a space.

[ ] = Comments and suggestions will appear in brackets.

Uppercase words = You specify the appropriate information for that command phrase, e.g., TYPEIN.

*OK or CR = one of the following:
a Carriage Return;
CTRL B -- executes the command and starts it over again up to the first place where you specify an address or other variable. You leave the Repeat Command mode by typing CTRL x.
CTRL E -- in the BASE subsystem only, places you in the Enter mode, entering a statement at the current marker location. You leave the Enter mode by typing CTRL X after the last OK to enter your last statement.

CTRL = hold down the control (CTRL) key WHILE typing the specified character.

<ESC> = the ESC, ALT MODE or ESCAPE key on your terminal.

ADDRESS: = Specify a location in an NLS file. End it with an OK. For current location, just type OK. Prompted by A:.

COMMANDS = You type some characters to tell the computer what to do. The characters you type are represented by the uppercase letters in each "command word"; the rest are lower case.

CONTROL MARKER (CM) = WHERE YOU ARE: Where the computer thinks you are pointing to (to some character in some file); you may move it by specifying an ADDRESS; this is where your command will be done. Note: your address must be followed by an OK or a Carriage Return.

*DESTINATION = ADDRESS OK
When referring to Group or Text, two ADDRESSES are needed.

LEVEL-ADJUST: specifies level relative to addressed statement -- type any number of u's [for up], d's [for down] followed by an OK, or just an OK for the same level, prompted by L:.

*SOURCE = ADDRESS OK
When referring to Group or Text, two ADDRESSES are needed.

*STRING: Character or Word or Visible or Invisible or Text or Link, prompted by C:

*STRUCTURE: Statement or Branch or Group or Plex, prompted by C:

TYPEIN = a string of characters from the keyboard, ending with an OK, prompted by T:. [TYPEIN has a special form when a FILENAME or Link or Ident is called for (You can tell from the noise words)].

VIEWSPECS: a string of one or more viewspec characters followed by OK, prompted by V: [type just OK if no viewspecs are to be entered]

*INTERMEDIATE TNLS COURSE OUTLINE

1. GETTING TO NLS

ARPA NETWORK [for a new connection where you dial in]

NETWORK CONNECTION:

[I] Type E [to get the Network's attention]

[II] Type @ O <SP> 43 CR [to open a connection to Office-1, Host 43; BBNB is 49]

You now should be connected to TENEX and will receive the usual notice to that effect and the @ (the TENEX ready signal).

TENEX

LOGIN PROCEDURE:

[III] Type LOG <SP> USERNAME <SP> PASSWORD <SP> CR
[The last space fills in your account number automatically; you're then ready to call NLS]

For some systems to accept lower case characters, it may be necessary to type the TENEX command:

NO RAISE CR

CALLING NLS:

[IV] Type NLS CR [it's not necessary to call NLS more than once during one login session]

*TENEX "EXECUTIVE"

Other ways to get to Tenex from NLS

Quit Nls CR [to leave NLS]
CONTINUE CR [to return to where you were]

2. ORGANIZATION OF THE SYSTEM

FILES & DIRECTORIES

Types of files [indicated by filename extensions]

TXT = sequential file which can be copied into NLS
COPY = a temporary sequential file, usually a message
NLS = an NLS file which you can load and read in NLS

* System creation of files: certain files are created automatically by the system and may have an effect on disk allocation, e.g., message.txt, message.copy, and other files that are necessary to support your NLS.

* [SHow Directory defaults to your directory -- you may see a list of the public files in other's directories]
<SP>SHow Directory (of) OK/T OK

FILE STRUCTURE

STATEMENT: The basic element of structure in a file
[each has a statement number]

Relationship between statements:

All statements have a "source" (may be the Origin) and may have statements as "substructure".

STRUCTURES made up of statements:

BRANCH: a statement plus all substructure (if any)

GROUP: set of contiguous branches at the same level and with same source

*PLEX: complete list of branches at the same level and with the same source.

*Other relationships between statements:

*END: last statement of branch

*UP: one level up from current statement

*DOWN: one level down from current statement

*BACK: immediately preceeding statement regardless of level

*NEXT: next statement regardless of level

*TAIL: last statement of plex at the level pointed to

*HEAD: first statement of plex

*SUCCESSOR: statement immediately succeeding current statement at same level with same source

*PREDECESSOR: statement immediately preceding current statement at same level with same source

3. PRINTING: to see specified view of stored information

LF [line feed prints the next statement regardless of level]

^ [print back one statement regardless of level]

Print STRUCTURE (at) DESTINATION VIEWSPECS

VIEWSPECS: to specify what you see, use the characters below when prompted with a V: and end with an OK.

* Or use the command:
<SP>SEt Viewspecs VIEWSPECS OK

* To list viewspecs in effect:
<SP>SHow Viewspecs OK

* To reset all viewspecs to the default values:
<SP>RESet Viewspecs OK

w = Default, all lines & levels (show all of the text)
m/n = numbers on/off
y/z = blank lines on/off

To clip levels and lines, use lower case viewspecs including:
a/b - show one level less/more
c/d - show all levels/show first level
e - show referenced statement level
g/h - show branch only/show all branches
q/r - show one line less/more
s/t - show all lines/show first lines only
w/x - show all lines, all levels/show one line, one level

* To format and show extra info, use uppercase viewspecs
A/B - level indenting on/off
C/D - snow/don't show statement names (explained in next section)
E/F - paginate/don't paginate
G/H - statement numbers right/left
K/L - show/don't show statement signatures

SIDS (Statement Identification Numbers)
I/J - SIDS on instead of statement numbers/statement numbers instead of SIDS (when m is on)
[can be used in place of statement numbers in NLS]
G/H - Numbers (SIDS or statement numbers) right side/left side (when viewspec m is on)

* Viewspecs may be combined -- the right most character has the final say. The effect is cumulative, e.g., V: xrc will show all levels, 2 lines.

4. ADDRESSING

To tell where the Control Marker is:

/ slash command shows Control Marker context

. period command shows statement number and character number;

[Note that addressing can be combined with editing, you do not have to move the marker separately, i.e., you can give an ADDRESS in an editing command]

*[Note the control marker location after complex editing, i.e., it moves with a moved structure, is at the last statement edited for the use of CTRL E, and so on]

* The slash / may be an ADDRESS element or a command.

* The backslash \ may also be an ADDRESS element or a command.

ADDRESSING WITHIN A FILE
Use the following which will be referred to as IN-FILE-ADDRESS:

STATEMENT NUMBER:
Automatically assigned to a statement, but not included in it.

CONTENT SEARCH: "TYPEIN" [must be surrounded by quotes]
where TYPEIN = the text to be searched for.

* "TYPEIN"=s [limits search to current statement]

* word-search: "word"=w
moves the CM to the next occurrence of that word.

* TAB command [repeats the previous search for word or content]

* STATEMENT NAMES: "name" a statement so it can be pointed to by typing its name in an ADDRESS, after any A: prompt. A name is the first set of characters in the statement between the delimiters.

* name-delimiters: Enclose, and define for system recognition, statement names. The default name delimiters are NULL NULL. This means that the first word of any statement is its name if no space precedes it.

*<SP>SEt Name (delimiters in) STRUCTURE (at) DESTINATION (left delimiter) CONTENT (right delimiter) CONTENT OK

*<SP>RESet Name (delimiters in) STRUCTURE (at) DESTINATION OK
[to default in useroptions]

*<SP>SHow Name (delimiters for statement at) DESTINATION OK

SID: Statement IDentifier: another number assigned to each statement, it's a permanent number (despite editing changes) [always beginning with a zero]

*Renumber a file's SIDs consecutively with:
<SP>RENumber Sids (in file) OK

IN-FILE-ADDRESSes within one statement:

* Letters PRECEDED IMMEDIATELY BY A PLUS (+) mean SKIP FORWARD, BY A MINUS (-) mean SKIP BACKWARD. A number between the plus or minus and the letter indicates the number of skips.

+e skip to end (last character) of statement
[always use a plus sign]

+f skip to front (first character) of statement
[always use a plus sign]

* l skip to link [+ or - preceding]

* w skip words [+ or - preceding]

* SHORT IN-FILE-ADDRESSes [precede by a period]:

* These address elements may be combined in series in the same address field along with any other address elements, and if logical will move the pointer to each address specified in sequence.
For example: Print Statement (at) .n.t OK OK will print the tail of the next plex.

* [A number before any of these letters indicates the number of moves (default for number is 1)].

* .l link [find and jump on the next link, see "Links"]

* .fr file return [to position in previous file]

* .r return [to previous control marker location within a file]

* .b back [one statement]

* .d down [one level]

* .e end [of branch]

* .h head [of plex]

* .n next [statement]

* .o origin [of file]

* .p predecessor [same level, same source]

* .s successor [same level, same source]

* .t tail [of plex]

* .u up [one level]

ADDRESSING BY JUMPING
[Note: Use the Jump command when you do not want the STRUCTURE at the new location printed]

TO FIND A WORD OR STRING OF CHARACTERS (CONTENT) [no quotes]:

Jump (to) Word First CONTENT VIEWSPECS OK

Jump (to) Word Next CONTENT VIEWSPECS OK

Jump (to) Content First CONTENT VIEWSPECS OK

Jump (to) Content Next CONTENT VIEWSPECS OK
[type a CTRL B for CONTENT in response to RPT: to continue to search for the same thing]

* TO FIND A STATEMENT BY ITS NAME:

* Jump (to) Name First CONTENT VIEWSPECS OK

* Jump (to) Name Next CONTENT VIEWSPECS OK

* Jump (to) Name Any CONTENT VIEWSPECS OK

* TO GO BACK TO PREVIOUS LOCATIONS WITHIN THE CURRENT FILE:

* Jump (to) Return OK ANSWER OK
[type an N for ANSWER - next flashback in stack will be echoed; repeat for file before that]

TO JUMP BY STRUCTURE:

* Jump (to) <SP>Next DESTINATION VIEWSPECS OK

Jump (to) Origin DESTINATION VIEWSPECS OK

* Jump (to) Back DESTINATION VIEWSPECS OK

Jump (to) End (of Branch) DESTINATION VIEWSPECS OK

* Jump (to) Tail DESTINATION VIEWSPECS OK

* Jump (to) Head DESTINATION VIEWSPECS OK

* Jump (to) Down DESTINATION VIEWSPECS OK

* Jump (to) Up DESTINATION VIEWSPECS OK

* Jump (to) Predecessor DESTINATION VIEWSPECS OK

* Jump (to) Successor DESTINATION VIEWSPECS OK

ADDRESSING BETWEEN FILES AND DIRECTORIES:

To address another file in your directory you need to add the FILENAME to the addresses within a file. To address a file in another user's directory, you need to add their DIRECTORY name as well as the filename. FILENAME and DIRECTORY must be followed by commas.
[These may be used after A: in any command]

To address another file:

A: FILENAME,IN-FILE-ADDRESS OK

[If an IN-FILE-ADDRESS is not specified it will be statement 0]

To address another user's file:

A: DIRECTORY,FILENAME,IN-FILE-ADDRESS OK
[e.g.: Copy Branch (from) BAIR,JHB,1 OK (to) 3a OK]

* You may add VIEWSPECS in an Address field in any of the above cases:

* A: ADDRESS:VIEWSPECS OK

* [Note that it is necessary to precede Viewspecs with a colon]

LINKS: special forms of text that may be used for addressing and other purposes.

Characteristics of Links:

-- it is text in a statement rather than typed in after the A:

-- must be surrounded by angle brackets < > (or parentheses)

-- may contain any logical Address

-- it may include viewspecs that will take effect at the address in the link

-- the following forms are valid:

<DIRECTORY,FILENAME,IN-FILE-ADDRESS:VIEWSPECS>

[Without optional Viewspecs:]
<DIRECTORY,FILENAME,IN-FILE-ADDRESS>

[or in current directory:]
<FILENAME,IN-FILE-ADDRESS>

[or in current file:]
<IN-FILE-ADDRESS>

* [or:] <:VIEWSPECS> [only the viewspecs will be changed]

Note that the different fields default to the current value if not specified (the same as addresses).

-- may include things other than addresses and/or viewspecs [which will be covered by your trainer as appropriate to your application]

To use a link that has been put in a statement, give the Address of the Statement that contains the link and the letter l preceded by a period after any A: , for example:

Jump (to) Address IN-FILE-ADDRESS .1 OK

* If there is more than one link in a statement, .1 will take the first link in the statement or the first link to the left of the CM.

TO GO BACK TO PREVIOUS FILES:

Jump (to) File Return OK ANSWER OK
[N for ANSWER - next filename in stack will be echoed; repeat for file before that;
Y for ANSWER selects that file]

5. EDITING

Syntax: VERB NOUN A: ADDRESS(ES) (L: LEVEL) (T: CONTENT) OK (OK? OK)

STRING and STRUCTURE = "nouns":

STRING: [one of the following command words that refers to part of a statement]

Character

Word [note that the system readjusts spaces]

Text [two addresses necessary]

* Visible [contiguous printing characters, readjusts spaces]

* Invisible [contiguous non-printing characters]

* Link [all characters between parentheses or <>]

STRUCTURE: [one of the following command words that refers to one or more statements]

Statement

Branch

Group [two addresses necessary]

* Plex

*To repeat a command up to the address or first variable: type <ESC> or CTRL B after BASE C:

INSERT

Insert Statement (to follow) DESTINATION LEVEL-ADJUST CONTENT OK

The LEVEL-ADJUST determines the level of a statement at a new location -- it is one of the following ended by an OK:

Just an OK = same level

u [position up a level from referenced statement]

d [position down a level from referenced statement]

Insert STRING (to follow) DESTINATION CONTENT OK

Continue to insert: CTRL E instead of OK puts you in the Enter statement mode. Type a CTRL X to get out.

DELETE

Delete File CONTENT OK

Delete STRUCTURE (at) DESTINATION OK

Delete STRING (at) DESTINATION OK

SUBSTITUTE

Substitute STRING in STRUCTURE (at) ADDRESS OK
(New STRING) T: TYPEIN OK
(Old STRING) T: TYPEIN OK Finished? S/Y/N: Y [for yes]
Substitutions made: number

[will replace the old STRING with new STRING every time it finds it in the STRUCTURE.]

MOVE

Move STRUCTURE (from) SOURCE (to follow) DESTINATION LEVEL-ADJUST OK

Move STRING (from) SOURCE (to follow) DESTINATION OK

COPY

Copy STRUCTURE (from) SOURCE (to follow) DESTINATION LEVEL-ADJUST OK

Copy STRING (from) SOURCE (to follow) DESTINATION OK

*Copy Directory (of) OK/T (to follow) DESTINATION LEVEL-ADJUST OK

REPLACE

Replace STRUCTURE (at) DESTINATION (by) CONTENT OK

*Replace STRING (at) DESTINATION (by) CONTENT OK

TRANSPOSE

Transpose STRUCTURE (at) DESTINATION (and) DESTINATION OK

*Transpose STRING (at) DESTINATION (and) DESTINATION OK

APPEND [joins two statements together to form one statement]

Append Statement (at) SOURCE (to) DESTINATION (join with) CONTENT OK
[CONTENT is text that will be added where the old and new statements join]

BREAK [to break a statement into two statements after the visible you point to]

Break Statement (at) DESTINATION LEVEL-ADJUST OK

*FORCE (case) [to change the case of characters]

* Force (Case) STRUCTURE (at) DESTINATION OK

* Force (Case) STRING (at) DESTINATION OK

* FORCE (case) Mode

* Force (Case) Mode Upper [or] Lower [or] First OK

* SORT [sorts statements alphabetically, see Help for default sort order]

* <SP>SOrt Plex DESTINATION OK

* <SP>SOrt Group DESTINATION OK

UPDATE FILE [not imperative, but good practice]

Update File OK

Update File Compact OK
[Note: this will ensure the efficient storage of a file that has been edited extensively. To find out the percent of efficiently used storage, use <SP>SHow File Status OK]

* Update File Old (version) OK

* Changing the modifications to an entire file:

* Delete Modifications (to file) OK (really?) OK

6. COMMUNICATING with other users

SENDMAIL SUBSYSTEM and the Journal

Goto (subsystem) Sendmail OK

Individual commands: instead of or in addition to Interrogate, you may use the following:

Title CONTENT OK

Distribute (for) Information (Only) (to)
IDENT/.LASTNAME OK
[You may give a series of IDENTS
or .LASTNAMES separated by commas]

Distribute (for) Action (to) IDENT/.LASTNAME OK

Comments CONTENT OK

* Authors CONTENT OK

To send a message or statement:

Message CONTENT OK

<SP>STatement (at) SOURCE OK

To send a structure or file:

<SP>Group (at) SOURCE OK

*<SP>PLex (at) SOURCE OK

Branch (at) SOURCE OK

File DESTINATION OK

<SP>SHow Status OK

Send (the mail) OK

To identify a user by lastname or ident:

<SP>SHow Record (for ident) .LASTNAME OK [precede by a period]

<SP>SHow Record (for ident) IDENT OK

* Dialog support application: design philosophy behind the word Journal

* Indexes to all public Journal items:
Jump (to) Link (userguides,locator,7:xbmg) OK

* Special commands:

  * <SP>PRivate OK [only those in the distribution can read it]

  * <SP>PUblic OK

  * <SP>INSert Status (form to follow) DESTINATION LEVEL-ADJUST OK

  * Process (sendmail form at) DESTINATION OK

  * <SP>INItialize (specifications) OK

* To leave the Sendmail subsystem to edit and then return without losing anything:
  Goto (subsystem) Base OK
  Quit OK [returns you to Sendmail]

SEND MESSAGE (Tenex)

Goto (subsystem) Tenex OK

```
SND CR      [The system will prompt you:]
(To (? for help):) TYPEIN CR  [lastnames separated by comma]
(cc (? for help):) TYPEIN CR  [lastnames separated by comma]
(subject:) TYPEIN CR     [subject of your message]
(message:) TYPEIN
CRTL Z      [to terminate the message]
(Q, S, ?, carriage return:) CR  [to send the message]
QU CR       [to return to NLS]
```

To abort a send message before sending it type a CTRL C

* Net distribution: the careful use of @ [you need to type 2 @'s if @ is the Net attention character; you will see 3 @'s]

* To submit a sequential file: After (message:) type: CTRL B then F and the complete TXT filename

7. TROUBLE SHOOTING AND HELP

Immediate assistance from the system:

Type ? for commands or needed information after any prompt.

HELP:

Type CTRL Q for help concerning what you are doing or type H for Help command (after typing H you can type any word in NLS you wish to know about). CTRL X gets you out of Help and back to where you were.

Help TYPEIN OK

Help OK

System Status:

Two CTRL T's [Note the words RUNNING or WAIT -- WAIT means the computer is waiting for you to do something]

<SP>SHow <SP>Disk (space status) OK [each user has a certain allocation of pages]

*<SP>SHow File Status OK
*Verify File OK

Send a message or sendmail item to: FEEDBACK

Call SRI/ARC, (415 326-6200, ext.3630)

Link to FEEDBACK

Remedies:

CTRL C [use only in emergencies to get to TENEX]
RESET CR
NLS CR

If over allocation:
<SP>EXpunge Directory OK
<SP>Trim Directory (no. of versions to keep) CONTENT OK (really?) OK

Update File Compact OK [restores file more efficiently in computer]
Delete Modifications OK (really?) OK [destroys all changes since the last update!]

* 8. OUTPUT FOR PRINTING

* Output processor directives

* The directives that must be inserted in a file to permit output processing can be inserted automatically by a program that attaches the subsystem Format (see Section 9.)

* The following directives are the most commonly used.
[precede with . and follow with ;]

* Pagination:

  * PBS paginate before statement

  * PES paginate at end of statement

  * Grab=n paginate if can't fit n lines on current page

  * PN=n set page number to n

* Headers:

  * H1=" " set header 1 (top of every page)

* Special text:

  * GD generate text for current date

* Vertical Spacing:

  * YBS=n blank distance (lines) between statements

  * YBL=n blank distance (lines) between lines

  * GCR generate carriage return

* Indenting:

  * Center=n center n lines

  * IFirst indentation of first line of every statement

* Numbering:
  * SN=On/Off left statement numbers on/off
  * SNF=n statement numbers right justified to the nth character position
* Stop printing:
  * Halt stop printing at this point; as if file ended here
  * Post=On/Off Post=Off postpones printing until it sees Post=On

* Output to a printer

  * Output (to) Remote (printer -- TIP) CONTENT (Port #) CONTENT OK (Send Form Feeds?) N (Simulate?) ANSWER (Wait at page break?) ANSWER (Go?) N (Type CA when ready, CD to abort) OK
    [If you have access to a printer wired to a TIP]

  * Output (to) Remote (printer -- TIP) CONTENT (Port #) CONTENT OK (Send Form Feeds?) Y (Wait at page break?) ANSWER (Go?) N (Type CA when ready, CD to abort) OK

* Output to a typewriter terminal

  * Output (to) Terminal OK (Send Form Feeds?) N (Simulate?) ANSWER (Wait at page break?) ANSWER (Go?) ANSWER [type N to allow time to position paper, type OK when ready]

    * [simulate sends line feeds instead of a formfeed]

  * Output (to) Terminal OK (Send Form Feeds?) Y (Wait at page break?) ANSWER (Go?) ANSWER

    * [your terminal must have a formfeed capability]

* Quick printing [use SENDPRINT or equivalent to send this file to your local printer]

  * Output (to) Quickprint File CONTENT OK

* Output to a sequential file [useful for sndmsg and Net transfer]

  * Output (to) Sequential File CONTENT OK

* COM: Computer Output to Microform, what it is -- for specialists only

* 9. PROGRAMS

  * Programs subsystem

    * Only Class I User Programs are covered in the Third course; most of these load as subsystems with NLS command words.

    * Goto (subsystem) Programs OK

    * Load Program CONTENT OK

    * Quit OK [to leave Programs]

    * Run Program CONTENT OK

  * User Programs subsystems -- see the Brief Guide to User Subsystems

*10. USEROPTIONS SUBSYSTEM
[Note: "or" is used here to indicate the alternative command words]

*Goto (subsystem) Useroptions OK

*Feedback Indenting [or] Length [or] Verbose [or] Terse
[indents, controls length, or turns noisewords on and off]

*<SP>Herald Length [or] Verbose [or] Terse

*<SP>Printoptions Right (margin is column) [or] Left (margin is column) [or] Bottom (margin is line) [or] Page (size is lines) [or] Indenting (per level) CONTENT OK:
[number of spaces for margins and level indenting]

*Name (delimiters defaults)(left delimiter) CONTENT (right) CONTENT OK:
[for new files]

*Show [type in the first letter of the commandword for any one of the above] OK:

PRACTICE

Use Strategies
If there is time available while your trainer is present, ask her to explain how to use the system to accomplish specific tasks, from daily routine tasks (such as message handling) to online compostion with multiple authors.

OTHER AVAILABLE COURSES:

INTRODUCTION TO DNLS
This is an introduction to the display version of NLS designed to follow the second TNLS course (it requires an understanding of structure and viewing). It covers the use of the special workstation required for DNLS as well as the special ways of pointing and displaying information that are available.

* DEX (Deferred execution) [Special Course, see the DEX Userguides]

EXAMPLE OF STRUCTURE:

```
< BAIR, MENU.NLS;1, >, 28-JAN-75 17:29 JHB ;;;;
1 SOUP
   1A VEGETABLE
   1B CREAM OF MUSHROOM
2 ENTREE
   2A  FRIED CHICKEN
   2B  SALMON
      2B1 WITH CREAM SAUCE
   2C  PRIME RIBS
3 DESSERT
   3A  PIE
      3A1  APPLE
         3A1A A LA MODE
      3A2  BLUEBERRY
   3B  ICE CREAM
      3B1  VANILLA
      3B2  PEPPERMINT
      3B3  MAPLENUT
      3B4  CHOCOLATE
4  BEVERAGE
   4A  TEA
   4B  COFFEE
```