MDK 29-APR-74 15:31 22855

note from Dave Crocker

Jim ... this note from DHC has a suggestion (in third paragraph) that you might be interested in ... Mike

note from Dave Crocker

24-APR-74 0957-PDT DCROCKER at USC-ISI: Estrin ident
Distribution: KUDLICK, keeney at SRI-ARC
Received at: 24-APR-74 09:58:08 1

Marcia -- Estrin doesn't have any other network address. The problem with the wording of "NO NETWORK ADDRESS" is that a relatively naive reader is inclined to think that the person has no Network Address (since that is what the statement says); whereas the reality is that the Journal consititutes a Net Mailbox. 1a

I don't have any strong suggestions about wording, "NO OTHER NET ADDRESS" is the best I can think of, offhand (and it isn't much longer than the current phrase). 1b

By the way, Mike, a side thought (which should be copied to Jim White, but I've already specified the recipient list for this note): Your FTP Server which currently looks for a slash (/) to decide whether it has a Journal or regular Tenex address (for mail) should also assume Journal ident, if the name is not a Tenex directory. 1c

E.g.: 1) Look for slash; if not there 2) try to match string with Tenex directory; if unsuccessful; try to match with ident; if successful, delivery thru Journal as "anonymous" (no author indicated) mail. 1d

That way I can receive Journal mail sent to DHC@OFFICE. More importantly, people without any other Net address can receive the mail. (That is, ist can be sent to them in a natural way). 1e

Dave. 1f

MDK 29-APR-74 15:31 22855

note from Dave Crocker

(J22855) 29-APR-74 15:31; Title: Author(s): Michael D. Kudlick/MDK;
Distribution: /JEW; Sub-Collections: SRI-ARC; Clerk: MDK;

Draft of Basic INLS Course (NSW)

This is a draft of the course to be used in Wash, to introduce NLS with the primary goal of simplicity and ease of learning. The Wash DC people are conected with the ARPA/NSW project (S. Crocker). It considered experimental and all feedback is welcome.

TNLS SYLABUS

BASIC TNLS COURSE

SRI-ARC

30 APR 74

Augmentation Research Center

STANFORD RESEARCH INSTITUTE
MENLO PARK, CALIFORNIA 94025

THE BASIC TNLS COURSE

INTRODUCTION TO AKW

AKW = Augmented Knowledge Workshop

PURPOSE OF SYSTEM: Augmentation of Knowledge Work

TNLS = Teletype oN Line System

Course Goals and Philosophy

THE TERMINAL AND USE

Similarities to and differences from a typewriter

COURSE ORGANIZATION

The course is organized by concepts of what a user can do with TNLS at this level. There are seven concepts (as listed below) that 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 text. Those commands numbered with a (2) are to be covered on the second day of the course.

GETTING TO NLS

NETWORK (if used)

TENEX Executive

TNLS CONCEPTS:

1. FILES
2. TYPING IN TEXT
3. PRINTING
4. ADDRESSING

5. EDITING

6. COMMUNICATING

7. TROUBLE SHOOTING AND HELP

GETTING TO NLS (For those familiar with using the Net and Tenex) .

NETWORK (if used)

Login protocols (Office-1 is host numer 43)

e, log and close, device code extrapadding

TENEX Executive (review)

Login procedure

log USERNAME PASSWORD ACCOUNT #

Group allocation quota

Directory listing

dir

Executive commands

delete

logout

Calling NLS

Type NLS, then (after the asterisk) type: v c m E CR

control c and continue

TNLS:

Abort Commands = control x

1. FILES

The origin statement

The initials file

New files

n[ull file F:] FILENAME CR

2. TYPING IN "TEXT"

Insert Statement

i[nsert] s[tatement at A:] ADDRESS CR CR

(2) Continue to insert = control b

backspace character = control a
backspace word = control w

3. PRINTING

Print and stop

p[rint] CR

Stop printing = control o

Print Statement (ADDRESS = statement number)

p[rint] s[tatement A:] ADDRESS CR CR

(2) Skip statement (control s)

4. ADDRESSING

Addressing within files

Statement numbers preceded by a period

SPACE command address

SPACE [A:] ADDRESS CR

(2) Content string: [string]

Addressing across files and directories

load file

l[oad] f[ile F:] FILENAME CR

(2) Link: (Fileowner,filename,statement number) OR
(filename,statement number)

5. EDITING

Commands:

Delete Statement

d[elete] s[tatement at A:] CR [OK?] CR

Substitute Text in Statement

s[ubstitute] t[ext in] s[tatement A:] ADDRESS CR
[<new text> T:] TYPEIN CR
[<for old text> T:] TYPEIN CR
[finished?] CR (yes)
[substitutions made= #]

Update

(2) Move Statement

m[ove] s[tatement] [to follow A:] ADDRESS CR [from A:]
ADDRESS CR CR

(2) formatting technique

insert carriage return = control v CR

u[pdate] CR [filename]

6. COMMUNICATING

(2) JOURNAL SYSTEM:

(2) Submit message or statement or file, idents (or ,name), and Interrogate

E[xecute Journal]

&S[ubmit] M[essage T:] TYPEIN CR

&&I[nterrogate] CR

[&&Title: T:] Example

[&&Distribution: I:] rww I: jcn I: dvn CR

[&&Status] CR (the following is the status typed by the system:)

[Catalog Number: Deferred

Author(s): JHB

Title: Example

Distribution: RWW JCN DVN

Subcollections: sri-arc

Clerk: JHB;

&&Go?] CR [Yes]

[Journal System in progress]
[Completed]

(2) Initials file = mail box

(2) Print Journal and empty mail box

p[rint] j[ournal] CR

TENEX ways: (review)

SNDMSG

Link (to) [username]; break links

7. TROUBLE SHOOTING AND HELP

(2) FEEDBACK mechanism:

SNDMSG to FEEDBACK or send a Journal item to ident FEED

HELP:

call or Link to (Bair (Jim) at SRI/ARC, (415 326-6200, ext,3614))

(2) Status commands

control t

(2) Remedies

control c, reset, NLS

Output file

PRACTICE

In addition to trying each command, there is a Primer designed to be used for practice.

TNLS COMMAND SUMMARY FOR THIS COURSE: (alphabetical) NLS supplies that which appears between brackets, CR = Carriage Retrun.

Backspace character = control a ; backspace word = control w

Carriage (formatting) return = control v CR

Continue to insert = control

Delete Statement:

d[elete] s[tatement at A:] CR [OK?] CR

Insert Statement:,

i[nsert] s[tatement at A:] ADDRESS CR CR

Link:

(Fileowner,filename,statement number) or (filename,statement number)

Load file:

l[oad] f[ile F:] FILENAME CR

Move Statement:

m[ove] s[tatement] [to follow A:] ADDRESS CR [from A:] ADDRESS CR CR

Null file:

n[ull file F:] FILENAME CR

Output file:

o[utput] f[ile F:] FILENAME CR

Print Statement:

p[rint] s[tatement A:] ADDRESS CR CR

Print:

p[rint] CR

Stop printing = control o

Skip to next statement = control s

SPACE [A:] ADDRESS CR

Substitute Text in Statement:

s[ubstitute] t[ext in] s[tatement A:] ADDRESS CR
[<new text> T:] TYPEIN CR
[<for old text> T:] TYPEIN CR
[finished?] CR [yes]
[substitutions made= #]

Update a file:

u[pdate] CR [filename]

JOURNAL SYSTEM:

Submit Message or Statement or file, idents (or ,name), and Interrogate

e[xecute] j[ournal CR]

s[ubmit] f[ile] at A: FILENAME CR OR

f[ile] CR OR
m[essage T:] TYPEIN CR

s[ubmit] s[tatement] at A: ADDRESS CR

f[ile] CR OR
m[essage T:] TYPEIN CR

EXAMPLE: (NLS supplies the prompts * and & and && and everything in brackets)

*E[xecute Journal]

&S[ubmit] S[tatement at A:] ADDRESS CR

&&I[nterrogate] CR

[&&Title: T:] Example

[&&Distribution: I:] rww I: jcn I: dvn CR

[&&Status] CR (the following is the status typed by the system:)

[Catalog Number: Deferred

Author(s): JHB

Title: Example

Distribution: RWW JCN DVN

Subcollections: sri-arc

Clerk: JHB;

&&Go?] N[o] CR (if you change your mind)

&&Q[uit] CR

Journal subcommands:

t[itle:] TYPEIN CR

d[istribution:] IDENT SPACE IDENT SPACE IDENT... CR

co[mments:] TYPEIN CR

g[o?] CR [yes]
(OR) n[o]

i[nterrogate] CR

st[atus] CR

g[o?] CR

(2) Print Journal and empty mail box

p[rint] j[ournal] CR

Draft of Basic TNLS Course (NSW)

(J22856) 29-APR-74 22:57; Title: Author(s): James H. Bair/JHB;
Distribution: /DLS CKM RWW CHI DVN DCE JCN MDK; Sub-Collections:
SRI-ARC; Clerk: JHB;
Origin: <BAIR>COURSENSW.NLS;8, 29-APR-74 22:47 JHB ;

BAH 30-APR-74 11:11 22857

NEW PSO GROUP ALLOCATION SCHEME

The following is a copy of the new time allocation scheme for PSO, the one agreed upon by the group in the central office last week, If it still meets the disfavor of some, I suggest they begin negotiations first with the party involved in the conflict, This new scheme takes effect May 1, 1974, Please abide with it until something better comes along, -bah

NEW PSO GROUP ALLOCATION SCHEME

MON 1

	8	9	10	11	12	1	2	3	4	5	TOT	
	-	-	--	--	--	-	-	-	-	-	---	1b
KIR											0	1c
BAH				x	x				x	x	4	1d
JML		x	x	x		x					4	1e
SLJ		x	x				x	x			4	1f
MEJ					x	x	x	x	x	x	6	1g
											--	1h
											18	1i

TUE 2

	8	9	10	11	12	1	2	3	4	5	TOT	
	-	-	--	--	--	-	-	-	-	-	---	2b
KIR		x	x	x	x	x					5	2c
BAH					x				x	x	3	2d
JML		x	x	x							3	2e
SLJ							x	x	x	x	4	2f
MEJ						x	x	x			3	2g
											--	2h
											18	2i

WED 3

	8	9	10	11	12	1	2	3	4	5	TOT	
	-	-	--	--	--	-	-	-	-	-	---	3b
KIR		x	x	x	x						4	3c
BAH				x	x				x	x	4	3d

NEW PSO GROUP ALLOCATION SCHEME

JML	x	x			x						3	3e
SLJ						x	x	x	x		4	3f
MEJ					x	x	x				3	3g
											--	3h
											18	3i

THU 4

	8	9	10	11	12	1	2	3	4	5	TOT	
	-	-	--	--	--	-	-	-	-	-	---	4b
KIR	x	x	x	x							4	4c
BAH			x	x				x	x		4	4d
JML	x	x			x						3	4e
SLJ						x	x	x	x		4	4f
MEJ					x	x	x				3	4g
											--	4h
											18	4i

FRI 5

	8	9	10	11	12	1	2	3	4	5	TOT	
	-	-	--	--	--	-	-	-	-	-	---	5b
KIR	x	x	x	x	x						5	5c
BAH				x				x	x		3	5d
JML	x	x	x					x	x		5	5e
SLJ						x	x				2	5f
MEJ					x	x	x				3	5g
											--	5h
											18	5i

NEW PSO GROUP ALLOCATION SCHEME

BY DAY 6

	MON	TUE	WED	THU	FRI	TOT	
	---	---	---	---	---	---	6b
KIR	0	5	4	4	5	18	6c
BAH	4	3	4	4	3	18	6d
JML	4	3	3	3	5	18	6e
SLJ	4	4	4	4	2	18	6f
MEJ	6	3	3	3	3	18	6g
	--	--	--	--	--	--	6h
	18	18	18	18	18	90	6i

NEW PSO GROUP ALLOCATION SCHEME

(J22857) 30-APR-74 11:11; Title: Author(s): Beauregard A. Hardeman/BAH; Distribution: /KIRK BAH JML SLJ MEJ; Sub-Collections: SRI-ARC; Clerk: BAH;

JHB 27-MAY-74 13:01 22858

The Basic TNLS Course

This basic course was designed with help from RWW, JCN, MDK, CHI, DVN, and CKM, and revised as a result of the courses given to NSW and Seismic people May 1 - 8. It is available in hardcopy for those who have need (all courses should be coordinated through User Development, however).

JHB 27-MAY-74 13:01 22858

BASIC TNLS COURSE

SRI-ARC

27 MAY 74

Augmentation Research Center

STANFORD RESEARCH INSTITUTE
MENLO PARK, CALIFORNIA 94025

TNLS SYLABUS

THE BASIC TNLS COURSE

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

COURSE GOALS

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 commands numbered with a (2) are to be covered on the second day of the course.

GETTING TO NLS

TERMINAL

NETWORK (if used)

TENEX Executive

TNLS CONCEPTS:

1. FILES

2. TYPING IN TEXT

3. TYPING OUT TEXT

4. ADDRESSING

5. EDITING

6. COMMUNICATING

7. TROUBLE SHOOTING AND HELP

DEFINITIONS FOR THE COURSE OUTLINE

control = hold down the control (ctrl) key while typing the specified character.

[ ] = brackets which enclose what the computer types out for you.

* = the TNLS ready signal. It means that you can type in a command.

GETTING TO NLS (review)

THE TERMINAL AND USE (if necessary)

Similarities to and differences from a typewriter

NETWORK (if used)

Net login, after establishing a phone connection type:

e (<> equals a space)

@<>d<>c<>e CR (Not necessary for all terminals)

@<>1<>43 CR (Office-1 is host number 43)

TENEX Executive (review)

Login procedure:

log USERNAME PASSWORD ACCOUNT CR

Group allocation quota: gro<esc>UPSTAT

Directory listing:

dir CR

Some executive commands:

delete

logout

Calling NLS:

Type NLS, then (after the asterisk) type: vcmyE CR

To return to the Exec:

control c

To continue where you were in TNLS:

continue CR

TNLS BASIC COURSE OUTLINE:

Abort Commands = control x

1, FILES

The origin statement

The initials file

New files

n[ull file F:] FILENAME CR

2, TYPING IN "TEXT"

Insert Statement (ADDRESS = ,statement number)

i[nsert] s[tatement at A:] ADDRESS CR [T:] = TYPEIN CR

Continue to insert = escape key

backspace character = control a
backspace word = control w

Insert Text at the end of a statement

(2) i[nsert] t[ext at A:] > CR [T:] TyPEIN CR

3, TYPING OUT TEXT

Printing

p[rint] CR

Stop printing = control o

(2) Skip statement = control s

Print Statement

p[rint] s[tatement A:] ADDRESS CR CR

(2) Easy print = \

4. ADDRESSING

Addressing within files

Statement numbers preceded by a period (NOTE: TNLS automatically renumbers statements when appropriate)

t ("tail") for the last statement in the file

(2) Content string: [string]

SPACE command address

SPACE [A:] ADDRESS CR

Addressing across files and directories

load file

l[oad] f[ile F:] FILENAME CR

(2) Link: (Fileowner,filename,statement number) OR (filename,statement number)

5. EDITING

To change text that has been typed in:

Delete Statement

d[elete] s[tatement at A:] CR [OK?] CR

Substitute Text in Statement

s[ubstitute] t[ext in] s[tatement A:] ADDRESS CR
[<new text> T:] TYPEIN (no more than 80 characters) CR
[<for old text> T:] TYPEIN CR
[finished?] CR [yes]
[substitutions made= #]

Update:

u[pdate] CR [filename]

(2) Move Statement:

m[ove] s[tatement to follow A:] ADDRESS CR [from A:]
ADDRESS CR CR

(2) Copy Statement:

C[opy] S[tatement to follow A:] ADDRESS CR [from A:]
ADDRESS CR CR

(2) formatting technique:

insert carriage return = control v CR

6. COMMUNICATING

(2) JOURNAL SYSTEM:

(2) Submit message using idents (or ,receivername) and Interrogate (where the system prompts you):

E[xecute] J[ournal]

&S[ubmit] M[essage T:] TYPEIN CR

&&I[nterrogate] CR

[&&Title: T:] Example

[&&Distribution: I:] rww <>[I:] jcn <>[I:] <>dvn CR

[&&Status] CR (the following is the status typed by the system:)

[Catalog Number: Deferred

Author(s): JHB (your ident)

Title: Example

Distribution: RWW JCN DVN

Subcollections: sri-arc

Clerk: JHB; (your ident)

&&Go?] CR [Yes]

[Journal System in progress]
[Completed]

(2) Submit statement (See the Command Summary for an example)

(2) Submit file (See the Command Summary)

(2) Initials file = mail box

(2) Print Journal

p[rint] j[ournal] CR

(2) Empty mail box: subsitute (read) for (journal)...

TENEX ways: (review)

SNDMSG

Link (to) [username]; break links

7. TROUBLE SHOOTING AND HELP

(2) FEEDBACK mechanism:

SNDMSG to FEEDBACK or send a Journal item to ident FEED

HELP:

call or Link to (Bair (Jim) at SRI/ARC, (415 326-6200, ext,3614))

(2) Status commands

control t

(2) Remedies

control c, reset, NLS

Output file

PRACTICE

In addition to trying each command, there is a Primer designed to be used for practice.

TNLS COMMAND SUMMARY FOR THIS COURSE: (alphabetical) NLS supplies that which appears between brackets. CR = Carriage Retrun.

BACKSPACE CHARACTER = control a ; BACKSPACE WORD = control w

CARRIAGE RETURN (formatting) = control v CR

CONTINUE TO INSERT = escape key (esc)

COPY STATEMENT

C[opy] S[tatement to follow A:] ADDRESS CR [from A:] ADDRESS CR CR

DELETE STATEMENT:

d[elete] s[tatement at A:] CR [OK?] CR

INSERT STATEMENT:,

i[nsert] s[tatement at A:] ADDRESS CR CR

INSERT TEXT at the end of a statement

(2) i[nsert] t[ext at A:] > CR [T:] TYPEIN CR

LINK:

(Fileowner,filename,statement number) or (filename,statement number)

LOAD FILE:

l[oad] f[ile F:] FILENAME CR

MOVE STATEMENT:

m[ove] s[tatement to follow A:] ADDRESS CR [from A:] ADDRESS CR CR

NULL FILE:

n[ull file F:] FILENAME CR

OUTPUT FILE:

o[utput] f[ile F:] FILENAME CR

PRINT STATEMENT:

p[rint] s[tatement A:] ADDRESS CR CR

Easy print = \

PRINT:

p[rint] CR

Stop printing = control o

Skip to next statement = control s

SPACE [A:] ADDRESS CR

SUBSTITUTE TEXT IN STATEMENT:

s[ubstitute] t[ext in] s[tatement A:] ADDRESS CR
[<new text> T:] TYPEIN CR
[<for old text> T:] TYPEIN CR
[finished?] CR [yes]
[substitutions made= #]

TAIL = t for ADDRESS

(the last statement in the file -- when single level)

UPDATE A FILE:

u[pdate] CR [filename]

JOURNAL SYSTEM:

Submit Message or Statement or File, idents (or ,receivername), and Interrogate:

e[xecute] j[ournal]

s[ubmit] f[ile at A:] FILENAME CR or
s[ubmit] s[tatement at A:] ADDRESS CR or
s[ubmit] m[essage T:] TYPEIN CR

EXAMPLE: (NLS supplies the prompts * and & and && and everything in brackets)

*E[xecute Journal]

&S[ubmit] S[tatement at A:] ADDRESS CR

&&I[nterrogate] CR

[&&Title: T:] Example

[&&Distribution I:] rww <>[I:] jcn <>[I:] dvn CR

[&&Status] CR (the following is the status typed by the system:)

[Catalog Number: Deferred

Author(s): JHB

Title: Example

Distribution: RWW JCN DVN

Subcollections: sri-arc

Clerk: JHB;

&&Go?] N[o] CR (if you change your mind)

&&Q[uit] CR

Journal subcommands:

t[itle:] TYPEIN CR

d[istribution I:]IDENT SPACE [I:]IDENT SPACE [I:]IDENT... CR

c[omments:] TYPEIN CR

g[o?] CR [yes]
(OR) n[o]

i[nterrogate] CR

st[atus] CR

g[o?] CR

(2) Print Journal

p[rint] j[ournal] CR

(2) Empty mail box: subsitute (read) for (journal)...

Development Analysis Tasks Between Now and July 1

Development-Analysis Tasks Desirable to Accomplish Before July 1, All numbers estimated man months. 1

New Command Language (EKM, KEV, DSM) (3-4) 1a

Bug fixes and Misc to get all subsystems and appropriate user programs running. 1a1

Help system speedup and improvements. (HGL) 1a2

After at least one month usage experience analyze comments and experience and suggest changes to language needed. Many people likely to be involved here for review. SRL and above team should put together recommendations. 1a3

Design Multi Host Journal System. (JEW) Will need review and discussion. (1) 1b

Design NLS Frontend System (CHI, with help from DIA and JEW) Will need review and discussion. (1) 1c

CML Interpreter 1c1

Execution Module Changes 1c2

CML to Execution Module protocol 1c3

Get PDP 11 in House (KEV, DIA) (.5) 1d

Acceptance testing 1d1

Operating system running 1d2

Recommend Display system for next year. (DIA, CHI) presently set on Hazeltine and Line Processors, will go with these unless something much better shows up at NCC. If it is recommended to go with something like the Megadata then what development effort required? (.25) 1e

Complete stage 0 FORMS system. (EKM) (1) 1f

MST Preliminary Planning (RWW) (1) 1g

Recommended hardware and support software. (DIA) Position paper needed here, main decision seems to be what display systems to recommend for large MITS and what configuration to recommend for small MITS. Need a couple intro sessions on ELF and BCPL. 1g1

Development Analysis Tasks Between Now and July 1

Items below should be initially looked at by everybody independently thinking and with some brainstorming sessions. Analysis should be thinking about stimulating some discussions and collecting ideas. 1g2

Work on methodology for how to determine what changes are needed in user interface, particularly for secretaries (MDK has some thoughts on command packaging). 1g2a

Generate ideas for new ways to represent and browse in information space. IS NLS the ultimate or are there other ideas that should be considered? 1g2b

Generate ideas for what to do for scheduling and task management. 1g2c

NLS Measusrement system, just data collection hooks. Data reduction programs latter. PR has a recommended set of data types to be collected. (KM) (1) 1h

Line Processor Developmet Finished (DIA) (.5) 1i

Recruiting (everybody) (.5) 1j

Training for new people. Big brother sister scheme. (.25) 1k

(DSM EKM) (JEW KM) 1k1

ARPA Final Report (RWW, HGL and others) (6) 1l

Plan for getting knowledge of system more adequately documeted or at least spread. NLS knowledge needs to be more deeply widespread. Better system overview documetation needed with guides to load map listings etc. Need some seminars and bull sessions here starting after NCC. (JEW, CHI organize whats needed) (1) 1m

Finish Present Journal study. (PR, SRL) (.5) 1n

Get FEEDBACK Process running smoothly (PR, SRL) (.5) 1o

Development Analysis Tasks Between Now and July 1

(J22859) 30-APR-74 12:42; Title: Author(s): Richard W. Watson/RWW;
Distribution: /SRI-ARC; Sub-Collections: SRI-ARC SRI-ARC; Clerk: RWW;

Information Retrieval - Specific and Philosophical

Jim and Dean -

1

Dean and I had a long talk over the phone Friday evening and these are some of the things we discussed:

2

(1) Our cataloging conventions cover almost everything the "client" desires, except for essentially two items: First, the client needs to identify the Principal Investigator of the project which produced the document being cataloged; and second, the client is working with a multiple "set" of keywords, which I gather (it seems a little vague here...possibly the client has not said, or some of the facets of this work are "private" in some manner and are not to be discussed) are derived from a source external to the client, to which they must conform, and when used, the source must be identifiable.

2a

(2) Dean is able and willing to make some (apparently fairly extensive) changes to the catalog programs.

2b

(3) The database being input probably will not be more than 50 documents immediately. Dean is going to wait until they have about 20 or so documents input, go over the database carefully, then run trial runs on the programs. This will give all of you some idea of (a) the extent and peculiarities of the database; (b) the viewpoint the client uses in cataloging; (c) the needs of the client, which will become more apparent as some documents are input and he sees what his needs are. From the processed, and formatted input, Dean can then have a clearer idea of how much program re-writing will actually be required.

2c

(4) The following detailed discussion of codes resolved some problems foreseen in the light of less information:

2d

The use of #2 org *b2 [organizationname] when *f1 is "r" -- that is, for reports being catalogued -- was explained and is understood. Dean had stated this rule in his "set of rules" (,22814,), but had not applied the rule in his illustration; this point is now clear and he understands what the cat progs will expect in this case.

2d1

The statement name consisting of paren three alphas and a number paren was discussed. Dean understands about the sort

programs in the cat-progs and will substitute on the database a paren one alpha character number paren for the present client naming convention, This will allow the client to retain his distinctive numbering system and will only occasionally give problems...that is, in an entry where a number used by the client is cited, then the substitution will also work on it....unless Dean creates a small program which is run on the database and changes only statement names, This of course, is a distinct possibility, but must be remembered and done before starting the cat-progs processing, However, it will not be a large task; simply must be listed in the check list of things to be accomplished during processing,

2d2

It was pointed out that the use of *f2 ?? to indicate what manner of holding the cataloged item consisted would be helpful to the client, This will give more flexibility, allowing many different kinds of things to be entered successfully in the database and enabling them to be found more easily,

2d3

It was also pointed out that good and practical use could be made of *z2 as a subcollection or "separating off of a particular category of material" field, There are already existing programs to pull out of the database items of like coding in *z2 field, allowing for catalogs to be made on subcollections, printouts of only certain items, etc,

2d4

The choice of *y4 with subnumbers of #1, #2, #3, etc, to cover the clients use of "multiple set" keywords was discussed, It was pointed out that the use of subnumbers under a *-level field required more programming than a simple use of something like *y2, *y3, *y4, *y5, *y6, *y7 would, (It is also worthy of reminding all, that *y8 and *y9 fields are already in use and the cat-progs actually use these fields and format in a particular way anything found in them,)

2d5

The use of *p1 for the Department name where the Principal Investigator works, with #1 under that for the name of the Principal Investigator, #3 for the phone number, #4 and #5 for the street address and city/state, was discussed without actually resolving the question or arriving at a meeting of the minds, Again, as in several of the other items, it seems to be a matter of definition,

2d6

Here I questioned to Dean, and still question, the wisdom of

using *p1 itself for the company's department name, while putting the name of the investigator as #1 under that field. Dean was too tired when we talked to continue the discussion further, and I would like to explore this thinking with him. Inevitably, I am convinced, he will run into trouble on this particular item.

2d6a

Dean stated that in his example he placed the coding for this item as:

2d6b

*p1 Augmentation Research Center #1 Douglas C. Engelbart #3 (415)326-6200, ext. 2220 #4 333 Ravenswood Avenue #5 Menlo Park, California 94025

2d6b1

because *p1 is the field for "project name", and "Augmentation Research Center" is the project name.

2d6c

What I tried to explain and never got a chance to finish, is that this is not true; Augmentation Research Center is not the project name. It is the department name of where the Principal Investigator works. Our group has had many project names, and usually works on more than one at a time. In most cases, the project name would be the name of the contract the group is working on. In certain cases on large, ongoing contracts of long duration, the project contract is given an overall name and each new revision or renewal of contract is given a particular sub-name. Examples are Project MAC, The Cambridge Project, MIT-Mathlab Project, RAND's VENUS Project. In our particular case any overall project name was usually spoken of as Doug Engelbart's NLS project, without giving it a formal name, although in the last couple of years Doug has settled on "the Knowledge Workshop" as his project name.

2d6d

For this particular aspect, I would suggest that *p1 be used as the manual indicates for the project name, such as Project MAC, or VENUS Project, or NSW Project (and *p1 be left blank if no project name given); *p2 be used as the manual indicates for a project number such as SRI Project 1868; *p3 be used for the name and address of the Principal Investigator in the following fashion:

2d6e

*p3 Douglas C. Engelbart #1 Director #2 Stanford Research

Institute #3 Augmentation Research Center #4 333
Ravenswood Avenue #5 Menlo Park, California 90425 #6
(415)326-6200, ext. 2220,

2d6e1

When there is more than one Principal Investigator LISTED ON THE DOCUMENT BEING CODED (example: Jacques Vallee and Roy Amara, Co-Principal Investigators on the FORUM Project), then it is logical to use *p4 with subnumbers in exactly the same manner as in *p3, for the second, or Co-, Principal Investigator. It will be wise therefore, in writing the additional module for the cat-progs to have the program pick up *p3 and, if there is a *p4, have the program pick up that also, listing them both as Co-PI's instead of as PI, as would be the case if only *p3 exists in the entry.

2d6f

In this manner possibilities for multiple projects under one Principal Investigator, or multiple Principal Investigators (possible even at multiple sites) working on one project may be easily handled in a more logical fashion. Existing coding conventions and any hooks into the programs will not be disturbed, and pulling out both the project name and the Principal Investigator(s) from the database with my special programs will be a simple matter of substitution in my programs of 3 characters at two places each.

2d6g

I hope this point is clear: (1) It is perfectly logical to have more than one Principal Investigator at more than one site working on a particular named project (all shown on one document); (2) by maintaining generically similar information at the same coding level, handling of the database is more logical instead of a crazy-quilt of patchwork; (3) it is easier to INSERT small modules of code into the programs, than to CHANGE existing interwoven functions of the program. (Experience has shown that an apparently simple, innocuous, small change will often throw off completely a later formatting step, making for some strange output from these programs.)

2d6h

A further word about item (2) immediately above: By stating that generically similar information is coded at the same coding level, I mean this: #1 under a *-level field is reserved for "title"...even under *c1, its meaning is a subtitle. #2 is designed for the organizational identification; #3 after that for the identifying department of the #2; #4 and #5 are the street address and

city/state/country, while #6 is the one which so far has somewhat varied meanings within the category of an identifying number. #6 under *c1 is page numbers, under *s1 is contract number. As suggested here, under *p3 and *p4 it would be a telephone number.

2d6i

One of the things Dean and I did not discuss is the fact there is much of the coded material in the database that the present catalog programs do not in any way touch...no formatting is done for them at all. This is not accident; material is coded for which at present we have had no catalog formatting effort underway. There is no reason why this should not be true of any information retrieval system. It is far wiser to attempt to code all information likely to be of interest to the client at first, than attempt the soul-shaking and budget shattering task of going back over the database, attempting to find the documents, and input the same material again -- that does not make sense. A small sampling for the database may be input on a trial basis, but from that sampling and a knowledgeable analysis of the management and research information needs of the CLIENT HIMSELF, an attempt should be made to do a complete coding job in one pass through.

3

During the first several months it will be inevitable that something will be thought of as needed that was not initially considered. That too, is one very large reason for making the coding conventions as modular and patterned as possible -- subsequent database insertions can then be done in a logical pattern and the result will be less likely to be a hodge podge of patches.

3a

With a correctly designed database, subsequent needs for catalog formatting of various kinds can be easily and flexibly overlaid over the original formatting programs and over the database itself.

3b

I hope all this is of some assistance in both structuring the particular needs of this particular client and of expressing some of the practical philosophy of information retrieval. Dean is doing a fine job and I hope that in no way either JCN or Dean will take anything I have expressed here or elsewhere in a negative manner. This is a highly complicated setup at ARC and carries many traps for the unwary coder and program-changer. Unfortunately we are saddled with a set of catalog programs which very effectively achieve the environment Djikstra warned against when he said that ponderous coding produced programs that tended to rule the human environment

with an iron hand,,they worked,,,but were so complicated and forbidding no one dare change them,,,they wind up as electronic dictators ruling their human operators,

4

Information Retrieval - Specific and Philosophical

(J22860) 30-APR-74 14:33; Title: Author(s): Mil E. Jernigan/MEJ;
Distribution: /JCN DCE NDM RWW DVN JMB; Keywords: Catalog Information
Retrieval IR; Sub-Collections: NIC SRI-ARC ; Clerk: MEJ;
Origin: <JERNIGAN>CATBIB.NLS;1, 30-APR-74 14:27 MEJ ;

this is the reply I received from Craig Fields re my draft.
Thought you'd be interested in seeing it, ... Mike

30-APR-74 1426-PDT FIELDS at USC-ISI: NIC SERVICES ANNOUNCEMENT
Distribution: KUDLICK, NORTON at SRI-ARC
Received at: 30-APR-74 14:27:53 1

1. NO NIC PROPOSAL HAS BEEN SUBMITTED TO ARPA, AND HENCE NONE HAS BEEN ACCEPTED. THE ANNOUNCEMENT SHOULD NOT GO OUT BEFOREHAND. 1a

2. QUESTIONS SHOULD GO TO JIM ALONE, NOT TO ME. 1b

3. IF JIM IS IN CHARGE THAT SHOULD BE STATED, ALONG WITH A GENERAL STATEMENT OF STAFFING. 1c

4. I DON'T THINK YOU SHOULD LIST SERVICES DISCONTINUED, PEOPLE WILL COMPLAIN ABOUT THE DISAPPEARANCE OF THINGS THEY NEVER KNEW EXISTED. 1d

BEST 1e

CRAIG 1f

(J22861) 30-APR-74 14:54; Title: Author(s): Michael D. Kudlick/MDK;
Distribution: /RWW JAKE; Sub-Collections: SRI-ARC; Clerk: MDK;

KEV 30-APR-74 16:24 22862

A COMMAND META LNAGUAGE FOR AN NLS FRONTEND - basis for my talk at nbs/siggraph workshop on machine independent graphics

this was to be the basis for my talk, however, at the last minute my allotted time was halved. (printing instructions: quickprint 1st branch; output processor 2nd branch (slides).)

A COMMAND META LNAGUAGE FOR AN NLS FRONTEND - basis for my talk at nbs/siggraph workshop on machine independent graphics

PAPER 1

** slide 1 ** A COMMAND META LANGUAGE FOR AN NLS FRONTEND 1a

** slide 2 ** INTRODUCTION 1b

In the first part of this discussion, I will explain ARC's goals, motivations, and plans for splitting NLS into two parts: 1b1

A frontend to interact with the user, and 1b1a

A backend to carry out the commands specified by the user in the frontend, 1b1b

In the second part of this discussion, I will go into detail about our concept of a frontend system which centers around a Control Meta Language for the specification of user interactions, 1b2

In my concluding remarks, I will go into where we are currently with repsect to accomplishing the ideas talked about, 1b3

1b4

TERMINOLOGY 1c

Before getting started however, I would like to define the terminology I will be using, 1c1

** slide 3 ** NLS 1c2

Over the past 10 years at the Augmentation Research Center (ARC) of SRI, we have been developing a computer and communications system, called NLS, to enhance the intellectual effectiveness of people by enhancing their ability to write, study and publish documents, correspondences, and notes; file and retrieve material; plan, organize and coordinate activities; and communicate with others through various media. NLS is a highly interactive system designed around well human-engineered display-based workstations, 1c2a

** slide 4 ** FRONTEND SYSTEM 1c3

A frontend system is a LOGICAL configuration of terminals, processing capability, and programs through which a user has access to, and interacts with, various subsystems, 1c3a

A COMMAND META LNAGUAGE FOR AN NLS FRONTEND - basis for my talk at nbs/siggraph workshop on machine independent graphics

FRONTEND PROCESS 1c4

A frontend process is a program that is running as part of a frontend system. 1c4a

BACKEND SYSTEM 1c5

A backend system is a LOGICAL configuration of processing capability, and programs which perform functions specified by the user during her interactions with the frontend system. 1c5a

BACKEND PROCESS 1c6

A backend process is a program that is running as part of a backend system. 1c6a

** slide 5 ** WORKSTATION 1c7

A workstation is a well human engineered combination of desk, display(s), input and pointing devices, (perhaps integrated telephone and intercom systems, audio input/output devices, micro-film readers,) and so forth. 1c7a

GRAMMAR 1c8

A grammar is a tree structured data structure that represents allowed user interactions. 1c8a

SUBSYSTEM 1c9

A subsystem is a coherent set of functions or tools, with its own command language (described by a grammar), and its own set of backend execution processes. Examples of subsystems would be a text editor, a mail subsystem, a numerical calculator, etc. 1c9a

** slide 6 ** CONTROL META LANGUAGE - CML 1c10

Control Meta Language (or CML) is a formal language developed at ARC for describing the command language and interaction of a subsystem (or an application program) with its human user. A program written in CML is compiled by the CML compiler and the object code produced is a grammar. 1c10a

CML INTERPRETTER 1c11

A CML interpreter is a program that interprets grammars

produced by the CML compiler. However, it could just as easily be a machine whose instruction set corresponds to the grammars produced by the CML compiler. 1c11a

USER-PROFILE 1c12

A user-profile is a data structure used by a command interpreter while interacting with the user, which describes to the interpreter how the system should appear to this user. 1c12a

1c13

A TWO PART NLS 1d

Now for our plans, goals, and motivations. 1d1

NLS is evolving into a two part system. 1d2

The frontend of the system will collect commands from, and in general interact with, the user. 1d3

** slide 7 ** AMong the responsibilites of the frontend system are: 1d3a

Prompting the user as to what state she is in at any time (e.g. which subsytem is currently being used) 1d3a1

Prompting the user as to what commands are available at any time 1d3a2

Prompting the user as to what action is required at any time (e.g. type something in, make a data selection, etc.) 1d3a3

Providing help to the user when requested to do so (e.g. providing a list of available commands when the user issues the ? command) 1d3a4

Collecting (and parsing) commands, including arguments, from the user, according to the currently active grammar 1d3a5

** slide 8 ** Passing (through a well defined protocol) a complete command specification on to the backend system 1d3a6

Loading new subsystem grammars from remote or local file systems 1d3a7

Passing error and other state information received from the backend on to the user, and 1d3a8

Manipulating, and displaying, the display data base in response to control information from the backend system 1d3a9

1d4

The backend of NLS will contain the core execution routines that will perform the commands specified by the user in her interactions with the frontend, 1d5

** slide 9 ** Among the responsibilites of the backend system are: 1d5a

Receiving a completed command specification from the frontend and manipulating the information data base in response to these commands 1d5a1

Detecting and passing error and other state information to the frontend as a result of the above manipulations, and 1d5a2

Passing control information to the frontend needed to manipulate the display data base in response to the above manipulations 1d5a3

1d6

We expect the frontend and backend processes of NLS to be subsets of generalized frontend and backend systems which provide tools to users, 1d7

In particular, NLS consists of many subsytems, each of which has its own grammar and related backend processes. However, even though there may be many subsystems, each having its own command language vocabulary, the way in which the user gets Help, is prompted, makes choices between alternatives, supplies parameters to commands, and so forth, is uniform thoughout all subsystems. 1d7a

Some of the subsystems currently (or soon to be) available in NLS are: 1d7b

** slide 10 ** A 2-dimensional editor subsystem for the compostion, editing, and formatting of textual information. 1d7b1

An identification subsystem for maintaining information about users, e.g. where they are to receive mail, what groups and/or organizations they belong to, etc. 1d7b2

A calculator subsystem, for performing arithmetic operations, that allows selection from and insertion into text files 1d7b3

A sendmail subsystem for distributing (and keeping track of) correspondences 1d7b4

** slide 11 ** A readmail subsystem fo assisting users in processing correspondences received from other users 1d7b5

A help subsystem to assist the user in learning NLS 1d7b6

A "programs" subsystem for helping programmers to implement and debug programs and grammars, and 1d7b7

A useroption subsystem to allow the user to customize the system for her needs or preferences 1d7b8

We expect these subsytems to be a starting point from which to provide other tools to users through the concept of frontend and backend systems. 1d7c

1d8

** slide 12 ** We expect many benefits from this split (otherwise we would not be doing it). Among the expected benefits are: 1d9

When the system is divided into its logical frontend and backend processes, and a protocol has been established for communication between these processes, it becomes possible to run frontend and backend processes on separate machines (possibly separated (or connected) by the ARPA Network). 1d9a

Frontend processes will most likely be run on a satellite/frontend machine; backend processes will most likely be run on a large timesharing machine. 1d9a1

For the sake of efficiency, and if there is room, it may be desirable to (and will be possible to) run some or all parts of one or more backend processes on the satellite machine. 1d9a2

Users will see an increase in responsiveness. 1d9b

A COMMAND META LNAGUAGE FOR AN NLS FRONTEND - basis for my talk at nbs/siggraph workshop on machine independent graphics

We have acquired many hours of experience with NLS and have formed certain strong feelings about the responsiveness requirements of such a system. The responsiveness we have been attaining from our loaded TENEX system is inadaquate, especially when being used from display terminals through the network. 1d9b1

The reponsiveness problems that we have observed are due in large measure to the fact that we are trying to run a program with very frequent activation and typically short computation per activation in a loaded timesharing system. Approximately 30% of the computing that NLS does is associated with command specification and display formating. We hope to remove much of this portion of NLS from the general time-sharing environment by moving it into a satellite machine. Thus, the user will profit through adaquate responsiveness and the portion of NLS left in the timesharing environment will recieve infrequent, command-at-a-time activations with significant computation per activation. This can be further enhanced by moving frequently used execution processes into the frontend system. 1d9b2

Users will be able to specify commands asynchronously with respect to their execution. 1d9c

A user will be able to specify new commands, and receive proper prompting, feedback, etc., without having to await the completion of previous commands. This is much more than merely being able to "type-ahead". 1d9c1

1d9d

** slide 13 ** The overall cost of a system will be reduced. 1d9e

by reducing the backend costs associated with very frequent activations, 1d9e1

by reducing network costs by transfering larger quantities of data at one time, rather than many small packets of data, and 1d9e2

by removing the 30% of NLS execution code that does command specification from the backend machine and moving it out to a dedicated satellite machine. 1d9e3

We expect that the additional equipment cost will be

easily offset by the reduction in backend machine processing. 1d9e4

** slide 14 ** We will be able to provide for well human engineered command specifcation. 1d9f

We feel that from a human engineering standpoint it is important to be able to give the user as many prompts and cues as she deems necessary during the specification of commands. This has been done to date through character-at-a-time interaction with the main timeshared computer. Clearly, line-at-a-time interaction is considerably more efficient, but lacks the prompting capability. We hope through a frontend system to achieve efficiency via command-at-a-time interaction with the main time-sharing backend system while still being able to give the user help during command specification. 1d9f1

We will have a beneficial modularity forced upon us. 1d9g

A by-product of this frontend-backend approach is that it forces the separation of command language from basic functions of a subsystem or application program. It forces us to describe through a protocol how to perform the basic operations that a subsystem makes available. This forced modularity will allow not only new command languages to easily make use of old funtions but also the development of new functions that make use of old functions through the protocols they support. This standardized application program interface should greatly facilitate future development. 1d9g1

User-specific data localized. 1d9h

The frontend system provides an ideal place to localize and utilize user-specific data. This data can be fetched from a remote or local file system when the frontend finds out who the human is and can not only influence how the system appears to her, but also can accomodate many generic functions in a way specifically tailored to her. It might for example provide her with simplified file naming, allowing her to use short names which the frontend will translate into full path names, if necesssary. 1d9h1

1d9i

** slide 15 ** We will be able to provide a uniform user interface. 1d9j

We feel that it is very important for a user to access her computer-based tools through a uniform, coherant interface. We expect this to be the largest single payoff of the frontend concept. The frontend system ALWAYS stands between the user and her tools and provides a uniform, well human engineered interface to these tools. Thus, although particular command languages may change to allow the user to refer to the functions a particular subsystem performs, the way in which the user gets Help, is prompted, makes choices between alternatives, supplies parameters to commands, and so forth, is uniform thoughout all subsystems. 1d9j1

1d10

** slide 16 ** To accomplish the above goals we have outlined a number of tasks, and have recognized a number of problems. 1d11

We must complete the logical split of NLS into frontend and backend processes. 1d11a

We must decide on an initial mapping of logical frontend and backend processes onto physical satellite and backend machines. 1d11b

We must choose a satellite machine for the frontend system, and choose an operating system for the satellite machine and a language for all frontend software. 1d11c

We must rewrite (or hopefully transliterate) the frontend programs so they can run on the satellite machine. 1d11d

1d11e

** slide 17 ** We must develop the necessary protocols for communication between the frontend and backend processes. 1d11f

We intend to try a Call-by-name protocol which will allow logical procedure calls from the frontend to backend systems (via the network). 1d11f1

There are problems involved here as to notifying the frontend system where the execution modules for individual commands live. We expect that at the same time that a grammar for a subsystem is loaded into the

frontend system, some sort of binding will occur that binds commands to the appropriate backend machines, 1d1if2

We must decide from where, and how, grammars for subsystems are loaded, 1d1ig

We must get a better understanding of where Network Graphics Protocol (NGP) fits into the picture, 1d1ih

We expect to actually drive the workstation displays by using NGP, 1d1ih1

We must address the major problem of synchronization, especially with regards to error recovery, 1d1ii

This becomes extremely difficult when we reach the point of the user specifying commands asynchroneously with their execution, 1d1iii

** slide 18 ** We must address the problem of where the file system lives, 1d1ij

Should a frontend system include a file system? Is it necessary that the frontend system have a file system? or can we get by with using only the file system of the backend system? or only a file system on the frontend system? 1d1ij1

1d12

** slide 19 ** THE SOFTWARE PART OF A FRONTEND SYSTEM 1e

Now that I have discussed where we think we are going and how we hope to get there, I'd like to go into more detail about the frontend system. As I said before, a frontend system is a logical configuration of terminals, processing capability, and programs. For the rest of this discussion, I will be concerned only with the software aspects of the frontend system, 1e1

** slide 20 ** Some likely software components (in addition to the operating system) of the frontend system are: 1e2

A command processor 1e2a

A user-profile 1e2b

a NGP package 1e2c

a Network Control Program (NCP) package 1e2d

and possibly a file system, and some backend processes 1e2e

1e3

THE COMMAND PROCESSOR 1e4

For specifying subsystem user interfaces (command languages, prompting, help facilities, etc.) we have developed a Control Meta Language (CML). (The CML compiler was written using TREE-META, a compiler compiler.) 1e4a

** slide 21 ** A machine was hypothesized which had primitive operations which interacted with the user (for example, to have her choose one of several alternatives in a command or select some text from the screen as a parameter to a command). 1e4a1

This hypothetical machine is a two address machine. The two addresses (in each instruction) are used to address the alternative(s) to this instruction and to address the successor instruction. At any point, any of the set of alternative instructions may be executed (based on user action) and the program counter moves to that instructions successor. Then, that instruction or any of its alternatives may be executed. The particular action(s) the user must take to execute one of the set of alternatives is dependent on the CML interpreter and the user-profile. 1e4a1a

** slide 22 ** A formal language and compiler were developed for this machine that allows one to describe a desired command language and interaction sequence. This language is CML. 1e4a1b

** slides 23 - 26 ** DEMO OF CML FROM SLIDES 1e4a1b1

** slides 27 - 32 ** The program (or object code) produced by the CML compiler is a tree structured grammar. 1e4a1c

An interpreter has been written to simulate this hypothetical machine on a PDP-10 for several types of display and typewriter terminals. (The command language specification is independent of the terminal type being supported with the exception of commands that only make sense for certain classes of terminals). 1e4a2

It is this interpreter that serves as a standard user interface, and is in fact the command parser or processor! 1e4a3

1e5

** slide 33 ** THE USER PROFILE 1e6

The user-profile is used by the command interpreter while interacting with the user. This data structure describes to the interpreter how the system should appear to this user (what actions she must take to disambiguate alternatives in commands, how much prompting to give her, which commands to make available to her, etc). 1e6a

1e7

** slide 34 ** WHERE WE ARE NOW 1f

Well, that covers what we hope to accomplish; now to let you know where we currently stand. 1f1

We have a running CML compiler and interpretter. 1f1a

expecting pdp 11-40 5/1 1f1b

logical spilt almost complete 1f1c

SLIDES 2

A COMMAND META LANGUAGE FOR AN NLS FRONTEND

2a1

TERMINOLOGY 2c1

NLS - 2c1a

A highly interactive computer and communications system 2c1a1

write, study, and publish documents 2c1a1a

file and retrieve material 2c1a1b

plan, organize and coordinate activities 2c1a1c

communicate with others 2c1a1d

TERMINOLOGY 2d1

FRONTEND SYSTEM - 2d1a

LOGICAL terminals, processing capability, and programs 2d1a1

FRONTEND PROCESS - 2d1b

Program running in a frontend system 2d1b1

BACKEND SYSTEM - 2d1c

LOGICAL processing capability and programs 2d1c1

BACKEND PROCESS - 2d1d

program running in a backend system 2d1d1

TERMINOLOGY 2e1

WORKSTATION = 2e1a

A well human engineered combination of desk, display(s), input and pointing devices, ... 2e1a1

GRAMMAR = 2e1b

A tree structured data structure representing user interactions 2e1b1

SUBSYSTEM = 2e1c

A coherent set of functions or software tools 2e1c1

TERMINOLOGY 2f1

CONTROL META LANGUAGE - CML - 2f1a

Formal language for describing command language and interaction of a subsystem with its human user 2f1a1

CML INTERPRETER - 2f1b

Interprets the grammars produced by the CML compiler 2f1b1

USER-PROFILE - 2f1c

Data structure which modifies how a command interpreter interacts with a user 2f1c1

FRONTEND SYSTEM RESPONSIBILITES 2g1

Prompt user as to her state 2g1a

Prompt user as to what commands are available 2g1b

Collect and parse commands from user 2g1c

Provide help to user when requested 2g1d

Collect complete command from user 2g1e

FRONTEND SYSTEM RESPONSIBILITES 2h1

Pass complete command to backend system 2h1a

Load new subsystem grammars 2h1b

Pass on error and other state information 2h1c

Manipulate, and display, the display data base 2h1d

BACKEND SYSTEM RESPONSIBILITES 2i1

Receive completed commands from frontend system 2i1a

Manipulate the information data base in response to these commands 2i1b

Detect errors 2i1c

Pass control information to frontend system 2i1d

NLS SUBSYSTEMS 2j1

EDITOR - 2j1a

compose, edit, and format textual information 2j1a1

IDENTIFICATION - 2j1b

maintain information about users 2j1b1

CALCULATOR - 2j1c

perform arithmetic operations 2j1c1

SENDMAIL - 2j1d

distribute and keep track of correspondences 2j1d1

NLS SUBSYSTEMS 2k1

READMAIL - 2k1a

assist users in processing correspondences 2k1a1

HELP - 2k1b

assist the user in learning 2k1b1

PROGRAMS - 2k1c

help implement programs and grammars 2k1c1

USEROPTION - 2k1d

allow the user to customize the system 2k1d1

BENEFITS	2m1
Overall cost of system reduced	2m1a
Reduced backend costs	2m1a1
Reduced network costs	2m1a2
Removal of 30% of NLS execution code	2m1a3

BENEFITS 2n1

Well human engineered command specifcation 2n1a

Modularity forced upon us 2n1b

User-specific data localized 2n1c

TASKS AND PROBLEMS 2q1

Develop necessary communication protocols 2q1a

Decide from where, and how, to load subsystem grammars 2q1b

Understand where NGP fits in 2q1c

Address the major problem of synchronization 2q1d

TASKS AND PROBLEMS 2r1

Decide where the file system lives 2r1a

Should a frontend system include a file system? 2r1a1

Is it necessary that the frontend system have a file system? 2r1a2

Can we get by with using only the file system of the frontend system? 2r1a3

Can we get by with using only the file system of the backend system? 2r1a4

SOFTWARE ASPECTS OF A FRONTEND SYSTEM 2s1

FRONTEND COMPONENTS 2t1

A command processor 2t1a

A user-profile 2t1b

A NGP package 2t1c

A NCP package 2t1d

possibly a file system 2t1e

Possibly some backend processes 2t1f

```
CML PARTIAL FORMAL DESRIPTION                                    2v1

 file = "FILE" ,ID $(rule / dcls)                                2v1a

    #subsys "FINISH";                                            2v1a1

 subsys = "SUBSYSTEM" ,ID "KEYWORD"
 ,SR                                                             2v1b

    #(command / rule) "END.";                                    2v1b1

 command = ("COMMAND" /
 "INITIALIZATION"                                                2v1c

    / "TERMINATION" / "RENTRY") rule
    ;                                                            2v1c1

 builtinrec =                                                    2v1d

    (("SSEL"/ "DSEL"/ "LSEL")                                    2v1d1

       '( param '))                                              2v1d1a

    / "VIEWSPECS"/ "LEVADJ";                                     2v1d2
```

```
                SAMPLE CML FILE                                  2w1

FILE nlslanguage                                                 2w1a

% COMMON RULES %                                                 2w1b

% DECLARATIONS %                                                 2w1c

% NLS EDITOR COMMANDS %                                          2w1d

   SUBSYSTEM nlseditor KEYWORD
   "EDITOR"                                                      2w1d1

   COMMAND %transpose%                                           2w1d2

   COMMAND %substitute%                                          2w1d3

   END.                                                          2w1d4

% SENDMAIL SUBSYSTEM COMMANDS %                                  2w1e

FINISH OF NLSLANGUAGE                                            2w1f
```

```
      SAMPLE CML RULES                                              2x1

% ENTITY DEFINITIONS %                                              2x1a

   editentity = textent / structure;                                2x1a1

% TEXT ENTITY DEFINITIONS %                                         2x1b

   textent = text1 / "TEXT"!L1! /
   "LINK"!L1!;                                                      2x1b1

   text1 = "CHARACTER"!L1! /
   "WORD"!L1!                                                       2x1b2

      / "VISIBLE"!L1! /
      "INVISIBLE"!L1!                                               2x1b2a

      / "NUMBER"!L1!;                                               2x1b2b
```

```
  SAMPLE CML DECLARATIONS                                    2y1

DECLARE PARSEFUNCTION                                        2y1a

   readconfirm, %reads next char if
   ca%                                                       2y1a1

   sp; %reads next char, TRUE if
   space%                                                    2y1a2

DECLARE EXT-KEYWORD                                          2y1b

   % STRUCTURAL ENTITIES %                                   2y1b1

      "BRANCH",                                              2y1b1a

      "STATEMENT";                                           2y1b1b
```

```
    SAMPLE CML COMMAND                                          2z1
COMMAND zreplace = "REPLACE"!L1!                                2z1a
    dent _ editentity                                           2z1a1
    <"at"> dest _ DSEL(dent)                                    2z1a2
    sent _ dent                                                 2z1a3
    <"by"> source _ LSEL(sent)                                  2z1a4
    CONFIRM                                                     2z1a5
    xreplace(dent, dest, sent,
    source);                                                    2z1a6
```

WHERE WE ARE NOW 2ab1

Have a running CML compiler and interpretter 2ab1a

Expecting DEC PDP 11/40 2ab1b

Logical spilt almost complete 2ab1c

KEV 30-APR-74 16:24 22862

A COMMAND META LNAGUAGE FOR AN NLS FRONTEND - basis for my talk at nbs/siggraph workshop on machine independent graphics

(J22862) 30-APR-74 16:24; Title: Author(s): Kenneth E. (Ken) Victor/KEV; Distribution: /SRI-ARC; Sub-Collections: SRI-ARC; Clerk: KEV;
Origin: (VICTOR, NBS/SIGRAPH-PAPER.NLS;2,), 30-APR-74 15:01 KEV ;

January '74 DRAFT Final Report Outline and Schedule

Branch 2 below is a refinement of the ouline presented in (IJOURNAL, 20579, 1:w). It looks forward to a modular report made up partly of papers already accepted for publication, and partly of modules that may be submitted for separate publication later. Each top-level section will include a brief introduction by RWW and or DVN that ties it in with the rest of ARC's work and the rest of the report) 1

Contract requirements: 1a

The draft (see--,1g) report is due six weeks after the end of the contract, that is on March 25th following February 10th givning us the benifit of the weekends, 1a1

The contract requires an abstract, a summary, a glossary, and head-and-tail matter (title page certain, forms, etc,) 1a2

Writer: 1b

In each case we imagine that the first ident will be the pusher for the job of writting that part of the report, The other idents are people assocatred with the work, 1b1

Author: 1c

We indend the idents listed for wriers as a first cut at authorship in the sense of credit for published work, There will be changes such as adding CFD where appropriate, DVN will listen to complaints and suggestions, 1c1

Reviewer: 1d

Any section that has not aready been through a publication process should be reviewd by some one familiar eith the work but not a writer, In some cases we have suggested a reviewer with an astrisk In other cases we wuld like the writer to suggest a reviewer when he gives an outline,, 1d1

Sources: 1e

We have suggested sources wherewe knew them to encourage writers to look to sources, In some cases they will serve with little change, 1e1

Pages: 1f

We give a guding guess at length in parentheses after each section, 1f1

January '74 DRAFT Final Report Outline and Schedule

Outlines: 1g

In the case of all papers that have not been published previously, we ask the writers for an outline to two levels below this one by Tuesday Jane 29th. It would be very sueful if writers would list additiona sources at this time. 1g1

Submission of Drafts: 1h

On the dates due, please submit outline and finished drafts to XXX by telling him/her they are ready in some file. XXX will copy them into the appropriate branch in (documentation,final). Please sumitted drafts for review by telling XXX and the reviewer. 1h1

Printing: 1i

The document due on March 25 is a draft. Theoretically we give that draft to the sponser, they make corrections, and return it to us for printing. In the past they have never changed the draft. I anticipate final printing in COM format, but that we do not need to prepare te file fo COM until the draft is fromally aproved. 1i1

New Group 1j

We hav created a new group, FINAL, which includes all the idents listed below. 1j1

Outline, Authorship Pages,, Status. 2

TOPIC	AUTHORS	SOURCES	
ABSTRACT	DVN *RWW (1p)		2b
GENERAL			2c
Introduction to Report			2c1
Summary (as contract requires(xjoournal,12345) (gjournal,21380,)	DVN *RWW DCE(5p)		2c1a
How this report is organized (IJOURNAL,20579,1)	DVN *RWW (1p)		2c1b

The Augmented Knowledge Workshop (finished))		DCE,RWW,JCN (link)	2c2
Some Basic Characteristics of a KnowledgeWorkshop System and Status of NLS with Respect to Them	(10p)	RWW *DCE(update of (20367,)	2c3
Coordinated Information Services for a Discipline or Mission Oriented Community		DCE (finished (12445,)	2c4
Aspects of ARC's Technology Transfer Strategy		RWW *DCE (5p) (RWW notes)	2c5
A View on the Symbioti Relationship Between ARC and the Applied R&D Communities		PR *RWW (10p) (PR has draft,)	2c6
SUPPORT to USERS			2d
The NLS Command Language and User Interface: An Overview		RWW,DCE,*CHI,DVN,MDK	2d1
A Command Meta-Language System foran Interactive System		CHI,DCW*	2d2
Query Software and Data Bases (20p)	HGL	KIRK EKM DVN (DIRT JMB *MDK DSK JAKE subcol,)	2d3
CHI's NCC Paper (finished)(?p)	CHI	(link)	2d4
DIA's NCC Paper (finished)(?p)	DIA	(link)	2d5
MEH's CompCon Paper (finished)(?p)	MEH	(link)	2d6
Experience with COM and the Output Processor(10p)	DVN	*NDM (COM subcol,)	2d7
An Offline Text Editing Facility	HGL		2d8
DIALOG SUPPORT SYSTEM			2e
Experience and Status of the NLS			

Journal, Ident, and Number Systems	(10p)	JEW,*RWW,JDH,PR	2e1
Design Considerations for a System to Distribute and Record Dialog	(10p)	JEW,JDH	2e2
INFORMATION MANAGEMENT			2f
An Overview of the NLS File System		HGL,*CHI	2f1
Thoughts on Needs for Personal Information Management		PR (writen)	2f2
Experience with Catalog and Directory Production (10p)		MDK,JBN,BAH (diareis)	2f3
ANALYSIS			2g
System Measurement Tools (10p)	DIA		2g1
Group Resource Allocation Design and Experience	(10p)	PR,*DCW	2g2
First Stdies of NLS Command Use and Timing	(15p)	SRL*PR (journal items)	2g3
The Analysis Function	(5p)	PR	2g4
SOFTWARE TOOLS and CONVENTIONS			2h
Software Engineering in NLS (20p)		HGL,KEV,*CHI,DCW,	2h1
THE NIC			2i
The NIC: Reflections after Three Years (15p)		MDK,*RWW	2i1
The Concept of an Evolutionary Information Center		PR (written)	2i2
The Resource Notebook (10p)	EJF		2i3
The NIC: Possible Directions (5p)	MDK		2i4
OPERATIONS			2j
Notes on Training for a Online Environment	(5p)	DVN,*JHB,JCN	2j1

January '74 DRAFT Final Report Outline and Schedule

The Knowledge Workshop: (5p) JCN
Utility Background and Plans 2j2

GLOSSARY 2k

Glossary jmb
(documentation,help,lexicon) 2k1

PRELIMINARY SCHEDULE OF UNPUBLISHED SECTIONS(,:BnDy)

O = Outline to XXX
D = Draft Due to reviewer
R = Review Due to Writer
MES = Rewriten draft due to XXX for Final Messaging
SRI = SRI Editing and Approval
PRINT = Printing

Week Ending:

	Feb 1	Feb 8	Feb 15	Feb 22	Mar 1	Mar 8	Mar 15	Mar 22			
ABSTRA	O	D	R				MES	SRI	PRINT	DVN	*RWW
GENERAL											
Summar	O	D	R							DVN	*RWW
Aspect ofTech	O	D	R							RWW	*DCE
Symbio	O	D	R							PR	*RWW
SUPORT TO USERS											
Comand Langua		O	D	R						RWW	*CHI
Meta- Langua		O	D	R						CHI	*DCW
Query/ Data		O	D	R						DVN	*MDK
COM & Output		O	D	R						DVN	*NDM
DEX II		O		D	R					HGL	*DCE

January '74 DRAFT Final Report Outline and Schedule

DIALOG SUPPORT SYSTEM

Journal etc.	O	D	R							JEW *RWW
Design		O	D	R						JEW *JDH

INFORMATION MANAGEMENT

File View	O	D	R							HGL *CHI
Catalog Prod	O	D	R							MDK *JBN

ANALYSIS

Measur ement	O		D	R						DIA *?
Group Alloca	O	D	R							PR *DCW
Usage	O	D	R							SRL *PR
Anal Funct	o	D	R							PR *RWW

SOFTWARE TOOLS AND CONVENTIONS

Soft ware		O		D	R					HGL *CHI

THE NIC

NIC Past	O		D	R						MDK *RWW
Evolu tion	O	D	R							PR *MDK
Note book	O	D	R							JAKE *MDK
NIC Future	O	D	R							MDK *RWW

OPERATIONS

January '74 DRAFT Final Report Outline and Schedule

```
Train
ing   !    O!    D!    R!     !     !     !     !     !  DVN *JHB

Utility
      ! O  !  D  !  R  !     !     !     !     !     !  JCN *JHB

GLOSSARY
Utility
      !    !D    !  R  !     !     !     !     !     !  JMB *DVN
```

DVN 30-APR-74 17:22 22863

January '74 DRAFT Final Report Outline and Schedule

(J22863) 30-APR-74 17:22; Title: Author(s): Dirk H. Van Nouhuys/DVN;
Distribution: /&DPCS RWW(for the record); Sub-Collections: SRI-ARC DPCS;
Clerk: DVN;
Origin: (VANNOUHUYS, FINAL.NLS;4,), 30-APR-74 15:01 DVN ;March 24

DVN 30-APR-74 17:32 22864

Revised Revised Qucik Reference to New NLS

If you have questions about this information, please bring them to the Mayday all-ARC meeting.

Revised Revised Qucik Reference to New NLS

This is a very brief account of the most important ways in which New Display NLS is different to use from Old Display NLS. 1

Questionmark (?) 2

The new NLS is blessed with excellent questionmark facilities. In both display and TNLS "?" at any point will list the choices available to you. Note that the posibilities change every time the system sees a new character. Thus "?" correctly gives you different answers each time you type and according to the recognition scheme see--6) you have chosen. 2a

In Expert recognition mode the space you put in before less common commands counts as a node on the questionmark tree. That is, a space and then a question mark at the command level will elicit a different and mutually exclusive list of possibilites from a question mark directly at the command level. (,6e) 2a1

If you want to insert "?" as a character of text, put <^V> a head of it. 2a2

HELP 3

<^Q> puts you into the HELP subsystem and into the HELP data base at a point corresponding to the command verb you took just before you struck the character. 3a

The HELP system attempts to provide its own instruction. 3a1

The HELP data base was basically written for new users of TNLS, the group we believed has the most desparate need. It gives short shrift to some commands peculiar to DNLS, e. g. the subvarieties of jump. We plan to fill out the DNLS material as priorities permit. 3b

The HELP system necessarilly grows a little behind the system it describes. There are some empty spots, some features of questionable design, and some program bugs. Please report them to the feedback system. (journal,22669,) 3c

Subsystems: 4

The commands in NLS have been divided into subsystems. The commands we use most are in the Editor subsystem. Others are available through: Calculator, Help, Identification, Sendmail, Programs, TENEX, and User-options. 4a

"Goto" takes you to another subsystem. Quit returns you to the previous subsystem in a ring. The name of your current subsystem

appears in the upper left corner of the screen,"<" prints out your subsystem ring, 4b

Execute allows you execute one command in a subsystem and pop back to the previous subsystem, 4c

The Journal: 5

The journal has split into two subsystems, Sendmail and Readmail, 5a

The only old commands that have a new command word in Sendmail are "Go" which has become "Done",and "Distribute" which has become "Send", 5b

Readmail doesn't work yet, You must read your mail as you do any NLS file, 5c

To send mail, Goto Sendmail as a subsystem, Commands are then availaibe to you in any order, (You don't have to begin by naming the thing you want to send,) The command "Initialize" gives you a fresh start without saying "Done", 5d

For information on new services in Sendmail, Goto Help and show "sendmail", 5e

Recognition: 6

New NLS has several command recognition schemes, My guess is that most people familiar with NLS will use a mode called "expert anticipatory", 6a

You may set your own recognition mode through the Useroption Subsystem (,14), 6b

<documentation,help,recognition>describes the possiblities, 6c

In expert mode some list of command words is possible at any point following the prompt C:, From that list, NLS will recognize and echo the most commonly used word by its first character, To use a command word that is not commonest, you have to type a space and then type characters until NLS can recognize, 6d

E,g, from the command base state, "S" will evoke "substitute", but to get "Set" you must type "<SP>se" and to get Stop "<SP>st", At the command base state, "L" will elicit "Load" but "<SP> L" will elicit "Logout", 6d1

I have found that 3/4 or more of my commands are first-character commands, 6d2

In expert mode, if you type an impossible letter, NLS will question you, but nicely allow you to put in another letter instead. I.e."O D P" will succeed althougth the command is now "Output Printer" because no "D" may follow "Output". 6e

New Command Names: 7

Old Commands: 7a

The most important commands that have changed the first command word are as follows, with the old name first. They are in the Editor Subsystem unless noted. 7a1

Execute Assimilate = You cannot now copy text through a filter. 7a1a

Execute Connect to Terminal = Connect to Terminal 7a1b

Execute Device Type = Simulate Terminal Type 7a1c

Execute File Verify = Verify File 7a1d

Execute Insert Sequential= Copy Sequential 7a1e

Execute Logout = Logout 7a1f

Execute Marker Fix = Mark Character (also Delete Marker) 7a1g

Execute Marker List = Show File Show Marker List 7a1h

Execute Status = Show Status 7a1i

Execute Unlock = Delete Modifications (also Undelte Modifications) 7a1j

Execute Viewchange = Goto Useroptions [subsystem] 7a1k

Note however that changes made via usroptions remain after you log out ad until you change them gain via useroptions. 7a1k1

Execute journal = Goto Sendmail [subsystem] 7a1l

Goto Display Area Format Character Size = Set Character Size 7a1m

Goto Display Area Vertical/Horizontal Split = Split Window Vertically/Horizontally. 7a1n

Goto Display Area Move Boundry = Move Boundry 7a1o

Link (in TENEX) = Connect to [In the NLS Editor] 7a1p

Null File = Create File 7a1q

Output Device Printer = Output Printer 7a1r

Output File = Update File Compact 7a1s

Sendprint (in TENEX) = Output Remote Terminal[In the NLS Editor] 7a1t

Update = Update File + Options 7a1u

Viewset = Set Viewspecs 7a1v

New Comands: 7b

Directory Commands:
Now you can deal with your directory in NLS as well as in TENEX, Delete File, Undelete File, Trim, Expunge, Connect to, etc are commands in NLS, Copy Directory copies your directory into a file as a plex, Options in Copy Directory allow you to see the information TENEX has about your files in various orders. Show Directory sends the same information on the screen so command accept wipes it away For more information, Goto Help and show "show directory" and "copy directory",. 7b1

The directory commands have a few bugs left. The most troublesome is that in specifying the options of "show directory" or "copy directory" you must take the option "for file", which limits readout to one file, last. 7b1a

Set Commands:
With Alternatives following Set you can control recognition, prompts, viewspecs, and create what used to be called a content analyser pattern, For every Set a Reset command returns you to the default, For more information, Goto Help and show "set". 7b2

Insert
If you end a command with the INSERT character (by default <^E>) the next command will be insert statement at the CM. 7b3

Repetition: 8

By default the system now returns to base command state after every command, BUT, if you end any command with A Repeat Character the system will carry out the command, return to base command state, and go forward in the same command until it meets a field

that is not a command word. The Repeat character is <^B> or he right hand two mouse buttons down an up. 8a

You may then step back through the command word-by-word with <^A>. The parser will continue to repeat the command in that way until you hit Command Delete. The effect is very much like creating a mode for each command like the old jump mode. 8a1

If you hit Repeat Character at command reset, it will repeat your last command out to the first field that is not a command word. 8b

Optionss and Alternatives 9

When several command terms may follow a given command term (as plex, branch, etc. may follow delete or typing or a bug may follow Replace) we speak of alternatives. When typing the Option Character allows you to put in command terms that are otherwise inacessible, we speak of options. With full prompting on, square brackets, [], inclose optional terms. 9a

By default <^U> is the option character 9b

Addressing 10

Bugging works as in old NLS. B: prompts for bugging. 10a

All other addresses are special cases of links. The prompt is A:. Following A: you can put a "directoryname,filename," separated by commas. You may force recognition with altmode. You may omit them and default to the directory you are connected to and the current file. After the filename position you may use the intrafile address forms in any order as in old TNLS. 10b

Thus a full NLS address looks like this: 10c

SITE,DIRECTORY,FILE,INFILEADDRESS 10c1

You can address a character in an off-screen statement by naming it in quotes in a string . Thus a full address of a certain ^o is (ARC,vanNouhuys,onetest,"crypto"). Following A: you would not need the parentheses and could force recognition after "vann" and "one". 10c2

In general you must type the option character <^U> to enter an address (prompted by A:). Jump to Link allows you to enter an address without asking for an option. 10d

Do NOT put periods in front of statement names or numbers or

SID's; instead put them in front of structural relationships (i.e, .u .d .b or .2d or .5p, or .uussr etc.). 10e

Prompting 11

New Display NLS has prompts like the V: and I: in old TNLS and the journal. You can set prompting in one of three modes: Full, Partial, or off. The default is Full, Partial does not show options and some alternatives. The complete list of prompts is: 11a

C: calls for a Command Word 11a1

A: calls for an Address. 11a2

T: calls for you to type something in, e.g. free text, or an ident. 11a3

OK: calls for confirmation of the command, usually CA or REPEAT (<control-b>). 11a4

B: calls for a Bug. 11a5

[], Square Brakets indicate that you have to use the Option Character to specify the thing named inside the brackets. In many cases, for example, you may either bug something on the screen or enter an Option Character followed by any Address. The prompt would be: B:/[A:]. 11a6

CA: calls for a command accept. 11a7

Y/N: calls for "y" for "yes" or "n" for "no" in commands like substitute where NLS wants to know if you want to repeat a specification step. 11a8

[**] shows that you may use <control-u> to gain acces to a list of optional command word. 11a9

V: calls for viewspecs 11a10

L: calls for level adjustment. 11a11

SP: calls for a space as a quasi CA e.g. following idents in journal submission. 11a12

RPT: calls for a <^B> to repeat the use of some buffer as in content searches. 11a13

"> ..." generally appears in the command feedback line when the

system is doing something; it is roughly synonymous with "RUNNING" 11a14

My experience is that it is nicest to run with prompts off once you get the hang of most of the commands you use. 11b

Jumping: 12

Jump commands no longer have a special mode of repetition; CA or <"B> terminate them in the same way they do other commands. 12a

In my use anyway, the most common jump command is simply "j" which echos "jump to" and takes a bug. 12b

Jump to Return and Jump to File Return no longer follow a ring, but go down a stack. Every time you move you add an address to the top of the stack. Jump to Ahead no longer exists. By default you have ten addresses on the stack. 12c

From/To 13

Now you Move and Copy things from someplace to someplace else instead of vice versa. It's not too hard to get used to and the noise words help. Be careful in the beginning. 13a

The Useroptions Subsystem 14

The Useroption Subsystem takes over some of the work of the old Viewchange System and adds many new ways for you to cut NLS to suit yourself. It controls feedback in the form of heralds, prompts, noise words, etc.; it controls recognition and default viewspecs; it redefines control characters (like the old NLSControlCharacters branch); and it formats pages in the TNLS print command. Execution of commands in the User Option subsystem rewrites a file (the profile) that determins how these paramaters are set when you log in from then on. 14a

The Programs Subsystem 15

It replaces and augments the old "Go to Program" commands. 15a

Old Commands with New Names: 15b

Get = Load 15b1

Deinstitute = Delete 15b2

Status = Show 15b3

For information on new commands, Goto HELP and show "Prorgrams". 15c

Omissions: 16

For the time being you cannot read files marked private at all in New NLS. 16a

In general user programs written for Old NLS will not run or compile. To have your user program converted to run in New NLS, see Elizabeth Micheal. 16b

Query will not run in New NLS. 16c

For the time being you cannot copy with a filter (the asimilate function). 16d

An Important Bug: 17

If you have bugged a character on the screen, you cannot back space to an earlier command step in the Substitute, Interogate, or Copy Directory commands. You may end up in a place where you have to go out and reset. 17a

To get Old NLS, type "Oldnls <CR>" at the Tenex level. 18

DVN 30-APR-74 17:32 22864

Revised Revised Qucik Reference to New NLS

(J22864) 30-APR-74 17:32; Title: Author(s): Dirk H. Van Nouhuys/DVN; Distribution: /SRI-ARC LEG NJN DHC JMB(I am going to talk about HELP at the all-arc meeting tomorrow, Send me any thoughts you want passed on); Sub-Collections: SRI-ARC; Obsoletes Document(s): 22852 18374; Clerk: DVN;

DIA 30-APR-74 18:05 22865

All-ARC meeting:

There is an all-ARC meeting Wed, 5/1 at 11:00, 1

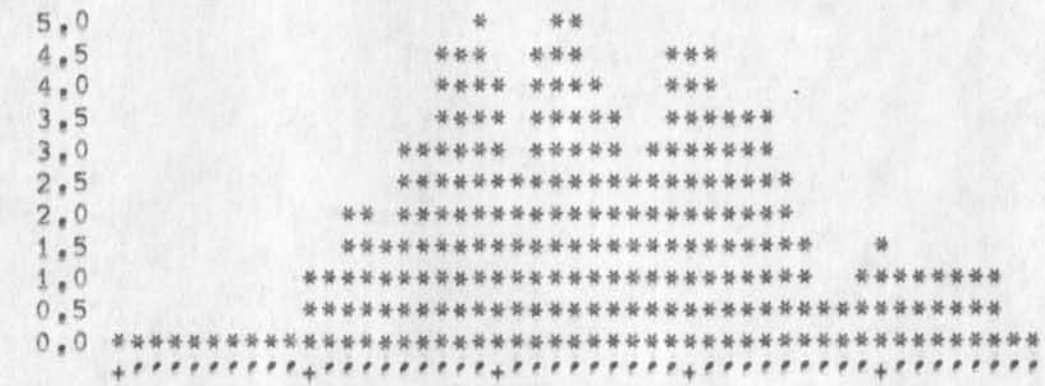
DIA 30-APR-74 18:05 22865

All-ARC meeting:

(J22865) 30-APR-74 18:05; Title: Author(s): Don I. Andrews/DIA;
Distribution: /SRI-ARC; Sub-Collections: SRI-ARC; Clerk: DIA;

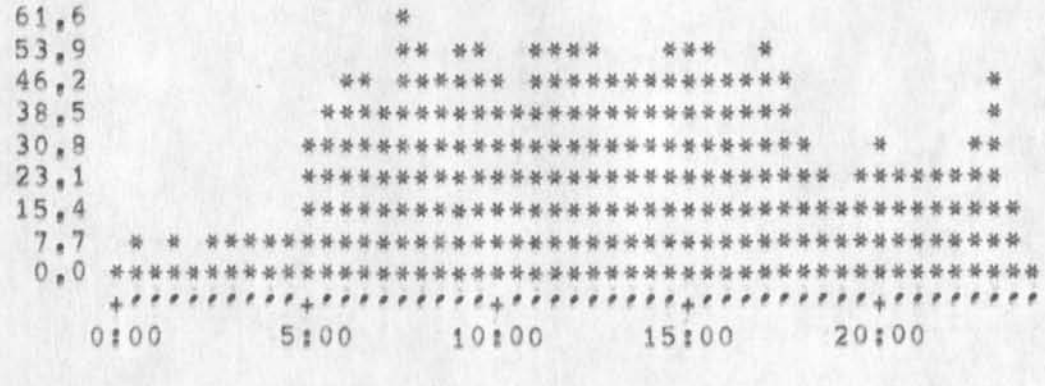
Superwatch Average Graphs for Week of 4/21/74

TIME PLOT OF AVERAGE NUMBER OF GO JOBS FOR WEEK OF 4/21/74
x axis labeled in units of hr:min, xunit = 30 minutes 1



1a

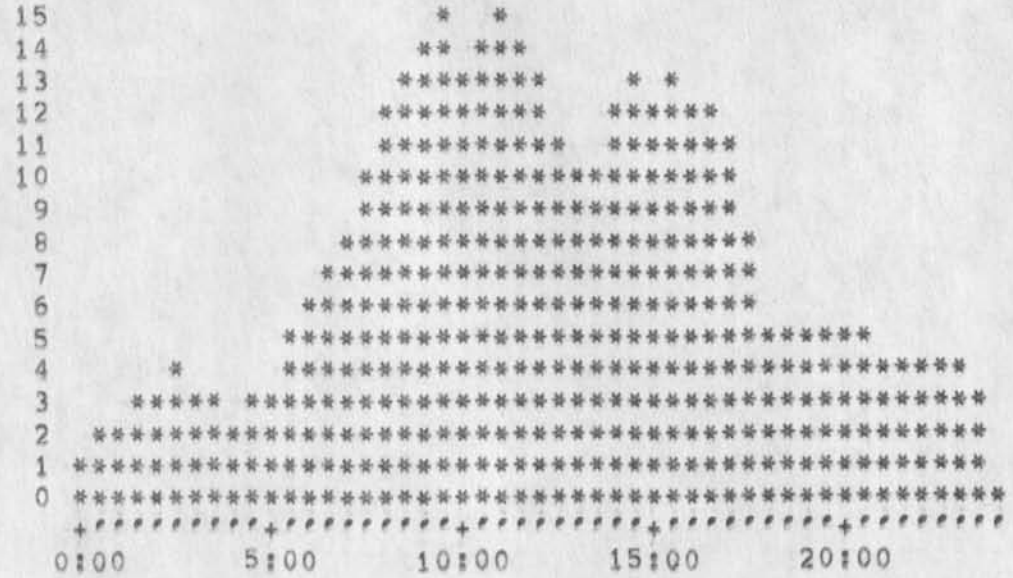
TIME PLOT OF AVERAGE PER CENT OF CPU TIME CHARGED TO USER ACCOUNTS
FOR WEEK OF 4/21/74
x axis labeled in units of hr:min, xunit = 30 minutes 2



2a

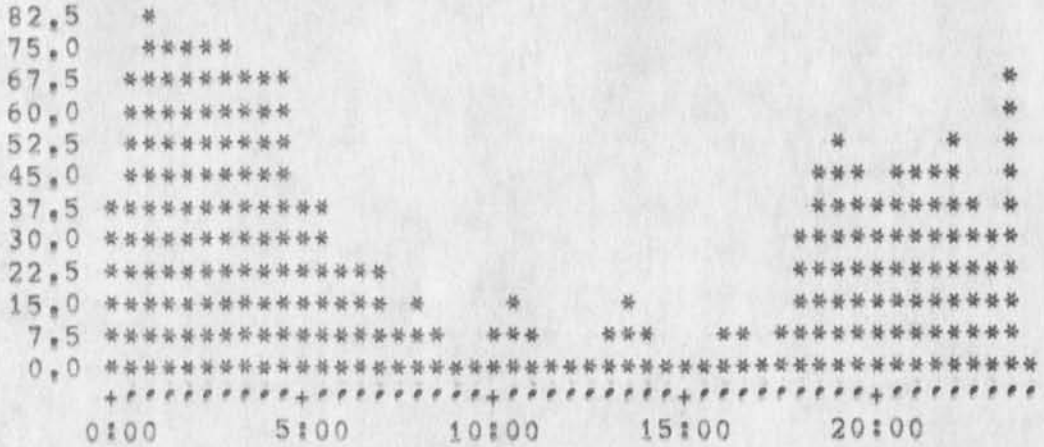
Superwatch Average Graphs for Week of 4/21/74

TIME PLOT OF AVERAGE NUMBER OF USERS FOR WEEK OF 4/21/74
x axis labeled in units of hr:min, xunit = 30 minutes 3



3a

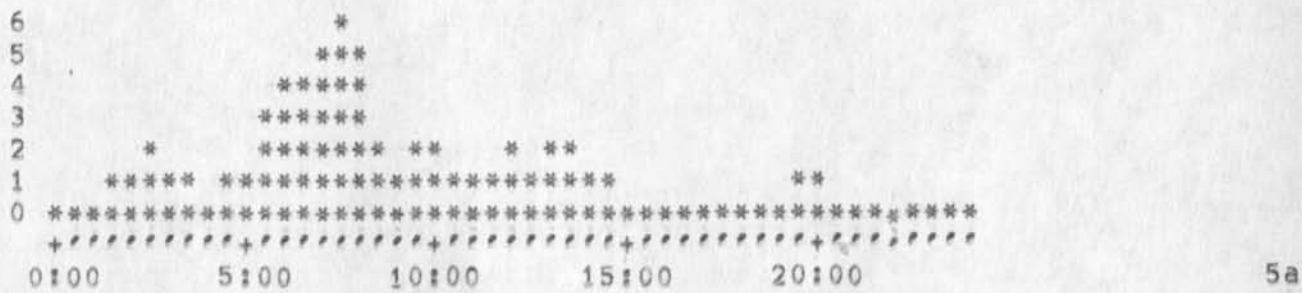
TIME PLOT OF AVERAGE IDLE TIME FOR WEEK OF 4/21/74
x axis labeled in units of hr:min, xunit = 30 minutes 4



4a

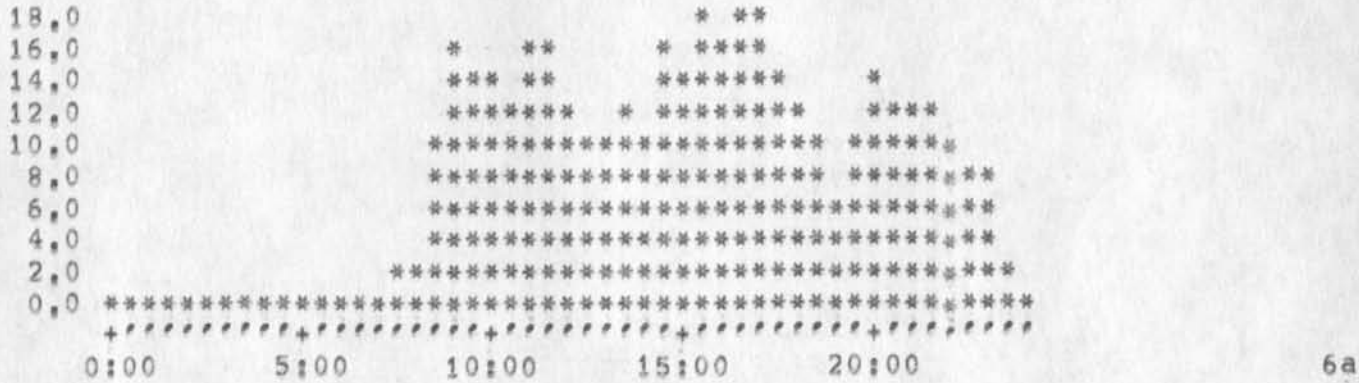
Superwatch Average Graphs for Week of 4/21/74

TIME PLOT OF AVERAGE NUMBER OF NETWORK USERS FOR WEEK OF 4/21/74
x axis labeled in units of hr:min, xunit = 30 minutes 5



5a

TIME PLOT OF AVERAGE PER CENT OF SYSTEM USED IN DNLS FOR WEEK OF 4/21/74
x axis labeled in units of hr:min, xunit = 30 minutes 6



6a

SRL 1-MAY-74 08:39 22866

Superwatch Average Graphs for Week of 4/21/74

(J22866) 1-MAY-74 08:39; Title: Author(s): Susan R. Lee/SRL;
Distribution: /JCN RWW DCE PR JCP DVN JAKE DLS BAH WRF; Sub-Collections:
SRI-ARC; Clerk: SRL;
Origin: <LEE>WEEK4/21GRAPHS.NLS;1, 1-MAY-74 08:35 SRL ;

MDK 1-MAY-74 09:12 22867

note to jcn re NIC draft and reply

Jim I'm sure you've seen Fields SNDMSG reply to my draft note on the upcoming NIC changes. I completely disagree with his point four, which was not to mention services that are to be discontinued. 1

At the most elemental level, if we fail to be explicit, we will get swamped with calls to explain further what's really going to happen. 2

At a more important level, not to mention these things explicitly is grossly unfair to those who currently depend on these services. Our interest should be in playing it straight with all those who have utilized our services, and helped them to mature. They may well be our customers in the future. They now will have to plan and make adjustments, and the more lead time they have the better off they'll be. 3

(J22867) 1-MAY-74 09:12; Title: Author(s): Michael D. Kudlick/MDK;
Distribution: /MDK; Sub-Collections: SRI-ARC; Clerk: MDK;

yet another epistle ... same subject as others, this, from JCN

1-MAY-74 1309-PDT NORTON: nic note: CF Position
Distribution: KUDLICK, watson, norton
Received at: 1-MAY-74 13:09:47 1

Hmm It's pretty clear that Craig didn't know what ARPA is cutting off with the NIC services being discontinued, Too bad, Shows a gap in understnding and I think not a great effort on their part to tell what was up whether it was good or nt, We MUST tell the users what thye will no longer get, AND WE WILL,, But it sure must be done in a way that doesn't make IPT look bad even if they are, I'm assuming you Mike and Dick can negotiate with Craig, 1a

His point that we cannot announce the new set of services until he gets (and maybe accepts at least verbally) the New Nc proposal is RIGHT,, We can't I'd appreciate a check with Jake to se if she's sent him the drft of the proposal she and I agreed she'd send to Craig, With that he might be willing to say go ahead,,,but the points about what's NOT being offerred still need negotition, 1b

His point ant he doesn't want the questions via sndmsg to him is OK I suspect that he is trying to keep discontented (if any in his mnd) users from bugging him,,OK I'll field the questions so to speak,, I've not got the messga in front of me so forget if there's any other thing I should comment upon,, That do it?? Good Luck Jim 1c

(J22869) 1-MAY-74 13:21; Title: Author(s): Michael D. Kudlick/MDK;
Distribution: /JAKE; Sub-Collections: SRI-ARC; Clerk: MDK;

```
FILE labels1 % L10 <KUDLICK>LABELS1 %                                    1
% PROGRAM TO GENERATE 80-char CARD IMAGES FOR NIC ADDRESS LABELS %       2
% global declarations %                                                  3
   DECLARE STRING blankcard = "
   ";                                                                    3a
   DECLARE STRING entry[2000], err[300], grpident[10], id[10];           3b
   DECLARE STRING nothing[5];                                            3c
   DECLARE idstid;                                                       3d
   DECLARE automatic;  % TRUE if automatic mode %                        3e
   DECLARE end, outstid;
                                                                         3f
% main control %                                                         4
   (labels1) PROCEDURE;                                                  4a
      LOCAL                                                              4a1
         i;   % loop index %                                            4a1a
      LOCAL TEXT POINTER tp1, tp2, tp3, tp4,  z1, z2;                    4a2
      LOCAL STRING inpstr[400]; % input collection %                     4a3
      LOCAL STRING nxtident[10], exp[300], noexp[300], name[300];        4a4
      LOCAL STRING grplist[6000];                                        4a5
      % initialization  %                                               4a6
         automatic _ FALSE;                                              4a6a
         outstid _ origin;                                               4a6b
         *nothing* _ NULL;                                               4a6c
         *inpstr* _ NULL;                                                4a6d
      %get run parameters%                                               4a7
         % GET OUTPUT FILE NAME %                                        4a7a
```

```
LOOP                                                                    4a7a1
   BEGIN                                                                4a7a1a
   typeas($" Output file = ");                                          4a7a1b
   *xlit* _ NULL;                                                       4a7a1c
   txtlit($xlit);                                                       4a7a1d
   IF NOT FIND SF(*xlit*) > [',] THEN *xlit* _ *xlit*,
   ",NLS";                                                              4a7a1e
   %  Open calls err if it doesn't find the file.  %                    4a7a1f
      ON SIGNAL ELSE GO TO newfile;                                     4a7a1f1
   IF outstid,stfile _ open(0, $xlit) THEN                              4a7a1g
      BEGIN                                                             4a7a1g1
      typeas($"  (old file) CONFIRM ");                                 4a7a1g2
      IF input() # CA THEN REPEAT LOOP;                                 4a7a1g3
      % using old file = position to tail %                             4a7a1g4
         outstid _ getail(getsub(outstid));                             4a7a1g4a
      EXIT LOOP;                                                        4a7a1g5
      END                                                               4a7a1g6
   ELSE                                                                 4a7a1h
      BEGIN                                                             4a7a1h1
      (newfile):                                                        4a7a1h2
      typeas($" (new file) CONFIRM ");                                  4a7a1h3
      IF input() # CA THEN REPEAT LOOP;                                 4a7a1h4
      ON SIGNAL ELSE NULL; %disarm all of them%                         4a7a1h5
      % force NLS extension %                                           4a7a1h6
         IF NOT FIND SF(*xlit*) > '< 1$(LD/'=) '>
```

```
            is(LD/'-) ^tp1 ([';] ^tp2 _ tp2 / TRUE ^tp2)
            THEN                                                4a7a1h6a

               BEGIN                                            4a7a1h6a1

               typeas($" cant understand filename:  ");         4a7a1h6a2

               typeas($xlit);                                   4a7a1h6a3

               REPEAT LOOP;                                     4a7a1h6a4

               END;                                             4a7a1h6a5

            *xlit* _ SF(*xlit*) tp1, ",NLS", tp2 SE(*xlit*);
                                                                4a7a1h6b
         IF NOT outstid,stfile _ opwk(0, $xlit) THEN            4a7a1h7

            typeas($" Bad File Name ") % (that's all it
            could be)  %                                        4a7a1h7a

         ELSE EXIT LOOP;                                        4a7a1h8

         END;                                                   4a7a1h9

      END;                                                      4a7a1i

   % GET IDENTS FROM USER %                                     4a7b

      IF NOT (idstid,stfile _ open(0,
      jfiname($"identfile")))THEN err($" Labels program unable
      to open IDENT FILE, ");                                   4a7b1

      ON SIGNAL ELSE                                            4a7b2

         BEGIN                                                  4a7b2a

         close(idstid,stfile);                                  4a7b2b

         RETURN;                                                4a7b2c

         END;                                                   4a7b2d

      typeas ($" Manual Ident Entry Mode? (CONFIRM) ");         4a7b3

      CASE answer() OF                                          4a7b4

         = TRUE:                                                4a7b4a

            % collect ident list %                              4a7b4a1
```

```
            identlist(sinpstr, idstid,stfile);                    4a7b4a1a
        ENDCASE                                                   4a7b4b
          BEGIN                                                   4a7b4b1
          crlf();                                                 4a7b4b2
          typeas(s"Automatic Mode");                              4a7b4b3
          crlf();                                                 4a7b4b4
          automatic _ TRUE;                                       4a7b4b5
          % set up the input string to specific values %          4a7b4b6
            *inpstr* _ SP, "TIPG", SP, "USERG", SP,
            "SERVERG", SP, "ASSOCG";                              4a7b4b6a
          END;                                                    4a7b4b7
    % GET IDENTS FROM IDENTFILE %                                 4a7c
      FIND SF(*inpstr*)"z1;                                       4a7c1
      crlf();                                                     4a7c2
      (mainloop):                                                 4a7c3
      LOOP                                                        4a7c4
        BEGIN                                                     4a7c4a
        *entry* _ NULL;  % ident file entry %                     4a7c4b
        *xlit* _ NULL;                                            4a7c4c
        *lit* _ NULL;                                             4a7c4d
        IF NOT FIND  z1 > s(SP/',) "z1 1s(LD/'=) "z2 THEN EXIT
        LOOP;                                                     4a7c4e
        *nxtident* _ z1 z2;                                       4a7c4f
        *grpident* _ NULL;                                        4a7c4g
        *grplist* _ NULL;                                         4a7c4h
        z1(1) _ z2(1);                                            4a7c4i
```

LABELS1 L10 program for oldnls ... doesn't include mem-list stuff

```
            IF NOT ckident($nxtident, sentry, idstid,stfile) THEN  4a7c4j
               BEGIN                                               4a7c4j1
               *xlit* _ *nxtident*, " ... skipping this ident:
               invalid";                                           4a7c4j2
               typeas($xlit);                                      4a7c4j3
               REPEAT LOOP;                                        4a7c4j4
               END;                                                4a7c4j5
            *xlit* _ *nxtident*, EOL;                              4a7c4k
            typeas($xlit);                                         4a7c4l
            IF orgrptst(sentry,0) THEN      % TRUE if org/group %  4a7c4m
               BEGIN                                               4a7c4m1
               *grpident* _ *nxtident*;                            4a7c4m2
               *grplist* _ *nxtident*;                             4a7c4m3
               % expand group or orgzn ident ... %                 4a7c4m4
               % ...   and build records for labels file %         4a7c4m5
               iexpmdk ($grplist, idstid,stfile);                  4a7c4m6
               crlf();                                             4a7c4m7
               END                                                 4a7c4m8
            ELSE process($nothing);         % individual ident  %  4a7c4n
            END; % of main loop %                                  4a7c4o
         % finished--clean up  %                                    4a8
            close(outstid,stfile);                                 4a8a
            close(idstid,stfile);                                  4a8b
         crlf();                                                   4a9
         typeas($" Label Processing Finished Normally ");          4a10
```

```
    RETURN END.                                                          4a11

(process) PROC (groupid);                                                5

  % output the cards for the entry in "entry", inserting the group
  ident "groupid" if supplied. %                                         5a

  LOCAL STRING name[300];                                                5b

  REF groupid;                                                           5c

  LOCAL TEXT POINTER current, tp1, tp2, t7, t8;                          5d

  LOCAL i;                                                               5e

  getinam(sentry, $name, 0, 0); % put name in "name" %                   5f

  % DO LINE 1 (really, "card image" #1): NAME and GROUPID %              5g

    IF FIND > (SF(*name*) ['*'] *tp1 *tp2 _tp2) THEN                     5g1

      *name* _ SF(*name*) tp2, SP, tp1 SE(*name*);                       5g1a

    *xlit* _ *blankcard*;                                                5g2

    IF name.L > 24 THEN                                                  5g3

      BEGIN                                                              5g3a

      *lit2* _  EOL, *name*, "     name too long, truncated to:";         5g3b

      typeas($lit2);                                                     5g3c

      FIND SF(*name*) *t7;                                               5g3d

      FIND SE(*name*) *t8;                                               5g3e

      LOOP                                                               5g3f

        BEGIN                                                            5g3f1

        IF name.L < 25 THEN EXIT LOOP                                    5g3f2

        ELSE                                                             5g3f3

          BEGIN                                                          5g3f3a
```

```
            IF NOT FIND t8 < SPT 1SNP "t8 > THEN FIND t7 > 24$24CH
            "t8;                                                    5g3f3b
            name,L _ t8[1];                                         5g3f3c
            REPEAT LOOP;                                            5g3f3d
            END;                                                    5g3f3e
         END;                                                       5g3f4
      crlf();                                                       5g3g
      typeas($name);                                                5g3h
      crlf();                                                       5g3i
      END;                                                          5g3j
   *xlit*[1 TO 1+name,L] _ *name*;                                  5g4
   IF NOT automatic THEN IF groupid,L THEN  *xlit*[30-groupid,L TO
   30] _ SP, *groupid*;                                             5g5
% DO LINES 2 THRU 8 %                                              5h
   end _ FALSE;                                                     5h1
   *lit2* _ NULL;                                                   5h2
   laddress(sentry, $lit2 %dest%, 0,0, idstid,stfile);              5h3
   *lit2* _ *xlit*[1 TO 30], EOL, *lit2*;                           5h4
   astruc($lit2); % capitalize %                                    5h5
   FIND SF(*lit2*) "current;                                        5h6
   FOR i _ 1 UP 1 UNTIL > 4 DO                                      5h7
      card($current, lit2,L);                                       5h7a
   IF NOT end THEN                                                  5h8
      BEGIN                                                         5h8a
      crlf();                                                       5h8b
      *xlit* _ *name*, " address label truncated ", EOL;            5h8c
```

```
        typeas($xlit);                                          5h8d
        END;                                                    5h8e
   RETURN END,                                                  5i

(card) PROC (current, leng);                                    6
   % build a card image from input, starting at textpointer
   "current".  "leng" is length of input string.  Leaves "current"
   updated.  Sets global "end" if applicable.  %                6a
   LOCAL column, difference;                                    6b
   REF current;                                                 6c
   LOCAL TEXT POINTER t1, t2, t7, t8;                           6d
   *xlit* _ *blankcard*;                                        6e
   IF NOT end THEN % more input %                               6f
      BEGIN                                                     6f1
      column _ 1;                                               6f2
      LOOP                                                      6f3
         BEGIN                                                  6f3a
         IF NOT FIND current > ([EOL] "t1 "t2 _t2) THEN         6f3b
            BEGIN                                               6f3b1
            t2[1] _ leng + 1;                                   6f3b2
            t1[1] _ leng;                                       6f3b3
            end _ TRUE;                                         6f3b4
            END;                                                6f3b5
         difference _ t2[1] - current[1];                       6f3c
         LOOP
           % This inner loop truncates the mail address to less than
```

```
LABELS1  L10 program for oldnls ... doesn't include mem-list stuff


         30 characters (ending just before a NP char), to fit on a
         label line;  it saves the position of truncation to allow
         the address to be continued on the next line %
                                                                    6f3d

            BEGIN                                                   6f3d1

            IF difference < 31 THEN EXIT LOOP                       6f3d2

            ELSE                                                    6f3d3

               BEGIN                                                6f3d3a

               IF NOT FIND t2 < SPT 1sNP "t2 > "t1 THEN EXIT LOOP;  6f3d3b

               difference _ t2[1] - current[1];                     6f3d3c

               REPEAT LOOP;                                         6f3d3d

               END;                                                 6f3d3e

            END;
                                                                    6f3d4

         *xlit* [column TO column + difference] _ current t2;       6f3e

         current[1] _ t1[1];                                        6f3f

         column _ column + 30;                                      6f3g

         IF column > 31 OR current[1] >= leng OR end THEN EXIT LOOP;
         % done with a whole card %                                 6f3h

         END;                                                       6f31

      END;                                                          6f4

   %truncate off any extra info in last field%                      6g

   *xlit*[61 TO 80] _ *blankcard*[61 TO 80];                        6h

   xlit.L _ 60;                                                     6i

   outstid _ cis(outstid, sxlit, sucdir);                           6j

   RETURN END.
                                                                    6k

(iexpmdk)PROC(delstr,idfnum);                                       7
```

```
   LOCAL TEXT POINTER ptr, z1, z2;                                    7a
   REF delstr;                                                        7b
   *delstr* _ *delstr*, ';;                                           7c
   makeptr(asrref(&delstr),$ptr);                                     7d
   intids(0);                                                         7e
   LOOP                                                               7f
      BEGIN                                                           7f1
      % get next individual ident within current group/orgzn  %       7f2
      *entry* _ NULL;                                                 7f3
      IF NOT getmdkids($ptr, $entry, 0, idfnum) THEN EXIT;            7f4
      % get full identfile info for this ident %                      7f5
      getiid($entry, 0, $z1, $z2);                                    7f6
      *auxlit* _ " ", z1 z2;                                          7f7
      typeas($auxlit);                                                7f8
      % format statements for CDC 6600 mailing labels program %       7f9
      process(IF automatic THEN $nothing ELSE $grpident);             7f10
      END;                                                            7f11
   RETURN;                                                            7g
   END.                                                               7h
                                                                      7i
(getmdkids) PROCEDURE (ptr, astr, infotype, idfnum);                   8
   LOCAL expchr, gpstid;                                              8a
   LOCAL TEXT POINTER idf, ide, tmpptr, srcptr, dstptr;               8b
   LOCAL STRING idstr[20], infostr[500];                              8c
   REF ptr, astr;                                                     8d
```

```
%reads an ident from pointer ptr into astring idstr, and then
calls ckident to get info on it,                                    8e

   IF infotype = 0, then it returns all of the info in astr, if
   infotype = 1, then the name only is returned,                    8e1

   Uses infostr as a work area%                                     8e2

expchr _ 0;                                                         8f

%first, read ident%                                                 8g

   LOOP                                                             8g1

      BEGIN                                                         8g1a

      CCPOS ptr;                                                    8g1b

      IF FIND $NP "idf '; THEN                                      8g1c

         BEGIN                                                      8g1c1

         IF NOT popids(&ptr) THEN RETURN(FALSE); %no more idents%   8g1c2

         END                                                        8g1c3

      ELSE EXIT LOOP;                                               8g1d

      END;                                                          8g1e

   IF NOT FIND idf [ NP / '; / '( ]                                 8g2

      "ptr _ptr "ide _ide THEN                                      8g2a

         err($"Ident List Format Error");                           8g2a1

   FIND ptr ($NP '( ['')] "ptr);                                    8g3

   IF FIND idf ('&/"") "idf  < CH > THEN expchr _ READC;            8g4

   *idstr* _ idf ide; %ident%                                       8g5

%Now get info, and check ident%                                     8h

   IF ckident($idstr, $infostr, idfnum : gpstid) THEN %return
   something%                                                       8h1

      BEGIN                                                         8h1a
```

LABELS1 L10 program for oldnls ... doesn't include mem-list stuff

```
        IF orgrptst($infostr, 0) THEN                                    8h1b
          BEGIN                                                          8h1b1
          expchr _ TRUE;                                                 8h1b2
          IF expchr THEN                                                 8h1b3
            BEGIN                                                        8h1b3a
            getimem($infostr, 0,  $dstptr, 0);                           8h1b3b
            IF FIND dstptr -EOL %membership list present% THEN           8h1b3c
              BEGIN                                                      8h1b3c1
              pushids(&ptr);                                             8h1b3c2
              dstptr _ gpstid;                                           8h1b3c3
              FIND dstptr "ptr;                                          8h1b3c4
              RETURN(getmdkids(&ptr, &astr, infotype, idfnum));          8h1b3c5
              END;                                                       8h1b3c6
            END;                                                         8h1b3d
          END;                                                           8h1b4
        END;                                                             8h1c
      %Now edit and append to astr %                                     8h2
        IF infotype = 1 THEN                                             8h2a
          getifnf($infostr,  $infostr);                                  8h2a1
        *astr* _ *astr*, *infostr*;                                      8h2b
    RETURN(TRUE) END,                                                    8i

                                                                         9

FINISH of label-generator                                                10
```

LABELS1 L10 program for oldnls ... doesn't include mem-list stuff

(J22870) 1-MAY-74 13:28; Title: Author(s): Michael D. Kudlick/MDK;
Distribution: /MDK; Sub-Collections: SRI-ARC; Clerk: MDK;
Origin: <KUDLICK>LABELS1.NLS;7, 28-MAR-74 13:23 MDK ;