RLL 3-Jul-77 12:54 41000

Information on Phone call costs; for the record

Obtained from the phone company.

Information on Phone call costs; for the record

Facts 1

Unit charges 1a

First 80 unit per line free 1a1

$.05 per unit 1a2

Key telephone service - individual lines 1b

Installation - $45/line - 3-4 week notice needed 1b1

Monthly - $7.50/line plus unit charges plus 5% tax 1b2

For more than 5 lines- $5/phone/month plus tax 1b3

$55/phone installation 1b4

additional charges 1b5

lights - $3/line/month 1b5a

$13/line installation 1b5b

intercom - 9 phones (9 numbers) 1b5c

$7.25/month 1b5c1

$48/installation 1b5c2

If no corporate officer has ever had a business phone the $50 deposit per line needed. Returned at the end of the year. 1b6

PBX service - 2 months notice needed 1c

Small switch board with rotary lines and extensions 1c1

Table of costs 1c2

Equipment	Monthly	Installation	Notes	
				1c2a
line each @	3.75	45		1c2b
cabinet	200.15	1000	up to 15 lines + 40 phones	1c2c

Information on Phone call costs; for the record

console	30.55	253	1c2d
truck lines @ needed	5.55	5 one per line	1c2e
circuit packs @	3.55	2 one per 2 stations	1c2f

Contract charge of $1000 is charged, not billed, amortized over 5 years. i.e., no charge if installed for 5 years prorated if remove before. 1c3

WATS 1d

Hourly $	Connect HR:	
		1d1
270	1st 10 hrs	1d2
16.95	10-40	1d3
13.42	40-90	1d4
9.77	90-140	1d5
2.29	140-200	1d6
1.09	200 +	1d7

WATS incoming (minimum of $30 per line) 1e

Hourly $	Connect HR:	
		1e1
14.77	0-30	1e2
13.47	30-60	1e3
11.95	60-100	1e4
10.00	100-150	1e5
3.91	150-220	1e6
2.20	220 +	1e7

Assumptions - n = number of employees 2

	:	:	:	2a

Information on Phone call costs; for the record

Number of lines	10	20	2b
Number of phones	20	30	2c
Home phone lines	10	10	2d
Connect hours - local	375*n	375*n	2e

On the average one call out going per person for about 15 minutes. 23 working days per month gives us 25*0.25*n = 6.25*n hours =6.25*n*60 UNITS = 375 * n UNITS 2e1

Connect hours - WATS	120	120	2f

Installation Table - Number of employees = 30 3

	Cost	L/R/U	Sum	L/R/U	Sum	
						3a
Lines	45	10	450.0	20	900.0	3b
Cabinet	1000	1	1000.0	1	1000.0	3c
Console	253	1	253.0	1	253.0	3d
Truck	5	10	50.0	20	100.0	3e
Circuit packs	2	10	20.0	20	40.0	3f
Itotals			1,773.		2,293.	3f1

Monthly Table 4

	Cost	L/R/U	Sum	L/R/U	Sum	
						4a
Line	3.75	10	37.5	20	75.0	4b
Cabinet	200.15	1	200.15	1	200.15	4c
Console	30.55	1	30.55	1	30.55	4d
Truck	5.55	10	55.5	20	111.0	4e
Packs	3.55	10	35.5	20	71.0	4f
Units	.05	11250	562.5	11250	562.5	4g

Information on Phone call costs; for the record

WATS	11.95	120	1434.0	120 1434.0	4h
Mtotals	:	:	: 2,355.7:	:2,484.:	4h1
Totals:					5

SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL
SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL
SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL

FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER
FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER
FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER

$JAKES41396 $JAKES41396 $JAKES41396 $JAKES41396 $JAKES41396 $JAKES4139
$JAKES41396 $JAKES41396 $JAKES41396 $JAKES41396 $JAKES41396 $JAKES4139
$JAKES41396 $JAKES41396 $JAKES41396 $JAKES41396 $JAKES41396 $JAKES4139

Tuesday, August 2, 1977 21:23:56 Tuesday, August 2, 1977 21:23:56 Tuesday, A
Tuesday, August 2, 1977 21:23:56 Tuesday, August 2, 1977 21:23:56 Tuesday, A
Tuesday, August 2, 1977 21:23:56 Tuesday, August 2, 1977 21:23:56 Tuesday, A

41396

< GJOURNAL, 41396.NLS.1, >, 2-Aug-77 16:40 XXX ;;;; .HJOURNAL="DLS 2-Aug-77 11:04 41396"; Title: .H1="Draft Resume for Duane Stone"; Author(s): Duane L. Stone/DLS; Distribution: /SRI-ARC([INFO-ONLY]) ; Sub-Collections: RADC SRI-ARC; Clerk: DLS; .IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PES; Origin: < STONE, RES2.NLS.1, >, 2-Aug-77 08:30 DLS ;;;; .H=".TABTO=55;Duane L Stone.GCR;RD1 Box 339.GCR;Vernon NY 13476.GCR;.GD;";.LBS=1;.NUMDASH=0;.PN=0;.PES;####;

.PEL; .PN=PN-1; .GCR;this is still a little rough, but I will be on vacation for the next 2 weeks, so wanted to fire it off. use as you feel appropriate.

JOB HISTORY.CENTER; .GCR;

From December 1975 until the Present.

I have supervised the technical activity in the Networks Group in the Interactive Processing Section of the Information Processing Division at Rome Air Development Center (RADC), located on Griffiss Air Force Base in central upstate New York. During this time, I have been:

a project engineer on portions of the NSW project, responsible for formulating a research program to assist the Tactical Air Forces in meeting their Command and Control objectives
and have consulted on the application of NLS to many AF installations.

As project engineer, I have been responsible for the definition, "sales", procurement and technical guidance of the efforts under my perview.

From Aug 1970 through Dec 1975

I worked in the Management Information Sciences Section at RADC, where I was responsible for the planning and conduct of a 5 year program for the assessment and integration of NLS into the Information Sciences Division. This effort included problem definition, planning, sales, acquisition of equipment and services, development of procedures and programs, training, evaluation and assessment. I was responsible for the analysis leading to the installation of the ARPANET node at RADC. I was responsible for the RADC portion of the NSW project and served on the NSW Steering Committee.

I was the technical monitor on all NLS and NSW efforts with SRI-ARC during this period. Although this role is primarilary one of management, I have been involved with a number of application efforts. I have written several "quickie" user programs, I have written one subsystem to automate the routine aspects of RADC correspondence preparation, I have written a help file for that subsystem, I have sheparded a large and complex programming specification manual through all phases of production from input to COM proofing, including the programming necessary to accomplish spelling correction and address list generation and maintenance.

From Jul 1961 through Jul 1970

I worked in the Textual Data Handling Section of the Intelligence Division at RADC. During these years I worked on exploritory and advanced development projects including:

hardware; typeset/typewriter/hand-printreaders (several languages) for automatic input from paper and film media, devices for inputing Chineese characters, combined video and character mask systems for lexical/graphical composition,

special purpose computers for retreiving data from film rolls, film chips, magnetic tape, drums and parallel memories.
software; computer programs and systems for automatic language translation, abstracting, indexing, dissemination, storage and retreival.
requirements analysis and system design; on-line retrieval and dissemination system for the Foreign Technology Division at Wright Patterson AFB OH, on-line document retrieval system for the Deputy Chief of Staff for Plans and Operations at Pentagon DC, batch reporting and retrieval system for the Northeast Communications Agency at Griffiss AFB NY, nation-wide communications system for the Department of Agriculture, wash DC, information management system for the Foreign Disclosure Community at Washington DC,
technique/system evaluation; manual film chip storage and retrieval techniques, human comprehension and retention evaluations of machine vs manual translations and abstracts, relevance/recall evaluations of manual and automatic indexing systems.

SUPPLEMENTARY INFORMATION .CENTER; .GCR;

Education:

Degrees:

BSEE, Clarkson College of Technology, Potsdam NY, 1957-61

Graduate courses:

"Modern Engineering Principles", Syracuse University, 1971
"Man/Environment Factors in System Management", University of Southern California, 1972
"Systems Analysis", University of Southern California, 1972
"Engineering Principles for Aerospace Management", University of Southern California, 1972
"Psychological Factors in System Management", University of Southern California, 1973

Undergraduate courses:

"COBOL Programming", State University of New York, 1970
"Modern Psychology", State University of New York, 1973

Short courses (one-two weeks):

"Solid State Devices", Rome Air Development Center, 1961
"Practical Writing", Rome Air Development Center, 1962
"Information Storage and Retreival", American University, 1962
"Pictorial Data Processing", Yeshciva University, 1963
"Information Systems", Syracuse University, 1965
"Probability and Random Processes", University of Michigan, 1966
"Effective Management", Rome Air Development Center, 1966
"Manager Performance Improvement", Rome Air Development Center, 1970
"Data Communications Systems and Applications", Sylvania, 1971
"Human Relations", Rome Air Development Center, 1974
"Structured Programming", Brandon Applied Systems, 1974

Strong Points:

I enjoy and feel that I am reasonably good at problem analysis, requirements definition, overall system design and planning. I am comfortable with on-line demos and presentations to smaller audiences. I have done this enumerable times, once before the AF Scientific Advisory Board, once before the SADPR-85 group, where I was video taped.

Limitations:
I do not enjoy and am not particularily good at "public speaking", ie, presentations to large audiences in a formal setting, although I have had to do this many times in the government.

Professional Activities:
I am a member of the ACM. I have been a member of the IEEE and the American Society for Information Science in the past.

Other work Experience:
I was an engineering trainee with US Steel during the summer of 1960, where I designed an electronic control system for grinding mills. I also had summer jobs while in college on farms, in a gas station and a food processing plant.

Other Interests:
I enjoy camping, motorcycling, farming and photography when I can find the time.

Age 37 Phone: (315) 829-2285.GCR; Married.GCR; Four children. GCR; Social Security Number: 099-32-3784.GCR; I have a Top Secret clearance..PES;

TECHNICAL PUBLICATIONS:.CENTER;.GCR;

Reports: (most of these were co-authored with others at RADC)

"Investigation of Aids for Manual Unit Record Filing Systems", RADC, Griffiss AFB NY 13441, TM-64-1. March 1964.

"Automatic Data Processing System for the Foreign Disclosure Community", RADC, Griffiss AFB NY 13441, TM-64-13. November 1964.

"Machine Extracting Progress", American Documentation Institute, October 1966.

"An Analysis of FTD Files for Document Reference Retrieval", RADC, Griffiss AFB NY 13441, TM-68-14. September 1967.

"RADC Technical Information Exploitation System (TIES)", RADC, Griffiss AFB NY 13441, TM-70-2. April 1970.

"An Experimental Automatic Extracting Facility", RADC, Griffiss AFB NY 13441, Unpublished April 1970.

"AFXDC Information Retrieval System", RADC, Griffiss AFB NY 13441, TM-69-22. April 1969.

"GEEIA File Study", RADC, Griffiss AFB NY 13441, TM-69-04. April 1970.

Journal References to significant documents
From 1971 to the present, my "publications" have all been via the Journal...to be filled in at a later date.

at a later date.

```
SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL
SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL
SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL   SRI-KL

WROBERTS   WROBERTS   WROBERTS   WROBERTS   WROBERTS   WROBERTS   WROBERTS   WRO
WROBERTS   WROBERTS   WROBERTS   WROBERTS   WROBERTS   WROBERTS   WROBERTS   WRO
WROBERTS   WROBERTS   WROBERTS   WROBERTS   WROBERTS   WROBERTS   WROBERTS   WRO


$WMR$FRED-TEXT   $WMR$FRED-TEXT   $WMR$FRED-TEXT   $WMR$FRED-TEXT   $WMR$FRED-TE
$WMR$FRED-TEXT   $WMR$FRED-TEXT   $WMR$FRED-TEXT   $WMR$FRED-TEXT   $WMR$FRED-TE
$WMR$FRED-TEXT   $WMR$FRED-TEXT   $WMR$FRED-TEXT   $WMR$FRED-TEXT   $WMR$FRED-TE

Tuesday, August 2, 1977 21:29:05   Tuesday, August 2, 1977 21:29:05   Tuesday, A
Tuesday, August 2, 1977 21:29:05   Tuesday, August 2, 1977 21:29:05   Tuesday, A
Tuesday, August 2, 1977 21:29:05   Tuesday, August 2, 1977 21:29:05   Tuesday, A
```

(CONTACT9) 18-Aug-77
By: DAP
Location: phone
Action name: DAP
Action date: 15-Sep-77
Action task: keep in touch
Link:
Documents:
Notes: Just a keep-in-touch call to let Dean know where things stand at ARC and to find out where they are at Xerox. He is in his new position now, and is working on APL-based marketing forecasting models. He feels that the basic problem in our approach to Xerox has been that we are tryint to promote a system packaged as a text processing document production tool when what is needed is the skills and technology to help them to integrate present and future tools. He suggested the possibility of writing a proposal to do such work, and mentioned a specific project to which it might be applicable. I suggested he get in touch with Ken Victor to explore possibilities.

Dean may want Ken and/or me to come to Rochester to make a presentation. I explained to him the difficulties in doing anything like this do to the fact that SRI is no longer supporting NLS marketing. We agreed to keep in touch and do the best we can despite such difficulties.

SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL
SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL
SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL

FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER
FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER
FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER

$JAKE$AD-MASTER-77 $JAKE$AD-MASTER-77 $JAKE$AD-MASTER-77 $JAKE$AD-MASTER-7
$JAKE$AD-MASTER-77 $JAKE$AD-MASTER-77 $JAKE$AD-MASTER-77 $JAKE$AD-MASTER-7
$JAKE$AD-MASTER-77 $JAKE$AD-MASTER-77 $JAKE$AD-MASTER-77 $JAKE$AD-MASTER-7

Thursday, August 18, 1977 20:01:12 Thursday, August 18, 1977 20:01:12 Thursd
Thursday, August 18, 1977 20:01:12 Thursday, August 18, 1977 20:01:12 Thursd
Thursday, August 18, 1977 20:01:12 Thursday, August 18, 1977 20:01:12 Thursd

John Day
RFC 732, NIC 41762 (Sept. 12, 1977)
Obsoletes RFC 731

TELNET Data Entry Terminal Option

1. Command Name and Code:

DET 20

2. Command Meanings

IAC WILL DET

The sender of this command REQUESTS or AGREES to send and receive subcommands to control the Data Entry Terminal.

IAC WONT DET

The sender of this command REFUSES to send and receive subcommands to control the Data Entry Terminal.

IAC DO DET

The sender of this command REQUESTS or AGREES to send and receive subcommands to control the Data Entry Terminal.

IAC DONT DET

The sender of this command REFUSES to send and receive subcommands to control the Data Entry Terminal.

The DET option uses five classes of subcommands: 1) to establish the requirements and capabilities of the application and the terminal, 2) to format the screen, and to control the 3) edit, 4) erasure, and 5) transmission functions. The subcommands that perform these functions are described below.

The Network Virtual Data Entry Terminal (NVDET)

The NVDET consists of a keyboard and a rectangular display. The keyboard is capable of generating all of the characters of the ASCII character set. In addition, the keyboard may possess a number of function keys which when pressed cause a FN subcommand to be sent. (Although most DET's will support one or more peripheral devices such as a paper tape reader or a printer, this

option does not consider their support. Support of peripheral devices should be treated by a separate option).

The screen of the data entry terminal is a rectangle M characters by N lines. The values of M and N are set by negotiating the Output Line Width and Output Page Size options, respectively. The next writing position (x,y) on the screen (where x is the character position and y is the position of the line on the screen) is indicated by a special display character called the cursor. The cursor may be moved to any position on the screen without disturbing any characters already on the screen. Cursor addressing in existing terminals utilizes several topologies and addressing methods. In order to make the burden of implementaton as easy as possible this protocol supports two topologies (the finite plane and the helical torus) and three addressing methods ((x,y); x and y, and relative increments). Since the finite plane with absolute addressing is the least ambiguous and the easiest to translate to and from the others, it is the default scheme used by the NVDET. The torodial form with either relative or absolute addressing is provided for convience.

Also the NVDET provides a mechanism for defining on the screen fields with special attributes. For example, characters entered into these fields may be displayed with brighter intensity, highlighted by reverse video or blinking, or protected from modification by the user. This latter feature is one of the most heavily used for applications where the DET displays a form to be filled out by the user.

The definition of the NVDET uses Telnet option subnegotiations to accomplish all of its functions. Since none of the ASCII characters sent in the data stream have been used to define these functions, the DET option can be used in a "raw" or even "rare" mode. In circumstances where the application program knows what kind of terminal is on the other end, it can send the ASCII characters required to control functions not supported by the option or an implementation. In general keeping all NVDET functions out of the data stream provides better flexibility.

Facility Functions (for detailed semantics see Section 5.)

IAC SB DET <DET facility subcommand><facility map> IAC SE

where <DET facility subcommand> is one 8-bit byte indicating the class of the facilities to be described, and <facility map> is a field of one or two 8-bit bytes containing flags describing the facilities required or desired by the sender.

The bits of the facility maps are numbered from the right starting at zero. Thus, if bit 2 is set the field will have a decimal value of 4. The values of the field are as follows:

Facility cmd: EDIT FACILITIES subcommand code: 1

Facility map:	Bit numbers
Toroidal Cursor Addressing	6
Incremental Cursor Addressing	5
Read Cursor Address	4
Line Insert/Delete	3
Char Insert/Delete	2
Back Tab	1
Positive Addressing only	0

where:

If the Toroidal Cursor Addressing bit is set, the sender requests or provides that the SKIP TO LINE and SKIP TO CHAR subcommands be supported.

If the Incremental Cursor Addressing bit is set, the sender requests or provides that the UP, DOWN, LEFT, and RIGHT subcommands be supported.

If the Read Cursor bit is set, the sender requests or provides the READ CURSOR subcommand.

If the Line Insert/Delete bit is set, the sender requests or provides that the LINE INSERT and LINE DELETE subcommands be supported.

If the Char Insert/Delete bit is set, the sender requests or provides that the CHAR INSERT and CHAR DELETE subcommands be supported.

If the Back Tab bit is set, the sender requests or provides that the BACK TAB subcommand be supported.

If the Positive Addressing bit is set, then the sender is informing the receiver that it can only move the cursor in the positive direction. (NOTE: Terminals that have this property also have a Home function to get back to the beginning.)

Facility cmd: ERASE FACILITIES subcommand code: 2

Facility map: Bit numbers

Erase Field	4
Erase Line	3
Erase Rest of Screen	2
Erase Rest of Line	1
Erase Rest of Field	0

where:

If a bit of the facility map for this facility command is set, the sender requests or provides the facility indicated by the bit. For a more complete description of each of these functions see the Erase Functions section below.

Facility cmd: TRANSMIT FACILITIES subcommand code: 3

Facility map: Bit numbers

Data Transmit	5
Transmit Line	4
Transmit Field	3
Transmit Rest of Screen	2
Transmit Rest of Line	1
Transmit Rest of Field	0

where:

If a bit of the facility map for this facility command is set, the sender requests or provides the facility indicated by the bit. For a more complete description of each of these functions see the Transmit Functions section below.

Facility cmd: FORMAT FACILITIES subcommand code: 4

facility map: Bit numbers

FN	byte 0	7
Modified		6
Light Pen		5
Repeat		4
Blinking		3
Reverse Video		2
Right Justification		1
Overstrike		0

Protection On/Off	byte 1	6
Protection		5
Alphabetic-only Protection		4
Numeric-only Protection		3
Intensity		0-2

where:

If the FN bit is set, the sender requests or provides the FN subcommand.

If the Modified bit is set, the sender requests or provides the ability to indicate fields that are modified and supports the TRANSMIT MODIFIED subcommand.

If the Light Pen bit is set, the sender requests or provides the support of a light pen, including the Pen Selectable attribute of the DATA FORMAT subcommand.

If the Repeat bit is set the sender requests or provides the REPEAT subcommand.

If the Blinking bit is set, the sender requests or provides the ability to highlight a string of characters by causing them to blink.

If the Reverse Video bit is set, the sender requests or provides the ability to highlight a string of characters by "reversing the video image," i.e., if the characters are normally displayed as black characters on a white background, this is reversed to be white characters on a black background, or vice versa.

If the Right Justification bit is set, the sender requests or provides the ability to cause entries of data to be right justified in the field.

If the Overstrike bit is set, the sender requests or provides the ability to superimpose one character over another on the screen much like a hard copy terminal would do if the print mechanism struck the same position on the paper with different characters.

If the Protection On/Off bit is set, the sender requests or provides the ability to turn on and off field protection.

If the Protection bit is set, the sender requests or provides the ability to protect certain strings of characters displayed on the screen from being altered by the user of the terminal. Setting this bit also implies that ERASE UNPROTECTED, DATA TRANSMIT, FIELD SEPARATOR, and TRANSMIT UNPROTECTED subcommands (see below) are supported.

If the Alphabetic-only Protection bit is set, the sender requests or provides the ability to constrain the user of the terminal such that he may only enter alphabetic data into certain areas of the screen.

If the Numeric-only Protection bit is set, the sender requests or provides the ability to constrain the user of the terminal such that he may only enter numerical data into certain areas of the screen.

The three bits of the Intensity field will contain a positive binary integer indicating the number of levels of intensity that the sender requests or provides for displaying the data. The value of the 3 bit field should be interpreted in the following way:

1	one visible intensity
2	two intensities; normal and bright
3	three intensities; off, normal, and bright
>3	>3 intensities; off, and the remaining levels proportioned from dimmest to brightest intensity.

For all of the above commands, if the appropriate bit in <facility map> is not set, then the sender does not request or provide that facility.

Editing Functions

IAC SB DET MOVE CURSOR <x><y> IAC SE subcommand code: 5

where <x> is an 8-bit byte containing a positive binary integer representing the character position of the cursor, <y> is an 8-bit byte containing a positive binary integer representing the line position of the cursor.

This subcommand moves the cursor to the absolute screen address (x,y) with the following boundary conditions:

if x>M-1, set x=M-1 and send an ERROR subcommand

if y>N-1, set y=N-1 and send an ERROR subcommand

This describes a finite plane topology on the screen.

IAC SB DET SKIP TO LINE <y> IAC SE subcommand code: 6

where <y> is a positive 8-bit binary number.

This subcommand moves the cursor to the absolute screen line y. x remains constant. For values of y>N-1

y = y mod N.

IAC SB DET SKIP TO CHAR <x> IAC SE subcommand code: 7

where <x> is a positive 8-bit binary number.

This subcommand moves the cursor to the absolute character position x. y remains constant, unless x>M-1 in which case:

x' = (x mod M)
y' = (y+(x DIV N))
where x' and y' are the new values of the cursor.

These last two subcommands define a toroidal topology on the screen.

IAC SB DET UP IAC SE subcommand code: 8

IAC SB DET DOWN IAC SE subcommand code: 9

IAC SB DET LEFT IAC SE subcommand code: 10

IAC SB DET RIGHT IAC SE subcommand code: 11

These subcommands are provided as a convenience for some terminals. The commands UP, DOWN, LEFT, and RIGHT are defined as

UP: (x,y)=(x, y-1 mod N)
DOWN: (x,y)=(x, y+1 mod N)
LEFT: (x,y)=(x-1, y); if x=0 then x-1 = 0

RIGHT: (x,y)=(x+1 mod M, y) and y = y+1 if x+1>M-1

Note: DOWN, LEFT, and RIGHT cannot always be replaced by the ASCII codes for linefeed, backspace, and space respectively. The latter are format effectors while the former are cursor controls.

IAC SB DET HOME IAC SE subcommand code: 12

This subcommand positions the cursor to (0,0). This is equivalent to a MOVE CURSOR 0,0 or the sequence SKIP TO LINE 0, SKIP TO CHAR 0. This subcommand is provided for convenience, since most terminals have it as a separate control.

IAC SB DET LINE INSERT IAC SE subcommand code: 13

This subcommand inserts a line of spaces between lines y (the current line, determined by the position of the cursor) and line y-1. Lines y through N-2 move down one line, i.e. line y becomes line y+1; y+1 becomes y+2, ...; N-2 becomes N-1. Line N-1 is lost off the bottom of the screen. The position of the cursor remains unchanged.

IAC SB DET LINE DELETE IAC SE subcommand code: 14

This subcommand deletes line y where y is the current line position of the cursor. Lines y+1 through N-1 move up one line, i.e. line y+1 becomes line y; y+2 becomes y+1; ...; N-1 becomes N-2. The N-1st line position is set to all spaces. The cursor position remains unchanged.

IAC SB DET CHAR INSERT IAC SE subcommand code: 15

This subcommand inserts the next character in the data stream between the xth and x-1st characters, where x is the current character position of the cursor. The xth through M-2nd characters on the line are shifted one character positon to the right. The new character is inserted at the vacated xth position. The M-1st character is lost. The position of the cursor remains unchanged.

IAC SB DET CHAR DELETE IAC SE subcommand code: 16

This subcommand deletes the character on the screen at the x-th position. The x-th character is removed and the characters x+1 through M-1 are shifted one character position to the left to become the x-th through M-2nd characters. The M-1st character

position is left empty. (For most terminals it will be set to a NUL or space.) The cursor position remains unchanged.

IAC SB DET READ CURSOR IAC SE subcommand code: 17

This subcommand requests the receiver to send the present position of the cursor to the sender.

IAC SB DET CURSOR POSITION <x><y> IAC SE subcommand code: 18

where <x> and <y> are positive 8-bit binary integers.

This subcommand is sent by a Telnet implementation in response to a READ CURSOR subcommand to convey the coordinates of the cursor to the other side. NOTE: x is less than M and y is less than N.

IAC SB DET REVERSE TAB IAC SE subcommand code: 19

This subcommand causes the cursor to move to the previous tab position. If none exists on the present line, the cursor moves to the previous line and so on until a tab is found or the address (0,0) is encountered. When field protection is in effect the cursor moves to the beginning of the preceding unprotected field.

Transmit Functions (For detailed semantics see Section 5.)

IAC SB DET TRANSMIT SCREEN IAC SE subcommand code: 20

This subcommand causes the terminal to transmit all characters on the screen from position (0,0) to (M-1,N-1). The cursor will be at (0,0) after the operation is complete.

IAC SB DET TRANSMIT UNPROTECTED IAC SE subcommand code: 21

This subcommand causes the terminal to transmit all characters in unprotected fields from position (0,0) to (M-1,N-1). The unprotected fields are separated by the field separator subcommand. The cursor will be at (0,0) or at the beginning of the first unprotected field after the operation is complete.

IAC SB DET TRANSMIT LINE IAC SE subcommand code: 22

This subcommand causes the terminal to transmit all data on the yth line where y is determined by the present position of the cursor. Data is sent from character position (0,y) to the

end-of-line or position (M-1,y) whichever comes first. The cursor position after the transmission is one character position after the end of line condition or the beginning of the next line, (0,y+1).

IAC SB DET TRANSMIT FIELD IAC SE subcommand code: 23

This subcommand causes the terminal to transmit all data in the field presently occupied by the cursor. The cursor position after the operation is complete is one character position after the end of the field or, if that position is protected, at the beginning of the next unprotected field.

IAC SB DET TRANSMIT REST OF SCREEN IAC SE subcommand code: 24

This subcommand causes the terminal to transmit all characters on the screen from position (x,y) to (M-1,N-1) or until the end of text. (x,y) is the current cursor position. The cursor position after the operation is one character position after the last text character, or (0,0) if the last filled character position is (M-1,N-1).

IAC SB DET TRANSMIT REST OF LINE IAC SE subcommand code: 25

This subcommand causes the terminal to transmit all characters on the yth line from position (x,y) to the end of line or (M-1,y) whichever comes first. (x,y) is the current cursor position. The cursor position after the operation is one character position after the last character of the line or the first character of the next line.

IAC SB DET TRANSMIT REST OF FIELD IAC SE subcommand code: 26

This subcommand causes the receiver to transmit the rest of the characters in the field currently occupied by the cursor. The cursor position after the operation is at the beginning of the next field.

IAC SB DET TRANSMIT MODIFIED IAC SE subcommand code: 27

This subcommand causes the receiver to transmit only those fields which have the modified attribute set. The cursor position after the operation is unchanged.

IAC SB DET DATA TRANSMIT <x><y> IAC SE subcommand code: 28

This subcommand is used to preface data sent from the terminal

in response to a user action or a TRANSMIT command. The parameters <x> and <y> indicate the initial position of the cursor. See the Transmit Subcommands subsection in Section 5 for more details. A DATA TRANSMIT subcommand may precede an entire transmission with each field being delineated by the FIELD SEPARATOR subcommand, as would be the case in a response to a TRANSMIT UNPROTECTED, or it may precede each field as would be the case in a response to a TRANSMIT MODIFIED.

Erase Functions

IAC SB DET ERASE SCREEN IAC SE subcommand code: 29

This subcommand causes all characters to be removed from the screen. All fields regardless of their attributes are deleted. The cursor position after the operation will be (0,0). Most terminals set the erased characters to either NUL or space characters.

IAC SB DET ERASE LINE IAC SE subcommand code: 30

This subcommand causes all characters on the yth line to be removed from the screen, where y is the line of the current cursor position. All fields regardless of their attributes are deleted. The cursor position after this operation will be (0,y). NOTE: This operation can be easily simulated by the sequence: LINE DELETE, LINE INSERT. However, the order is important to insure that no data is lost off the bottom of the screen.

IAC SB DET ERASE FIELD IAC SE subcommand code: 31

This subcommand causes all characters in the field occupied by the cursor to be removed. The cursor position after the operation is at the beginning of the field.

IAC SB DET ERASE REST OF SCREEN IAC SE subcommand code: 32

This subcommand causes all characters from position (x,y) to (M-1,N-1) to be removed from the screen. All fields regardless of their attributes are deleted. The cursor position after the operation is unchanged. This is equivalent to doing an ERASE REST OF LINE plus a LINE DELETE for lines greater than y.

IAC SB DET ERASE REST OF LINE IAC SE subcommand code: 33

This subcommand causes all characters from position (x,y) to (M-1,y) to be removed from the screen. All fields regardless of their attributes are deleted. The cursor position after the operation is unchanged.

IAC SB DET ERASE REST OF FIELD IAC SE subcommand code: 34

This subcommand causes all characters from position (x,y) to the end of the current field to be removed from the screen. The cursor position after the operation is unchanged.

IAC SB DET ERASE UNPROTECTED IAC SE subcommand code: 35

This subcommand causes all characters on the screen in unprotected fields to be removed from the screen. The cursor position after the operation is at (0,0) or, if that position is protected, at the beginning of the first unprotected field.

Format Functions

IAC SB DET FORMAT DATA <format map><count> IAC SE
subcommand code: 36

where <format map> is a two-byte field containing the following flags:

Byte 0	
Blinking	7
Reverse Video	6
Right Justification	5
Protection	3-4
Intensity	0-2
Byte 1	
Modified	1
Pen Selectable	0

where:

If the Blinking bit is set, the following field of <count> characters should have the Blinking attribute applied to it by the receiver.

If the Reverse Video bit is set, the following field of <count> characters should be displayed by the receiver with video reversed.

If the Right Justification bit is set, the input entered into the field of <count> characters should be right justified.

The Protection field is two bits wide and may take on the following values:

0	no protection
1	protected
2	alphabetic only
3	numeric only

The protection attribute specifies that the other side may modify any character (no protection), modify no characters (protected), enter only alphabetical characters (A-Z, and a-z) (alphabetic only), or enter only numerical characters (0-9,+,.,and -) (numeric only) in the following field of <count> bytes.

The Intensity field is 3 bits wide and should be interpreted in the following way:

> The values 0-6 should be used as an indication of the relative brightness to be used when displaying the characters in or entered into the following field <count> characters wide. The number of levels of brightness available should have been obtained previously by the Format Facility subcommand. The exact algorithm for mapping these values to the available levels of intensity is left to the implementors. A value of 7 in the intensity field indicates that the brightness should be off, and any characters in or entered into the field should not be displayed.

If the Modified bit is set, the field is considered to have been modified and will be transmitted in response to a TRANSMIT MODIFIED subcommand.

If the Pen Selectable bit is set, the field can be selected with the light pen. NOTE: Use of the light pen should be the subject of another Telnet option.

<count> is 2 bytes that should be interpreted as a positive 16-bit binary integer representing the number of characters following this command which are affected by it.

Data sent to the terminal or the Using Host for unwritten areas of the screen not in the scope of the count should be displayed with the default values of the format map. The default values

are No Blinking, Normal Video, No Justification, No Protection and Normal Intensity. For example, suppose a FORMAT DATA subcommand was sent to the terminal with attributes Blinking and Protected and a count of 5 followed by the string "Name: John Doe". The string "Name:" would be protected and blinking, but the string "John Doe" would not be.

This subcommand is used to format data to be displayed on the screen of the terminal. The <format map> describes the attributes that the field <count> bytes wide should have. This field is to start at the position of the cursor when the command is acted upon. The next <count> displayable characters in the data stream are used to fill the field. Subsequent REPEAT subcommands may be used to specify the contents of this field. If the sender specifies attributes that have not been agreed upon by the use of the Format Facility subcommand, the Telnet process should send an Error Subcommand to the sender, but format the screen as if the bit had not been set.

IAC SB DET REPEAT <count><char> IAC SE subcommand code: 37

where <count> is a positive 8-bit binary integer. <char> is an 8-bit byte containing an ASCII character.

This subcommand is used to perform data compression on data being transferred to the terminal by encoding strings of identical characters as the character and a count. The repeated characters may be part of a field specified.

IAC SB DET SUPPRESS PROTECTION <negotiation> IAC SE
subcommand code: 38

where <negotiation> may have the values of the Telnet option negotiation:

251	WILL
252	WONT
253	DO
254	DONT

This subcommand is used to suppress the field protection in a non-destructive manner. Many data entry terminals provide the means by which protection may be turned on and off without modifying the contents of the screen or the terminal's memory. Thus, the protection may be turned off and back on without retransmitting the form.

The default setting of the option is that protection is on, in other words

IAC SB DET SUPPRESS PROTECTION WONT IAC SE
IAC SB DET SUPPRESS PROTECTION DONT IAC SE

Negotiation of this subcommand follows the same rules as negotiations of the Telnet options.

IAC SB DET FIELD SEPARATOR IAC SE subcommand code: 39

It is necessary when transmitting only the unprotected portion of the screen to provide a means for delimiting the fields. Existing DETs use a variety of ASCII characters such as Tab, Group Separator, Unit Separator, etc. In order to maintain transparency of the NVDET this subcommand is used to separate the fields. Clearly, this incurs rather high overhead. This overhead can be avoided by using the Byte Macro Option (see Appendix 3).

Miscellaneous Commands

IAC SB DET FN <code> IAC SE subcommand code: 40

where: <code> is one byte.

Many data-entry terminals provide a set of "function" keys which when pressed send a one-character command to the server. This subcommand describes such a facility. The values of the <code> field are defined by the user and server. The option merely provides the means to transfer the information.

IAC SB DET ERROR <cmd> <error code> IAC SE subcommand code: 41

where:

<cmd> is a byte containing the subcommand code of the subcommand in error.

<error code> is a byte containing an error code.

(For a list of the defined error codes see Appendix 2.)

This subcommand is provided to allow DET option implementations to report errors they detect to the corresponding Telnet process. At this point it is worth reiterating that the philosophy of this option is that when an error is detected it

should be reported; however, the implementation should attempt its best effort to carry out the intent of the subcommand or data in error.

3. Default and Minimal Implementation Specifications

Default

WON'T DET -- DON'T DET

Neither host wishes to use the Data Entry Terminal option.

Minimal Implementation

DET EDIT FACILITIES
DET ERASE FACILITIES
DET TRANSMIT FACILITIES
DET FORMAT FACILITIES
DET MOVE CURSOR <x><y>
DET HOME
DET ERASE SCREEN
DET TRANSMIT SCREEN
DET FORMAT DATA
DET ERROR <cmd> <error code>

In the case of formatting the data, the minimal implementation should be able to support a low and high level of intensity and protection for all or no characters in a field. These functions, however, are not required.

The minimal implementation also requires that the Output Line Width and Output Page Size Telnet options be supported.

4. Motivation

The Telnet protocol was originally designed to provide a means for scroll-mode terminals, such as the standard teletype, to communicate with processes through the network. This was suitable for the vast majority of terminals and users at that time. However, as use of the network has increased into other areas, especially areas where the network is considered to provide a production environment for other work, the desires and requirements of the user community have changed. Therefore, it is necessary to consider supporting facilities that were not initially supported. This Telnet option attempts to do that for applications that require data entry terminals.

This option in effect defines the Network Virtual Data Entry Terminal. Although the description of this option is quite long, this does not imply that the Telnet protocol is a poor vehicle for this facility. Data Entry Terminals are rather complex and varied in their abilities. This option attempts to support both the minimal set of useful functions that are either common to all or can be easily simulated and the more sophisticated functions supplied in some terminals.

Unlike most real data entry terminals where the terminal functions are encoded into one or more characters of the native character set, this option performs all such controls within the Telnet subnegotiation mechanism. This allows programs that are intimately familiar with the kind of terminal they are communicating with to send commands that may not be supported by either the option or the implementation. In other words, it is possible to operate in a "raw" or at least "rare" mode using as much of the option as necessary.

Although many data entry terminals support a variety of peripheral devices such as printers, cassettes, etc. it is beyond the scope of this option to entertain such considerations. A separate option should be defined to handle this aspect of these devices.

5. Description

General Notes

All implementations of this option are required to support a certain minimal set of the subcommands for this option. Section 3 contains a complete list of the subcommands in this minimal set. In keeping with the Telnet protocol philosophy that an implementation should not have to be able to parse commands it does not implement, every subcommand of this option is either in the minimal set or is covered by one of the facility subcommands. An implementation must "negotiate" with its correspondent for permission to use subcommands not in the minimal set before using them. For details of this negotiation process see the section below on facility subcommands.

Most data entry terminals are used in a half duplex mode. (Although most DET's on the market can be used either as data entry terminals or as standard interactive terminals, we are only concerned here with their use as DET's.) When this option is used, it is suggested that the following Telnet options be refused: Echo, Remote Controlled Transmission and Echoing, and Suppress Go-Ahead. However, this option could be used to support a simple full duplex CRT based application using the basic cursor control functions provided here. For these cases, one or more of the above list of options might be required. (Support of sophisticated interactive calligraphic applications is beyond the scope of this option and should be done by another option or the Network Graphics Protocol.)

In RFC 728, it was noted that a synch sequence can cause undesired interactions between Telnet Control functions and the data stream. A synch sequence causes data, but not control functions, to be flushed. If a control function which has an effect on the data immediately following it is present in the data stream when a synch sequence occurs, the control function will have its effect, not on the intended data, but on the data immediately following the Data Mark. These DET subcommands are susceptible to this pitfall:

CHAR INSERT
DATA TRANSMIT
FORMAT DATA

The undesired interactions are best avoided by the receiver of the synch sequence deleting these subcommands and all data associated with them before continuing to process the control functions.

This implies that the Data Mark should not occur in the middle of the data associated with these subcommands.

Facility Subcommands

These four subcommands are used by the User and Server implementations to negotiate the subcommands and attributes of the terminal that may be utilized. This negotiation can be viewed as the terminal (User Host) indicating what facilities are provided and the Server Host (or application program) indicating what facilities are desired.

When Sent: A Server Telnet implementation using the DET option must send a facility subcommand requesting the use of a particular subcommand or terminal attribute not in the minimal implementation before the first use of that subcommand or attribute. The User Telnet implementation should respond as quickly as possible with its reply. Neither the User nor Server are required to negotiate one subcommand at a time. Also, a Telnet implementation responding to a facility subcommand is not required to give permission only for that subcommand. It may send a format map indicating all facilities of that class which it supports. However, a Telnet implementation requesting facilities must send a facility subcommand before its first use of the subcommand regardless of whether earlier negotiations have indicated the facility is provided. The facility cannot be used until a corresponding facility subcommand has been received. There are no other constraints on when the facility subcommands may be sent. In particular, it is not necessary for an application to know at the beginning of a session all facilities that it will use.

Action When Recieved: There are two possible actions that may be taken when a facility subcommand is received depending on whether the receiver is a requestor or a provider (User).

Requestor: When a facility subcommand is received by a requestor and it is in the state of Waiting for a Reply, it should go into the state of Not Waiting. It should then take the facility map it had sent and form the logical intersection with the facility map received. (For the Intensity attribute, one should take the minimum of the number received and the number requested.) The result indicates the facilities successfully negotiated. NOTE: if the receiver is not in the Waiting for Reply state, then this is the provider case described next.

Provider: When a facility subcommand is received, it should send a facility subcommand with a facility map of the facilities it

provides as soon as possible. It should then determine what new facilities it is providing for the Requestor by forming the logical intersection of the facility map received and the one sent.

NOTE: Although in most cases the requestor will be the Server Host and the provider will be the User Host supporting the terminal, this distinction may not always be true.

Transmit Subcommands

There are two kinds of transmit subcommands: those used to request that data be sent to the requestor, and one to preface data sent to the requestor. The first kind allow the requestor to control when, from where and to some degree how much data is transmitted from the terminal. Their explanation is straightforward and may be found in Section 2.

Data may be sent from the terminal as a result of two events: the user of the terminal caused the transmission or in response to a transmit subcommand. Some programs may wish to know from where on the screen the transmission began. (This is reasonable, since the terminal user may move the cursor around considerably before transmitting.) Other programs may not need such information. The DATA TRANSMIT subcommand is provided in case this function is needed. When used this subcommand prefaces data coming from the terminal. The parameters <x> and <y> give the screen coordinates of the beginning of the transmission. <x> must be less than or equal to M-1 and <y> must be less than or equal to N-1. It is assumed that all data between this DATA TRANSMIT and the next one starts at the coordinates given by the first subcommand and continues filling each line thereafter according to the constraints of the screen and the format effectors in the data. Thus an intelligent or sloppy user-host DET implementation (depending on your point of view) need only include a DATA TRANSMIT subcommand when the new starting point is different from the last ending point.

6. Sample Interaction

The nomenclature of RFC 726 will be used to describe this example. To quote that RFC:

"S:" is sent from serving host to using host.
"U:" is sent from using host to serving host.
"T:" is entered by the terminal user.
"P:" is printed on the terminal.

Text surrounded by square brackets ([]) is commentary. Text surrounded by angle brackets (<>) is to be taken as a single unit, e.g, carriage return is <cr>, and the decimal value 27 is represented <27>.

We assume that the user has established the Telnet connection, logged on, and an application program has just been started either by the user directly or through a canned start up procedure. The presentation on the page is meant to merely group entities together and does not imply the position of message boundaries. One should assume that any part of the dialogue may be sent as one or many messages. The first action of the program or Telnet is to negotiate the DET option:

S: <IAC><DO><DET>

U: <IAC><WILL><DET>

S:<IAC><DO><OUTPUT PAGE SIZE> [First negotiate the screen size. In this case we are asking the user the size of the terminal. This could have been done before the DET option was negotiated.]

U:<IAC><WILL><NAOP>

U:<IAC><SB><NAOP><DR><25><IAC><SE>

S:<IAC><SB><NAOP><DS><0><IAC><SE>

S:<IAC><DO><OUTPUT LINE WIDTH>

```
U:<IAC><SB><NAOL><DR><80><IAC><SE>          [Defines the screen to be
                                            25 lines by 80 characters.
                                            The server may use this
                                            information when formatting
                                            the screen.]

S:<IAC><SB><NAOL><DS><0><IAC><SE>

S:<IAC><SB><DET><FORMAT FACILITIES>
<Repeat><Protection, 3 Levels
Intensity><IAC><SE>                         [Now set the terminal
                                            attributes.]

U:<IAC><SB><DET><FORMAT FACILITIES>
<Repeat, Blinking><Protection, 3
Levels Intensity><IAC><SE>

S:<IAC><SB><DET><ERASE SCREEN><IAC><SE>     [Erase the screen and start
                                            sending the form.]

   <IAC><SB><DET><FORMAT DATA>
   <Protection=1, Intensity=1><0>
   <5><IAC><SE>Name:

   <IAC><SB><DET><MOVE CURSOR><0><1><IAC><SE>

   <IAC><SB><DET><FORMAT DATA>
   <Protection=1, Intensity=1><0>
   <8><IAC><SE>Address:

   <IAC><SB><MOVE CURSOR><0><4><IAC><SE>

   <IAC><SB><DET><FORMAT DATA>
   <Protection=1, Intensity=1><0>
   <17><IAC><SE>Telephone number:

   <IAC><SB><DET><MOVE CURSOR><32><4><IAC><SE>

   <IAC><SB><DET><FORMAT DATA>
   <Protection=1, Intensity=1><0>
   <24><IAC><SE>Social Security Number:

   <IAC><SB><DET><FORMAT DATA>
   <Protection=1, Intensity=7>
   <0><11><IAC><SE>                         [Establish a field that
                                            doesn't display what is
                                            typed into it.]
```

```
   <IAC><SB><DET><MOVE CURSOR><32><5><IAC><SE>

   <IAC><SB><DET><FORMAT FACILITIES>
   <Blinking><0><IAC><SE>                    [Get permission to use
                                             Blinking Attribute.]

U:<IAC><SB><DET><FORMAT FACILITIES>
<Repeat, Blinking><Protection,
3 Levels Intensity><IAC><SE>

S:<IAC><SB><DET><FORMAT DATA>
<Blinking=1, Protection=1,
Intensity=1><0><29><IAC><SE>

   Your SSN will not be printed.

   <IAC><SB><DET><HOME><IAC><SE>
   <IAC><GA>
```

The previous exchange has placed a form on the screen that looks like:

```
Name:
Address:
Telephone Number:                        Social Security Number:
                                  "Your SSN will not be printed."
```

where the quoted string is blinking.

The terminal user is now free to fill in the form provided. He positions the cursor at the beginning of the first field (this usually is done by hitting the tab key) and begins typing. We do not show this interaction since it does not generate any interaction with the User Telnet program or the network. After the terminal user has completed filling in the form, he strikes the transmit key to send the unprotected part of the form, but first the User Telnet program negotiates the Byte Macro Option to condense the Field Separator subcommand:

```
U:<IAC><DO><BM>                          [Negotiate Byte
                                         MacroOption.]

S:<IAC><WILL><BM>                        [Define decimal 166 to be
                                         the Field Separator
                                         subcommand (see Appendix
                                         3)]
```

U:<IAC><SB><BM><DEFINE>
<166><6><IAC SB DET FIELD
SEPARATOR IAC SE><IAC><SE>

S:<IAC><SB><BM><ACCEPT><166><IAC><SE> [The server accepts the macro.]

U:<IAC><SB><DET><DATA TRANSMIT><0><6><IAC><SE>
John Doe <166> 1515 Elm St., Urbana, Il 61801
<166> 217-333-9999 <166> 123-45-6789 <166>

S:<IAC><SB><DET><ERASE SCREEN><IAC><SE>
Thank you.

And so on.

Appendix 1 - Subcommands, opcodes and syntax

1	EDIT FACILITIES	<Facilty map>
2	ERASE FACILITIES	<Facility map>
3	TRANSMIT FACILITIES	<Facility map>
4	FORMAT FACILITIES	<Facility map 1> <Facility map 2>
5	MOVE CURSOR	<x> <y>
6	SKIP TO LINE	<y>
7	SKIP TO CHAR	<x>
8	UP	
9	DOWN	
10	LEFT	
11	RIGHT	
12	HOME	
13	LINE INSERT	
14	LINE DELETE	
15	CHAR INSERT	
16	CHAR DELETE	
17	READ CURSOR	
18	CURSOR POSITION	<x><y>
19	REVERSE TAB	
20	TRANSMIT SCREEN	
21	TRANSMIT UNPROTECTED	
22	TRANSMIT LINE	
23	TRANSMIT FIELD	
24	TRANSMIT REST OF SCREEN	
25	TRANSMIT REST OF LINE	
26	TRANSMIT REST OF FIELD	
27	TRANSMIT MODIFIED	
28	DATA TRANSMIT <x><y>	
29	ERASE SCREEN	
30	ERASE LINE	
31	ERASE FIELD	
32	ERASE REST OF SCREEN	
33	ERASE REST OF LINE	
34	ERASE REST OF FIELD	
35	ERASE UNPROTECTED	
36	FORMAT DATA <format map>	
37	REPEAT <count><char>	
38	SUPPRESS PROTECTION <negotiation>	
39	FIELD SEPARATOR	
40	FN <code>	
41	ERROR <cmd><error code>	

Appendix 2 - Error Codes

1	Facility not previously negotiated.
2	Illegal subcommand code.
3	Cursor Address Out of Bounds.
4	Undefined FN value.
5	Can't negotiate acceptable line width.
6	Can't negotiate acceptable page length.
7	Illegal parameter in subcommand.
8	Syntax error in parsing subcommand.
9	Too many parameters in subcommand.
10	Too few parameters in subcommand.
11	Undefined parameter value
12	Unsupported combination of Format Attributes

Appendix 3 - Use of the Byte Macro Option

One of the major drawbacks of the DET option is that because the functions are encoded as Telnet option subnegotiations a fairly high overhead is incurred. A function like Character Insert which is encoded as a single byte in most terminals requires six bytes in the DET option. Originally the only other solution that would have accomplished the same transparency that the use of subcommands provides would have been to define additional Telnet control functions. However, since this would entail modification of the Telnet protocol itself, it was felt that this was not a wise solution. Since then the Telnet Byte Macro Option (RFC 729) has been defined. This option allows the user and server Telnets to map an arbitrary character string into a single byte which is then transferred over the net. Thus the Byte Macro Option provides the means for implementations to avoid the overhead for heavily used subcommands. The rest of this appendix suggests how the Byte Macro Option should be applied to the DET option.

In keeping with the specification of the Byte Macro Option, macro bytes will be chosen from the range 128 to 239. For the DET option, it is suggested that macro bytes be chosen by adding the subcommand code to 128. In addition, an unofficial DET subcommand might be defined indicating that each side was willing to support macro bytes for all subcommands (but not necessarily support all of the subcommands themselves) according to this algorithm. This subcommand would be:

IAC SB DET DET-MACRO <negotiation> IAC SE subcommand code: 254

where <negotiation> may have the values of the Telnet option negotiation:

251	WILL
252	WONT
253	DO
254	DONT

This subcommand is sent by a Telnet implementation to indicate its willingness to adopt byte macros for all of the DET subcommands according to the following algorithm:

The macro byte for subcommand i will be i+128 and will represent the following string for parameterless subcommands:

IAC SB DET <subcommand code> IAC SE

and the following string for subcommands with parameters:

IAC SB DET <subcommand code>

The default setting for this subcommand is that the macros are not in effect, in other words,

IAC SB DET DET-MACRO WONT IAC SE
IAC SB DET DET-MACRO DONT IAC SE

Negotiation of this subcommand follows the same rules as negotiations of the Telnet options.

References

1. ADM-1 Interactive Display Terminal Operator's Handbook
Lear-Siegler, Inc. 7410-31.

2. ADM-Interactive Display Terminal Operator's Handbook
Lear-Siegler, Inc. EID, 1974.

3. Burroughs TD 700/800 Reference Manual, Burroughs Corp., 1973

4. Burroughs TD 820 Reference Manual, Burroughs Corp. 1975.

5. CC-40 Communications Station: General Information Manual.
Computer Communication, Inc. Pub. No. MI-1100. 1974.

6. Crocker, David. "Telnet Byte Macro Option," RFC 729, 1977.

7. Data Entry Virtual Terminal Protocol for Euronet, DRAFT, 1977.

8. Day, John. "A Minor Pitfall in the Telnet Protocol," RFC 728,
1977.

9. Hazeltine 2000 Desk Top Display Operating Instructions.
Hazeltine IB-1866A, 1870.

10. How to Use the Consul 980: A Terminal Operator's Guide and
Interface Manual. Applied Digital Data Systems, Inc. 98-3000.

11. How to Use the Consul 520: A Terminal Operator's Guide and
Interface Manual. Applied Digital Data Systems, Inc. 52-3000.

12. Honeywell 7700 Series Visual Information Projection (VIP)
Systems: Preliminary Edition. 1973.

13. An Introduction to the IBM 3270 Information Display System. IBM
GA27-2739-4. 1973.

14. Naffah, N. "Protocole Appareil Virtuel type Ecran" Reseau
Cyclades. TER 536. 1976.

15. Postel, Jon and Crocker, David. "Remote Controlled Transmission
and Echoing Telnet Option", RFC 726 NIC 39237, Mar. 1977.

16. Schicker, Peter. "Virtual Terminal Protocol (Proposal 2). INWG
Protocol Note #32., 1976.

17. UNISCOPE Display Terminal : Programmer Reference . Sperry-Univac UP-7807 Rev. 2, 1975.

18. Universal Terminal System 400: System Description. Sperry-Univac UP-8357, 1976.

19. Walden, David C. "Telnet Output Line Width Option." NIC # 20196, 1973, also in ARPANET Protocol Handbook, 1976.

20. Walden, David C. "Telnet Output Page Size" NIC # 20197, 1973, also in ARPANET Protocol Handbook, 1976.

SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL
SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL
SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL SRI-KL

FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER
FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER
FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER FEINLER

$JAKE$41769 $JAKE$41769 $JAKE$41769 $JAKE$41769 $JAKE$41769 $JAKE$417
$JAKE$41769 $JAKE$41769 $JAKE$41769 $JAKE$41769 $JAKE$41769 $JAKE$417
$JAKE$41769 $JAKE$41769 $JAKE$41769 $JAKE$41769 $JAKE$41769 $JAKE$417

Wednesday, September 14, 1977 17:30:26 Wednesday, September 14, 1977 17:30:26
Wednesday, September 14, 1977 17:30:26 Wednesday, September 14, 1977 17:30:26
Wednesday, September 14, 1977 17:30:26 Wednesday, September 14, 1977 17:30:26

< GJOURNAL, 41769.NLS.1, >, 13-Sep-77 10:44 XXX ;;;; Title:
Author(s): James H. Bair/JHB; Distribution: /IFIP-WG63([INFO-ONLY])
IFIP-D([INFO-ONLY]) ; Sub-Collections: SRI-ARC IFIP-WG63 IFIP-D;
Clerk: JHB; Origin: < BAIR, POUZIN-REPORT.NLS.4, >, 13-Sep-77
10:11 JHB ;;;; ####;

Meeting: Working Group Meeting at the IFIP Annual Congress, Toronto, August 10

Attendees: J. Carlisle, R. Uhlig, G. Edwards, K. Samuelson, S. Rolhfs, M. Tyler (for R. Pye), L. Day, J. Bair -- chairperson; Guests: L. Nauges, M. Seider, S. Winkler.

Format of this report: Conclusions and plans will be listed, including items not specifically discussed at the meeting but which are part of the activities to be reported to TC6.

Activities

Panel Sessions:

Three panel sessions were sponsored at the congress which were well attended. The impact seems to have been that desired by the WG: to raise issues and increase the consciousness of the data processing community about the design and impact of application systems from the user's perspective. The WG has sponsored a number of panels to date, which have, on the whole, effectively supported the group's aims. A report on the IFIP '77 panels is attached (Annex A).

Thanks to Gwen Edwards for coordinating and arranging the 2 panels on electronic mail, and to Gwen, Chris Evans, and Robert Taylor for chairing the sessions.

Previous meeting:

The WG meeting in Berlin in June, 1977, considered the important issue of group scope as the basis of dialogue about the issues in human-computer communication. The majority of those in attendance recommended that the Group's scope be expanded to include non-computer based technology. Information technology developments will result in an increasingly blurred line between digital and non-digital technologies. The real concern is system design for optimum communication between the user and the machine, which may be video or some yet undiscovered device.

This recommendation has implications for the group title and the group's relationship to IFIP structure. No action is planned at this time except to continue dialogue to promote understanding of the alternative perspectives within the group. The data processing community presently feels well defined and it may be some years before the this issue becomes of general concern.

Other discussion concerned the IFIP panels and plans for WG activities. The panel at the ICA Congress in Berlin went well, but a larger attendance was expected. The Communication Sciences disciplines do not appear ready to address communication problems between computers and humans even though I was invited to put on the panel.

NSF Proposal, "Operational Trials of Electronic Information Exchange"

The proposal was submitted to NSF in July to use the EIES teleconferencing system (by Murry Turoff) to support the group's interaction for one year. Focus would be on two efforts, the assessment of EIES and a report on the issues, hypotheses, and principles of human-computer communication. The assessment would be a report to NSF, and the report would be to IFIP for publication through North-Holland. 12 group members have agreed to participate, although foreign participation through TELENET may be hindered by the PTTs.

The submission of this proposal, which has passed the first stages of review at NSF, is a significant achievment for WG 6.3 even if it is not funded.

Plans

ICCC

It has been proposed that ICCC become an affiliate member of IFIP. Thus, a working relationship is being established at higher levels of IFIP. Stan Winkler's welcome attendance at our Toronto meeting was in an effort to collaborate, specifically on the next two items below.

Panel session at ICCC 1978

Ron Uhlig has volunteered to put together an invited panel session on interactive, computer-based, office support systems. Correspondance is ongoing with Prof. Inose, ICCC Program Chairman, and is encouraging to date. Given WG 6.3's charter, the session should focus upon the human-computer communication factors in interactive, computer-based, office support systems.

Tutorial on the Office of the Future

A proposal is being submitted to the ICCC Executive Council Meeting, 8 Sept. 77, by Ron Uhlig (representing WG 6.3) and Louis Pouzin to run a 2 day tutorial entitled, "The Office of the Future: Technology Applications and Impacts." This will sponsored by IFIP WG 6.3. A copy of the proposal is attached (Annex B).

ACTION item: Contact the TC6 representative from Japan with a copy of the proposal and outline of the tutorial before the Oct. TC6 meeting.

The potential value to IFIP is proposed to be proceedings, published through North-Holland, the royalities going to IFIP. It seems that this may require some financial support from IFIP, both through the WG 6.3 budget and possibly through a short term loan to cover preparation expenses.

Other Possibilities for the Tutorial: The tutorial is ideal to present in conjunction with many professional conferences in the computer and related fields. The next most likely conference will be the Teleinformatics Conference sponsored by TC 6 in Paris,

1979. Other conferences include those of AFIPS (NCC), ASIS, BCS (Britisn Computer Society), SICOB (French Computer Society) and so on. The tutorial could conceivably stand alone if sponsored by a business or other institution (e.g. IASA, Communication Studies & Planning, Ltd .(London), SOFTLAB & German Research Dept. (Munich), U. of Stockholm).

Revenue production necessary

An important IFIP guideline was announced at the meeting: the group's activities should bring in revenue for IFIP. One of IFIP's major sources of funds is the sale of publications through North-Holland. Our policy must be to plan to produce publishable material as has WG 6.1. Ideas and volunteer organizers are welcome.

Teleinformatics Conference, Paris, 1979

TC 6 has planned to launch a major conference in the general area of digital technology applications. Louis Pouzin has been selected to be program chairman. Of particular interest to WG 6.3 is that strong emphasis will be placed upon the users' needs and design requirements.

I hereby offer WG 6.3's services in developing the program, refereeing papers, getting speakers, and the like. It appears very appropriate since the group's emphasis is similar to what has been proposed.

Publish transcription of IFIP '77 Panels

Tapes were made by N. Leduc of Bell-Canada (colleague of Gwen Edwards) of the WG 6.3 sponsored panels. S. Rohlfs has volunteered to have the tapes transcribed if they can be provided to her. Speakers would then have to edit the transcripts for publication through IFIP.

WG 6.3 Meetings

Next presently scheduled meeting: NCC, Anahiem, Calif., June 1978

Possibilities:

ICCC in Kyoto, Japan, Sept. 1978.

Moscow: We have an invitation to organize a meeting in Eastern Europe or USSR, particularly Moscow, from A. Butrimenko, currently at IASA. The problems might be prohibitive, however, it's worth considering.

Six Year Plan needed:

Other ideas for a 6 year plan for IFIP are needed and hereby solicited from WG 6.3 members. It is important for members, particularly those who have not been active recently, to submit ideas that they then could actively support. Ron's leadership of

the ICCC Panel and tutorial is an example..PES;

Annex A: .SN=Off;"Report on the IFIP '77 Panels Sponsored by the Human-Computer Communications Working Group" .IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PBS; .YBS=1;.H="SRI ARC No. 41534.Split; Bair, IFIP '77 Panels Report";

.PN=1; .GCR;This is a brief summary (about 2 pp) of the panels I sponsored at the IFIP Congress in Toronto last week.

.Center=4;.LM=5;.PN=1;Report on the IFIP '77 Panels Sponsored by the Human-Computer Communications Working Group
by Jim Bair, Chairman
(Stanford Research Institute, Menlo Park, CA 94025)

Earlier this year, I reported that three panel sessions were planned for the IFIP Congress in Toronto this August. This is a brief overview of the sessions, each attended by over 400 persons. Our working group is planning to publish a transcript of the sessions if the tapes are useable (we will announce the availability.)

"The Communication with Naive Computer Users," chaired by Chris Evans (NPL, England), focused on four applications areas. The first, computer-based instruction, was appropriately discussed by D. Bitzer, the father of PLATO. He reviewed the basic design strategy of PLATO, emphasizing the large user population that can be effectively reached through centrally based computer graphics. The main point was that graphics displays, emphasizing pictures over text, have proved to be the most effective means of communication with naive users. Included in PLATO is a dynamic modelling capability that can immediately reflect in a chart or graph changes in variables specified by the user, e.g., the effect of population density on transportion systems. Another point was the usefulness of synchronous, graphic communication, particularly for online assistance to users. PLATO users can "send" graphics and simultaneously share a graphics image. The addition of message switching effectively links users in an online learning community.

The second application discussed was a fixed-format, form-oriented system (e.g. airline reservation), by S. Rohlfs (SOFTLAB, Munich). Among the important points raised were the semantic communication problems of the computer command language and concepts and the difficulty of handling non-verbal components of communication through computer interaction. Basically, there is a loss of the emotional context which helps establish meaning in face-to-face communication. Although the problem is not well enough understood for a solution, care can be taken by designers to ensure linguistic consistency and mutually exclusive definitions, avoiding synonyms and homonyms.

The third application area, office information systems, dealt with the problem of using the many application systems available to an office worker or anyone who works with knowledge as his job. I presented some of the problems and solutions, and the problems

with the solutions. The basic problems centered on the fact that there is an extremely heterogeneous and diverse spectrum of users, services, networks, hardware, and software that would be used for most of the working day. This heavy use of such a variety of systems is compounded by the rapid change of technology and new forms of organization all being transferred to the user in a huge innovation change. The solution is an integrated approach that uses one interface to all services. This interface has one command language and one display presentation format, and is heavily supported by online and personal assistance. This approach to the computer support of all information sevices also has some tradeoffs, such as complexity, cost, and the need for protocols between the interface and the different systems.

The fourth area represented the largest application clientele, the TV viewing public. M. Tyler (London) discussed the information delivery media being tested in England for home use. TV usage for news will lead to computer applications that require careful attention to the display complexity, ease of use, and information searching methods, such as tree search. Of greatest concern is that these systems be evaluated in terms of their impact upon the user and the organization.

Questions from the audience raised the need for more definitive and prescriptive information about interface design and questions about the relationship of such applications to personal power.

The panel entitled, "Electronic Mail I -- Message System Designers," chaired by R. Taylor (Xerox Research) was remarkably consistent across the three speakers, A. Vezza (MIT), R. Stoltz (USC-ISI), and T. Meyer (BBN). They basically agreed on the requirements, problem areas, features, and implications.

Message systems must be flexible, evolutionary, adaptive to individual users, responsive, include user level support, and be based on user behavior. Problem areas include making the interface transparent to the user, authenticating authorship (computer signatures?), retrieval (heavy emphasis was placed upon managing the message data base), security, and inconsistency between different systems. Several features were elaborated in detail, including distribution lists, header information (author, i.d., date/time, location, distribution list, etc.), mulit-host delivery, automatic forwarding, and so on. It was generally agreed that message systems will have implications for the organization, changing the way it operates and providing an organizational data base--in essence, office automation. Since the message systems are evolving to encompass most office activities, models of the organization and application-level research are needed. In an echo of the first panel, the problems of message channel "bandwidth" were cited, resulting in the need to support pictorial and voice communication.

The designer's panel was followed by, "Electronic Mail II, Message System Users," a sequence plotted to spark some controversy. However, the panelists did not follow the panel chairperson's lead (G. Edwards, Bell Canada), which addressed specific problems with

message system implementation, e.g. ,the extremely complex message headers.

JCR Licklider (MIT) did not claim symbiosis for message systems, but presented some ideas that could promote it. He noted that the user's model should correspond to the actual system, and I suppose that he meant the system should be designed from the user's model and not the reverse. In addition to the issues raised by the designers, there are two problems which are critical: quantity of messages and ease of use. In many cases, there are so many messages (many feet of paper printout) that the advantage of quick response is lost. System learnability, a function of ease of use, must be incremental. The apt metaphor, "even the longest journal begins with a single step," drove home the point. The overall need to evaluate systems in terms of user needs and impacts was reiterated.

A unique application area, message systems for the medical professional, was presented by K. Samuelson (U. of Stockholm). Message systems are ideal to bring together distributed specialists and support staff in a team approach to a medical situation. Centers, formed by general practitioners, could bring together specialists through a combination of telemetry, video, and text transmission. The multimedia approach would not always be necessary--a large portion of patients are worried rather than "sick," requiring consultation only. In these cases, the medium is the cure as well as the message.

R. Uhlig (US Army) and J. Carlisle (USC) discussed message system value and impact on an organization from two perspectives, that of the manager and the researcher respectively. Experiences with a distributed, intraorganization user group of over 300 persons provided excellent evidence of the potential advantages of message systems. The list of advantages is too long to present here. Overall, the effects appeared to be an improved corporate awareness due to the increased "connectivity" possible among personnel. In spite of the potential to increase organizational effectiveness and promote human relations, message systems were NOT seen as a replacement for conventional communication, but as a new medium that was easy to use and accepted by all levels of staff.

The critical need for research on the impacts of message systems was reiterated. One research question was about the effect on psychological inhibitions usually present in face-to-face interaction and the subsequent changes in etiquette and human-to-human interface. A large effort has already been made to provide the features necessary to fill user requirements, and the features noted in this presentation echoed those of the system designers. Thus, it is now time to invest in research to understand impacts upon sociology, well being, and productivity.

Uhlig and Carlisle echoed a theme recurring throughout the conference whenever applications were discussed: current message systems are just the tip of the iceberg of "office automation" possibilities. Indeed, office automation is careening into a

multi-billion dollar business, and this is occurring before we have an adequte understanding of the effects.

+International Federation for Information Processing

.PBS;.PN=1;.H=""; ANNEX B: Plan for tutorial -- "The Office of the Future: Technology Applications & Impacts".lLev=3;

Length of tutorial: 2 days

Target audience (paying) 100 people

Minimum audience 50 people

Latest date for cancellation of seminar if minimum is not met is 10 August 1978.

Title -- "The Office of the Future: Technology Applications & Impacts"

Registration fee: $200 ($100 per day per student) or 60,000 yen

Need to check the exchange rate, but it has been stable at around 300 yen to the dollar for a number of years.

Late fee - add $25 after 1 Aug 78.

No one day registrations will be allowed.

Target income: $ 20,000 (US)

Minimum income: $10,000

Method of payment: US Dollars or Japanese Yen.

This needs to be consistent with whatever ICCC is doing.

Question: What kind of financial problems need to be dealt with in terms of taking money out of Japan? I will discuss this with the ICCC Executive Council.

Expenditure plan:

Total of costs (other than speakers) $ 8000

Room rental: $800 ($400/day)

I will discuss this with the ICCC Executive Council and ask that this be checked out by the ICCC Treasurer, Ed Boyar, when he goes to Japan after the next ICCC Exec Council meeting to discuss arrangements for the whole ICCC with the Japanese program committee.

Catered luncheon for attendees at tutorial at $15 each per day
==> $3000.00

Honoraria for luncheon speakers - $200

Arrange for two leaders in Japanese data processing to speak.

Reproduction of speakers notes for passing out at tutorial: $2000

This is based on an estimate of 200 pages of material at ten cents per page for 100 people.

Production and mailing of advance information: $ 1230.00

This is based on an estimate of mailing 3000 brochures at a cost of 41 cents per brochure delivered to prospective attendee. The 41 cent estimate is based on a cost of 10 cents to produce a copy of the brochure and 31 cents for international first class mail from the USA. Since many of the brochures will go to people inside the USA, this number is on the high side. We may have to move some of the money from the Miscellaneous category to this category, since I don't know whether we will have to pay to have a brochure designed, and it might cost more to reproduce, if it is a "fancy" brochure. In any case, we do want to use the ICCC mailing list, and we want it to go to the same people who receive the ICCC mailing, hopefully at the same time.

Miscellaneous expenses - $770

Principal speakers: $12,000 ($3000 each)

Note: If registration drops as low as 50 people, only the total cost of the catered luncheon would drop. In that case, the cost of the luncheons for the two days would drop to $1500. The total of all costs other than speakers in that case would be $5730. That would leave $4270 to be divided among the four speakers, or approximately $1000 per speaker. That would just about cover the cost of transportation of the speakers to and from Japan and hotel rooms, and that is why the cut-off for the tutorial would be a minimum of 50 registered attendees. The principal speakers must be guaranteed a minimum of $1000 each, or the tutorial would have to be cancelled. If only 50 people registered, the "miscellaneous expenses" category above would drop to $270. If the registration exceeded 100 people, the extra money would be divided equally among the principal speakers.

Speakers: (Each of the speakers listed below has been contacted and has agreed to participate as a principal speaker in the tutorial)

Dave Farber, on hardware to support the "Office of the Future".

Louis Pouzin, on the role of computer networks in the "Office of the Future".

Ron Uhlig on Applications in the "Office of the Future".

Jim Bair on Psycho-social and organizational impacts of the "Office of the Future" (this should be especially interesting in Japan, considering the Japanese management approach, which is very different from the rest of the world, and especially from the USA).

Format: (each speaker needs to develop about a half day outline and circulate to the other speakers as soon as possible)

First half day: Introduction and discussion of applications (Uhlig)

second half day: The role of computer networks (Pouzin)

Third half day: /hardware (Farber)

Fourth half day: Impacts (Bair)

Basis of income for IFIP WG 6.3: Each speaker is expected to develop a comprehensive set of notes which will be provided to the registrants at the tutorial. Based on the results of the tutorial, these notes will be edited by the speakers after the tutorial, and the final edited version will be provided to Jim Bair, Chairman of WG 6.3. Jim will arange for publication of the tutorial notes by the IFIP publisher, North-Holland Publishing Company, for sale to the international community.

RFC # 733
NIC #41952

Obsoletes: RFC #561 (NIC #18516)
RFC #680 (NIC #32116)
RFC #724 (NIC #37435)

STANDARD FOR THE FORMAT OF ARPA NETWORK TEXT MESSAGES(1)

21 November 1977

by

David H. Crocker
The Rand Corporation

John J. Vittal
Bolt Beranek and Newman Inc.

Kenneth T. Pogran
Massachusets Institute of Technology

D. Austin Henderson, Jr.(2)
Bolt Beranek and Newman Inc.

---

(1)This work was supported by the Defense Advanced Research Projects Agency of the Department of Defense, under contract Nos. N00014-75-C-0661, MDA903-76-C-0212, and DAHC15-73-C0181.

(2)The authors' postal addresses are: D. Crocker, The Rand Corporation, Information Sciences Dept., 1700 Main St., Santa Monica, California 90406; J. Vittal & D. A. Henderson, Bolt Beranek & Newman, 50 Moulton St., Cambridge, Massachusetts 02138; and K. Pogran, MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, Massachusetts 02139. The authors' ARPANET addresses are: DCrocker at Rand-Unix, Vittal at BBN-TenexD, Pogran at MIT-Multics, and Henderson at BBN-TenexD.

PREFACE

ARPA's Committee on Computer-Aided Human Communication (CAHCOM) wishes to promulgate a standard for the format of ARPA Network text message (mail) headers which will reasonably meet the needs of the various message service subsystems on the Network today. The authors of this document constitute the CAHCOM subcommittee charged with the task of developing this new standard.

Essentially, we specify a revision to ARPANET Request for Comments (RFC) 561, "Standardizing Network Mail Headers", and RFC 680, "Message Transmission Protocol". This revision removes and compacts portions of the previous syntax and adds several features to network address specification. In particular, we focus on people and not mailboxes as recipients and allow reference to stored address lists. We expect this syntax to provide sufficient capabilities to meet most users' immediate needs and, therefore, give developers enough breathing room to produce a new mail transmission protocol "properly". We believe that there is enough of a consensus in the Network community in favor of such a standard syntax to make possible its adoption at this time. An earlier draft of this specification was published as RFC #724, "Proposed Official Standard for the Format of ARPA Network Messages" and contained extensive discussion of the background and issues in ARPANET mail standards.

This specification was developed over the course of one year, using the ARPANET mail environment, itself, to provide an on-going forum for discussing the capabilities to be included. More than twenty individuals, from across the country, participated in this discussion and we would like to acknowledge their considerable efforts. The syntax of the standard was originally specified in the Backus-Naur Form (BNF) meta-language. Ken L. Harrenstien, of SRI International, was responsible for re-coding the BNF into an augmented BNF which compacts the specification and allows increased comprehensibility.

CONTENTS

I. INTRODUCTION

This standard specifies a syntax for text messages which are passed between computer users within the framework of "electronic mail". The standard supersedes the informal standards specified in ARPANET Request for Comments numbers 561, "Standardizing Network Mail Headers", and 680, "Message Transmission Protocol". In this document, a general framework is first described; the formal syntax is then specified, followed by a discussion of the semantics. Finally, a number of examples are given.

This specification is intended strictly as a definition of what is to be passed between hosts on the ARPANET. It is NOT intended to dictate either features which systems on the Network are expected to support, or user interfaces to message creating or reading programs.

A distinction should be made between what the specification REQUIRES and what it ALLOWS. Messages can be made complex and rich with formally-structured components of information or can be kept small and simple, with a minimum of such information. Also, the standard simplifies the interpretation of differing visual formats in messages. These simplifications facilitate the formal specification and indicate what the OFFICIAL semantics are for messages. Only the visual aspect of a message is affected and not the interpretation of information within it. Implementors may choose to retain such visual distinctions.

II. FRAMEWORK

Since there are many message systems which exist outside the ARPANET environment, as well as those within it, it may be useful to consider the general framework, and resulting capabilities and limitations, provided by this standard.

Messages are expected to consist of lines of text. No special provisions are made, at this time, for encoding drawings, facsimile, speech, or structured text.

No significant consideration has been given to questions of data compression or transmission/storage efficiency. The standard, in fact, tends to be very free with the number of bits consumed. For example, field names are specified as free text, rather than special terse codes.

A general "memo" framework is used. That is, a message consists of some information, in a rigid format, followed by the main part of the message, which is text and whose format is not specified in this document. The syntax of several fields of the rigidly-formated ("header") section is defined in this specification; some of the header fields must be included in all messages. The syntax which distinguishes between headers is specified separately from the internal syntax for particular headers. This separation is intended to allow extremely simple parsers to operate on the overall structure of messages, without concern for the detailed structure of individual headers. Appendix B is provided to facilitate construction of these simple parsers. In addition to the fields specified in this document, it is expected that other fields will gain common use. User-defined header fields allow systems to extend their functionality while maintaining a uniform framework. The approach is similar to that of the TELNET protocol, in that a basic standard is defined which includes a mechanism for (optionally) extending itself. As necessary, the authors of this document will regulate the publishing of specifications for these "extension-fields", through the same mechanisms used to publish this document.

Such a framework severely constrains document tone and appearance and is primarily useful for most intra-organization communications and relatively structured inter-organization communication. A more robust environment might allow for multi-font, multi-color, multi-dimension encoding of information. A less robust environment, as is present in most single-machine message systems, would more severely constrain the ability to add fields and the decision to include specific fields. In contrast to paper-based communication, it is interesting to note that the

RECEIVER of a message can exercise an extraordinary amount of control over the message's appearance. The amount of actual control available to message receivers is contingent upon the capabilities of their individual message systems.

III. SYNTAX

This syntax is given in five parts. The first part describes the notation used in the specification. The second part describes the base-level lexical analyzers which feed the higher-level parser described in the succeeding sections. The third part gives a general syntax for messages and standard header fields; and the fourth part specifies the syntax of addresses. A final part specifies some general syntax which supports the other sections.

A. NOTATIONAL CONVENTIONS

These specifications are made in an augmented Backus-Naur Form (BNF). Differences from standard BNF involve the naming of rules, the indication of repetition and of "local" alternatives.

1. Rule naming

Angle brackets ("<", ">") are not used, in general. The name of a rule is simply the name itself, rather than "<name>". Quotation-marks enclose literal text (which may be upper and/or lower case). Certain basic rules are in uppercase, such as SPACE, TAB, CRLF, DIGIT, ALPHA, etc. Angle brackets are used in rule definitions, and in the rest of this document, whenever their presence will facilitate discerning the use of rule names.

2. Parentheses: Local alternatives

Elements enclosed in parentheses are treated as a single element. Thus, "(elem (foo / bar) elem)" allows "(elem foo elem)" and "(elem bar elem)".

3. * construct: Repetition

The character "*" preceding an element indicates repetition. The full form is:

<l>*<m>element

indicating at least <l> and at most <m> occurrences of element. Default values are 0 and infinity so that "*(element)" allows any number, including zero; "1*element" requires at least one; and "1*2element" allows one or two.

4. <number>element

"<n>(element)" is equivalent to "<n>*<n>(element)"; that is, exactly <n> occurrences of (element). Thus 2DIGIT is a 2-digit number, and 3ALPHA is a string of three alphabetic characters.

5. # construct: Lists

A construct "#" is defined, similar to "*", as follows:

<l>#<m>element

indicating at least <l> and at most <m> elements, each separated by one or more commas (","). This makes the usual form of lists very easy; a rule such as '(element *("," element))' can be shown as "1#element". Wherever this construct is used, null elements are allowed, but do not contribute to the count of elements present. That is, "(element),,(element)" is permitted, but counts as only two elements. Therefore, where at least one element is required, at least one non-null element must be present.

6. [optional]

Square brackets enclose optional elements; "[foo bar]" is equivalent to "*1(foo bar)".

7. ; Comments

A semi-colon, set off some distance to the right of rule text, starts a comment which continues to the end of line. This is a simple way of including useful notes in parallel with the specifications.

B. LEXICAL ANALYSIS OF MESSAGES

1. General Description

A message consists of headers and, optionally, a body (i.e. a series of text lines). The text part is just a sequence of lines containing ASCII characters; it is separated from the headers by a null line (i.e., a line with nothing preceding the CRLF).

a. Folding and unfolding of headers

Each header item can be viewed as a single, logical line of ASCII characters. For convenience, the field-body portion of this conceptual entity can be split into a multiple-line representation (i.e., "folded"). The general rule is that wherever there can be linear-white-space (NOT simply LWSP-chars), a CRLF immediately followed by AT LEAST one LWSP-char can instead be inserted. (However, a header's name and the following colon (":"), which occur at the beginning of the header item, may NOT be folded onto multiple lines.) Thus, the single line

```
To: "Joe Dokes & J. Harvey" <ddd at Host>, JJV at BBN
```

can be represented as

```
To: "Joe Dokes & J. Harvey" <ddd at Host>,
    JJV at BBN
```

and

```
To: "Joe Dokes & J. Harvey"
                <ddd at Host>,
 JJV at BBN
```

and

```
To: "Joe Dokes
 & J. Harvey" <ddd at Host>, JJV at BBN
```

The process of moving from this folded multiple-line representation of a header field to its single line representation will be called "unfolding". Unfolding is accomplished by regarding CRLF immediately followed by a LWSP-char as equivalent to the LWSP-char.

b. Structure of header fields

Once header fields have been unfolded, they may be viewed as being composed of a field-name followed by a colon (":"), followed by a field-body. The field-name must be composed of printable ASCII characters (i.e., characters which have values between 33. and 126., decimal, except colon) and LWSP-chars. The field-body may be composed of any ASCII characters (other than an unquoted CRLF, which has been removed by unfolding).

Certain field-bodies of header fields may be interpreted according to an internal syntax which some systems may wish to parse. These fields will be referred to as "structured" fields. Examples include fields containing dates and

addresses. Other fields, such as "Subject" and "Comments", are regarded simply as strings of text.

NOTE: Field-names, unstructured field bodies and structured field bodies each are scanned by their own, INDEPENDENT "lexical" analyzer.

c. Field-names

To aid in the creation and reading of field-names, the free insertion of LWSP-chars is allowed in reasonable places.

Rather than obscuring the syntax specification for field-name with the explicit syntax for these LWSP-chars, the existence of a "lexical" analyzer is assumed. The analyzer interprets the text which comprises the field-name as a sequence of field-name atoms (fnatoms) separated by LWSP-chars

Note that ONLY LWSP-chars may occur between the fnatoms of a field-name and that CRLFs may NOT. In addition, comments are NOT lexically recognized, as such, but parenthesized strings are legal as part of field-names. These constraints are different from what is permissible within structured field bodies. In particular, this means that header field-names must wholly occur on the FIRST line of a folded header item and may NOT be split across two or more lines.

d. Unstructured field bodies

For some fields, such as "Subject" and "Comments", no structuring is assumed; and they are treated simply as texts, like those in the message body. Rules of folding apply to these fields, so that such field bodies which occupy several lines must therefore have the second and successive lines indented by at least one LWSP-char.

e. Structured field bodies

To aid in the creation and reading of structured fields, the free insertion of linear-white-space (which permits folding by inclusion of CRLFs) is allowed in reasonable places. Rather than obscuring the syntax specifications for these structured fields with explicit syntax for this linear-white-space, the existence of another "lexical" analyzer is assumed. This analyzer does not apply for field bodies which are simply unstructured strings of text, as described above. It provides an interpretation of the unfolded text comprising the body of the field as a sequence of lexical symbols. These symbols are:

- individual special characters
- quoted-strings

- comments
- atoms

The first three of these symbols are self-delimiting. Atoms are not; they therefore are delimited by the self-delimiting symbols and by linear-white-space. For the purposes of re-generating sequences of atoms and quoted-strings, exactly one SPACE is assumed to exist and should be used between them. (Also, in Section III.B.3.a, note the rules concerning treatment of multiple continguous LWSP-chars.)

So, for example, the folded body of an address field

```
":sysmail"@  Some-Host,
Muhammed(I am  the greatest)Ali  at(the)WBA
```

is analyzed into the following lexical symbols and types:

":sysmail"	quoted string
@	special
Some-Host	atom
,	special
Muhammed	atom
(I am the greatest)	comment
Ali	atom
at	atom
(the)	comment
WBA	atom

The cononical representations for the data in these addresses are the following strings (note that there is exactly one SPACE between words):

```
:sysmail at Some-Host
```

and

```
Muhammed Ali at WBA
```

2. Formal Definitions

The first four rules, below, indicate a meta-syntax for fields, without regard to their particular type or internal syntax. The remaining rules define basic syntactic structures which are used by the rules in Sections III.C, III.D, and III.E.

```
field      =  field-name ":" [ field-body ] CRLF

field-name =  fnatom *( LWSP-char [fnatom] )
```

```
fnatom       =  1*<any CHAR, excluding CTLs, SPACE, and ":">

field-body   =  field-body-contents
                [CRLF LWSP-char field-body]

field-body-contents = <the TELNET ASCII characters making up the
                field-body, as defined in the following sections,
                and consisting of combinations of atom, quoted-
                string, and specials tokens, or else consisting of
                texts>

                                                ; (  Octal, Decimal.)
CHAR         =  <any TELNET ASCII character>    ; (  0-177,  0.-127.)
ALPHA        =  <any TELNET ASCII alphabetic character>
                                                ; (101-132, 65.- 90.)
                                                ; (141-172, 97.-122.)
DIGIT        =  <any TELNET ASCII digit>        ; ( 60- 71, 48.- 57.)
CTL          =  <any TELNET ASCII control       ; (  0- 37,  0.- 31.)
                 character and DEL>             ; (    177,    127.)
CR           =  <TELNET ASCII carriage return>  ; (     15,     13.)
LF           =  <TELNET ASCII linefeed>         ; (     12,     10.)
SPACE        =  <TELNET ASCII space>            ; (     40,     32.)
HTAB         =  <TELNET ASCII horizontal-tab>   ; (     11,      9.)
<">          =  <TELNET ASCII quote mark>       ; (     42,     34.)
CRLF         =  CR LF

LWSP-char    =  SPACE / HTAB                    ; semantics = SPACE
linear-white-space =  1*([CRLF] LWSP-char)      ; semantics = SPACE
                                                ; CRLF => folding

specials     =  "(" / ")" / "<" / ">" / "@"     ; To use in a word,
             /  "," / ";" / ":" / "\" / <">     ;  word must be a
                                                ;  quoted-string.

delimiters   =  specials / comment / linear-white-space

text         =  <any CHAR, including bare       ; => atoms, specials,
                 CR and/or bare LF, but NOT     ;  comments and
                 including CRLF>                ;  quoted-strings are
                                                ;  NOT interpreted.

atom         =  1*<any CHAR except specials and CTLs>

quoted-string = <"> *(qtext/quoted-pair) <">; Any number of qtext
                                                ;   chars or any
                                                ;   quoted char.

qtext        =  <any CHAR excepting <">         ; => may be folded
                 and CR, and including
                 linear-white-space>
```

```
comment     =  "(" *(ctext / comment / quoted-pair) ")"
ctext       =  <any CHAR excluding "(",     ; => may be folded
               ")" and CR, and including
               linear-white-space>

quoted-pair =  "\" CHAR
```

3. Clarifications

a. "White space"

Remember that in field-names and structured field bodies, MULTIPLE LINEAR WHITE SPACE TELNET ASCII CHARACTERS (namely HTABs and SPACEs) ARE TREATED AS SINGLE SPACES AND MAY FREELY SURROUND ANY SYMBOL. In all header fields, the only place in which at least one space is REQUIRED is at the beginning of continuation lines in a folded field. When passing text to processes which do not interpret text according to this standard (e.g., ARPANET FTP mail servers), then exactly one SPACE should be used in place of arbitrary linear-white-space and comment sequences.

WHEREVER A MEMBER OF THE LIST OF <DELIMITER>S IS ALLOWED, LWSP-CHARS MAY ALSO OCCUR BEFORE AND/OR AFTER IT.

Writers of mail-sending (i.e. header generating) programs should realize that there is no Network-wide definition of the effect of horizontal-tab TELNET ASCII characters on the appearance of text at another Network host; therefore, the use of tabs in message headers, though permitted, is discouraged.

Note that during transmissions across the ARPANET using TELNET NVT connections, data must conform to TELNET NVT conventions (e.g., CR must be followed by either LF, making a CRLF, or <null>, if the CR is to stand alone).

b. Comments

Comments are detected as such only within field-bodies of structured fields. A comment is a set of TELNET ASCII characters, which is not within a quoted-string and which is enclosed in matching parentheses; parentheses nest, so that if an unquoted left parenthesis occurs in a comment string, there must also be a matching right parenthesis. When a comment is used to act as the delimiter between a sequence of two lexical symbols, such as two atoms, it is lexically equivalent with one SPACE, for the purposes of regenerating the sequence, such as when passing the sequence onto an FTP mail server.

In particular comments are NOT passed to the FTP server, as part of a MAIL or MLFL command, since comments are not part of the "formal" address.

If a comment is to be "folded" onto multiple lines, then the syntax for folding must be adhered to. (See items III.B.1.a, above, and III.B.3.f, below.) Note that the official semantics therefore do not "see" any unquoted CRLFs which are in comments, although particular parsing programs may wish to note their presence. For these programs, it would be reasonable to interpret a "CRLF LWSP-char" as being a CRLF which is part of the comment; i.e., the CRLF is kept and the LWSP-char is discarded. Quoted CRLFs (i.e., a backslash followed by a CR followed by a LF) still must be followed by at least one LWSP-char.

c. Delimiting and quoting characters

The quote character (backslash) and characters which delimit syntactic units are not, generally, to be taken as data which are part of the delimited or quoted unit(s). The one exception is SPACE. In particular, the quotation-marks which define a quoted-string, the parentheses which define a comment and the backslash which quotes a following character are NOT part of the quoted-string, comment or quoted character. A quotation-mark which is to be part of a quoted-string, a parenthesis which is to be part of a comment and a backslash which is to be part of either must each be preceded by the quote-character backslash ("\"). Note that the syntax allows any character to be quoted within a quoted-string or comment; however only certain characters MUST be quoted to be included as data. These characters are those which are not part of the alternate text group (i.e., ctext or qtext).

A single SPACE is assumed to exist between contiguous words in a phrase, and this interpretation is independent of the actual number of LWSP-chars which the creator places between the words. To include more than one SPACE, the creator must make the LWSP-chars be part of a quoted-string.

Quotation marks which delimit a quoted string and backslashes which quote the following character should NOT accompany the quoted-string when the string is used with processes that do not interpret data according to this specification (e.g., ARPANET FTP mail servers).

d. Quoted-strings

Where permitted (i.e., in words in structured fields) quoted-strings are treated as a single symbol (i.e. equivalent to an atom, syntactically). If a quoted-string is to be "folded" onto multiple lines, then the syntax for folding must be adhered to. (See items III.B.1.a, above, and III.B.3.f, below.) Note that the official semantics therefore do not "see" any bare CRLFs which are in quoted-strings, although particular parsing programs may wish to note their presence. For these programs, it would be reasonable to interpret a "CRLF LWSP-char" as being a CRLF which is part of the quoted-string; i.e., the CRLF is kept and the LWSP-char is discarded. Quoted CRLFs (i.e., a backslash followed by a CR followed by a LF) are also subject to rules of folding, but the presence of the quoting character (backslash) explicitly indicates that the CRLF is data to the quoted string. Stripping off the first following LWSP-char is also appropriate when parsing quoted CRLFs.

e. Bracketing characters

There are three types of brackets which must be well nested:

- o Parentheses are used to indicate comments.
- o Angle brackets ("<" and ">") are generally used to indicate the presence of at least one machine-usable code (e.g., delimiting mailboxes).
- o Colon/semi-colon (":" and ";") are used in address specifications to indicate that the included list of addresses are to be treated as a group.

f. Case independence of certain specials atoms

Certain atoms, which are represented in the syntax as literal alphabetic strings, can be represented in any combination of upper and lower case. These are:

- field-name,
- "Include", "Postal" and equivalent atoms in a ":"<atom>":" address specification,
- "at", in a host-indicator,
- node,
- day-of-week,
- month, and
- zones.

When matching an atom against one of these literals, case is to be ignored. For example, the field-names "From", "FROM",

"from", and even "FroM" should all be treated identically. However, the case shown in this specification is suggested for message-creating processes. Note that, at the level of this specification, case IS relevant to other words and texts. Also see Section IV.A.1.f, below.

g. Folding long lines

Each header item (field of the message) may be represented on exactly one line consisting of the name of the field and its body; this is what the parser sees. For readability, it is recommended that the field-body portion of long header items be "folded" onto multiple lines of the actual header. "Long" is commonly interpreted to mean greater than 65 or 72 characters. The former length is recommended as a limit, but it is not imposed by this standard.

h. Backspace characters

Backspace TELNET ASCII characters (ASCII BS, decimal 8.) may be included in texts and quoted-strings to effect overstriking; however, any use of backspaces which effects an overstrike to the left of the beginning of the text or quoted-string is prohibited.

C. GENERAL SYNTAX OF MESSAGES:

NOTE: Due to an artifact of the notational conventions, the syntax indicates that, when present, "Date", "From", "Sender", and "Reply-To" fields must be in a particular order. These header items must be unique (occur exactly once). However header fields, in fact, are NOT required to occur in any particular order, except that the message body must occur AFTER the headers. For readability and ease of parsing by simple systems, it is recommended that headers be sent in the order "Date", "From", "Subject", "Sender", "To", "cc", etc. This specification permits multiple occurrences of most optional-fields. However, their interpretation is not specified here, and their use is strongly discouraged.

The following syntax for the bodies of various fields should be thought of as describing each field body as a single long string (or line). The section on Lexical Analysis (section II.B) indicates how such long strings can be represented on more than one line in the actual transmitted message.

```
message      =  fields *( CRLF *text )        ; Everything after
                                              ;  first null line
                                              ;  is message body

fields       =  date-field                    ; Creation time-stamp
                originator-fields             ;  & author id are
                *optional-field               ;  required: others
                                              ;  are all optional

originator-fields =
                (  "From"      ":" mailbox    ; Single author
                  ["Reply-To" ":" #address] )
               / (  "From"     ":" 1#address  ; Multiple authors &
                    "Sender"   ":" mailbox    ;  may have non-mach-
                  ["Reply-To" ":" #address] );  ine addresses

date-field   =  "Date"        ":" date-time

optional-field  =
                "To"          ":" #address
               / "cc"         ":" #address
               / "bcc"        ":" #address    ; Blind carbon
               / "Subject"    ":" *text
               / "Comments"   ":" *text
               / "Message-ID" ":" mach-id     ; Only one allowed
               / "In-Reply-To""":" #(phrase / mach-id)
               / "References" ":" #(phrase / mach-id)
               / "Keywords"   ":" #phrase
               / extension-field               ; To be defined in
                                               ;  supplemental
                                               ;  specifications
               / user-defined-field            ; Must have unique
                                               ;  field-name & may
                                               ;  be pre-empted

extension-field = <Any field which is defined in a document
                published as a formal extension to this
                specification>

user-defined-field = <Any field which has not been defined in
                this specification or published as an extension to
                this specification; names for such fields must be
                unique and may be preempted by published
                extensions>
```

D. SYNTAX OF GENERAL ADDRESSEE ITEMS

```
address     =  host-phrase                      ; Machine mailbox
            / ( [phrase] "<" #address ">")      ; Individual / List
            / ( [phrase] ":" #address ";")      ; Group
            /  quoted-string                    ; Arbitrary text
            / (":" ( "Include"                  ; File, w/ addr list
                   / "Postal"                   ; (U.S.) Postal addr
                   /  atom )                    ; Extended data type
               ":" address)

mailbox     =  host-phrase /  (phrase mach-id)

mach-id     =  "<" host-phrase ">"              ; Contents must never
                                                ;  be modified!
```

E. SUPPORTING CONSTRUCTS

```
host-phrase =  phrase  host-indicator           ; Basic address

host-indicator =  1*( ("at" / "@") node )       ; Right-most node is
                                                ;  at top of network
                                                ;  hierarchy; left-
                                                ;  most must be host

node        =  word / 1*DIGIT                   ; Official host or
                                                ;  network name or
                                                ;  decimal address


date-time   =  [ day-of-week "," ] date time

day-of-week =  "Monday"    / "Mon"  / "Tuesday"   / "Tue"
            /  "Wednesday" / "Wed"  / "Thursday"  / "Thu"
            /  "Friday"    / "Fri"  / "Saturday"  / "Sat"
            /  "Sunday"    / "Sun"

date        =  1*2DIGIT ["-"] month             ; day month year
               ["-"] (2DIGIT /4DIGIT)           ;  e.g. 20 Aug [19]77

month       =  "January"   / "Jan"  / "February"  / "Feb"
            /  "March"     / "Mar"  / "April"     / "Apr"
            /  "May"                / "June"      / "Jun"
            /  "July"      / "Jul"  / "August"    / "Aug"
            /  "September" / "Sep"  / "October"   / "Oct"
            /  "November"  / "Nov"  / "December"  / "Dec"
```

```
time        =  hour zone                        ; ANSI and Military
                                                ;  (seconds optional)

hour        =  2DIGIT [":"] 2DIGIT [ [":"] 2DIGIT ]
                                                ; 0000[00] - 2359[59]

zone        = ( ["-"] ( "GMT"                   ; Relative to GMT:
                                                ; North American
                  /  "NST" /                    ;  Newfoundland:-3:30
                  /  "AST" / "ADT"              ;  Atlantic: - 4/ - 3
                  /  "EST" / "EDT"              ;  Eastern:  - 5/ - 4
                  /  "CST" / "CDT"              ;  Central:  - 6/ - 5
                  /  "MST" / "MDT"              ;  Mountain: - 7/ - 6
                  /  "PST" / "PDT"              ;  Pacific:  - 8/ - 7
                  /  "YST" / "YDT"              ;  Yukon:    - 9/ - 8
                  /  "HST" / "HDT"              ;  Haw/Ala   -10/ - 9
                  /  "BST" / "BDT"              ;  Bering:   -11/ -10
                     1ALPHA       ))            ; Military: Z = GMT;
                                                ;  A:-1; (J not used)
                                                ;  M:-12; N:+1; Y:+12
            / ( ("+" / "-") 4DIGIT )            ; Local differential
                                                ;  hours/min. (HHMM)

phrase      =  1*word                           ; Sequence of words.
                                                ;  Separation seman-
                                                ;  tically = SPACE

word        =  atom / quoted-string
```

IV. SEMANTICS

A. ADDRESS FIELDS

1. General

a. The phrase part of a host-phrase in an address specification (i.e., the host's name for the mailbox) is understood to be whatever the receiving FTP Server allows (for example, TENEX systems do not now understand addresses of the form "P. D. Q. Bach", but another system might).

Note that a mailbox is a conceptual entity which does not necessarily pertain to file storage. For example, some sites may choose to print mail on their line printer and deliver the output to the addressee's desk.

An individual may have several mailboxes and a group of individuals may wish to receive mail as a single unit (i.e., a distribution list). The second and third alternatives of an address list (#address) allow naming a collection of subordinate addresses list(s). Recipient mailboxes are specified within the bracketed part ("<" - ">" or ":" - ";") of such named lists. The use of angle-brackets ("<", ">") is intended for the cases of individuals with multiple mailboxes and of special mailbox lists; it is not expected to be nested more than one level, although the specification allows such nesting. The use of colon/semi-colon (":", ";") is intended for the case of groups. Groups can be expected to nest (i.e., to contain subgroups). For both individuals and groups, a copy of the transmitted message is to be sent to EACH mailbox listed. For the case of a special list, treatment of addresses is defined in the relevant subsections of this section.

b. The inclusion of bare quoted-strings as addresses (i.e., the fourth address-form alternative) is allowed as a syntactic convenience, but no semantics are defined for their use. However, it is reasonable, when replicating an address list, to replicate ALL of its members, including quoted-strings.

c. ":Include:" specifications are used to refer to one or more locations containing stored address lists (#address). If more than one location is referenced, the address part of the Include phrase must be a list (#address) surrounded by angle-brackets, as per the "Individual / List" alternative of <address>. Constituent addresses must resolve to a host-

phrase; only they have any meaning within this construct. The phrase part of indicated host-phrases should contain text which the referenced host can resolve to a file. This standard is not a protocol and so does not prescribe HOW data is to be retrieved from the file. However, the following requirements are made:

- The file must be accessible through the local operating system interface (if it exists), given adequate user access rights; and

- If a host has an FTP server and a user is able to retrieve any files from the host using that server, then the file must be accessible through FTP, using DEFAULT transfer settings, given adequate user access rights.

It is intended that this mechanism allow programs to retrieve such lists automatically.

The interpretation of such a file reference follows. This is not intended to imply any particular implementation scheme, but is presented to aid in understanding the notion of including file contents in address lists:

- Elements of the address list part are alternates and the contents of ONLY ONE of them are to be included in the resultant address list.

- The contents of the file indicated by a member host-phrase are treated as an address list and are inserted as an address list (#address) in the position of the path item in the syntax. That is, the TELNET ASCII characters specifying the entire Include <address> is replaced by the contents of one of the files to which the host-phrase(s), of the address list (#address), refers. Therefore, the contents of each file, indicated by an Include address, must be syntactically self-contained and must adhere to the full syntax prescribed herein for an address list.

d. ":Postal:" specifications are used to indicate (U.S.) postal addresses, but can be treated the same as quoted-string addresses. To reference a list of postal addresses, the list must conform to the "Individual / List" alternative of <address>. The ":Include:" alternative also is valid.

e. The "*:* atom *:*" syntax is intended as a general mechanism for indicating specially data-typed addresses. As with extension-fields, the authors of this document will regulate

the publishing of specifications for these extended data-types. In the absence of defined semantics, any occurrence of an address in this form may be treated as a quoted-string address.

f. A node name must be THE official name of a network or a host, or else a decimal number indicating the Network address for that network or host, at the time the message is created. The USE OF NUMBERS IS STRONGLY DISCOURAGED and is permitted only due to the occasional necessity of bypassing local name tables. For the ARPANET, official names are maintained by the Network Information Center at SRI International, Menlo Park, California.

Whenever a message might be transmitted or migrate to a host on another network, full hierarchical addresses must be specified. These are indicated as a series of words, separated by at-sign or "at" indications. The communication environment is assumed to consist of a collection of networks organized as independent "trees" except for connections between the root nodes. That is, only the roots can act as gateways between these independent networks. While other actual connections may exist, it is believed that presuming this type of organization will provide a reliable method for describing VALID, if not EFFICIENT, paths between hosts. A typical full mailbox specification might therefore look like:

Friendly User @ hosta @ local-net1 @ major-netq

In the simplest case, a mail-sending host should transmit the message to the node which is mentioned last (farthest to the right), strip off that node reference from the specification, and then pass the remaining host-phrase to the recipient host (in the ARPANET, its FTP server) for it to process. This treats the remaining portion of the host-indicator merely as the terminating part of the phrase.

NOTE: When passing any portion of a host-indicator onto a process which does not interpret data according to this standard (e.g., ARPANET FTP servers), "@" must be used and not "at" and it must not be preceded or followed by any LWSP-chars. Using the above example, the following string would be passed to the major-netq gateway:

Friendly User@hosta@local-net1

When the sending host has more knowledge of the network environment, then it should send the message along a more efficient path, making appropriate changes to the form of the host-phrase which it gives to the recipient host.

To use the above specification as an example: If a sending hostb also were part of local-net1, then it could send the message directly to hosta and would give only the phrase "Friendly User" to hosta's mail-receiving program. If hostb were part of local-net2, along with hostc, and happened to know that hosta and hostc were part of another local-net, then hostb could send the message to hostc to the address "Friendly User@hosta".

The phrase in a host-phrase is intended to be meaningful only to the indicated receiving host. To all other hosts, the phrase is to be treated as an uninterpreted string. No case transformations should be (automatically) performed on the phrase. The phrase is passed to the local host's mail sending program; it is the responsibility of the destination host's mail receiving (distribution) program to perform case mapping on this phrase, if required, to deliver the mail.

2. Originator Fields

WARNING: The standard allows only a subset of the combinations possible with the From, Sender, and Reply-To fields. The limitation is intentional.

a. From

This field contains the identity of the person(s) who wished this message to be sent. The message-creation process should default this field to be a single machine address, indicating the AGENT (person or process) entering the message. If this is NOT done, the "Sender" field MUST be present; if this IS done, the "Sender" field is optional.

b. Sender

This field contains the identity of the AGENT (person or process) who sends the message. It is intended for use when the sender is not the author of the message, or to indicate who among a group of authors actually sent the message. If the contents of the "Sender" field would be completely redundant with the "From" field, then the "Sender" field need not be present and its use is discouraged (though still legal); in particular, the "Sender" field MUST be present if it is NOT the same as the "From" Field.

The Sender host-phrase includes a phrase which must correspond to a specific agent (i.e., a human user or a computer program) rather than a standard address. This indicates the expectation that the field will identify the single AGENT (person or process) responsible for sending the

mail and not simply include the name of a mailbox from which the mail was sent. For example in the case of a shared login name, the name, by itself, would not be adequate. The phrase part of the host-phrase, which refers to this agent, is expected to be a computer system term, and not (for example) a generalized person reference which can be used outside the network text message context.

Since the critical function served by the "Sender" field is the identification of the agent responsible for sending mail and since computer programs cannot be held accountable for their behavior, is strongly recommended that when a computer program generates a message, the HUMAN who is responsible for that program be referenced as part of the "Sender" field host-phrase.

c. Reply-To

This field provides a general mechanism for indicating any mailbox(es) to which responses are to be sent. Three typical uses for this feature can be distinguished. In the first case, the author(s) may not have regular machine-based mailboxes and therefore wish(es) to indicate an alternate machine address. In the second case, an author may wish additional persons to be made aware of, or responsible for, responses; responders should send their replies to the "Reply-To" mailbox(es) listed in the original message. A somewhat different use may be of some help to "text message teleconferencing" groups equipped with automatic distribution services: include the address of that service in the "Reply-To" field of all messages submitted to the teleconference; then participants can "reply" to conference submissions to guarantee the correct distribution of any submission of their own.

Reply-To fields are NOT required to contain any machine addresses (i.e., host-phrases). Note, however, that the absence of even one valid network address will tend to prevent software systems from automatically assisting users in conveniently responding to mail.

NOTE: For systems which automatically generate address lists for replies to messages, the following recommendations are made:

- The receiver, when replying to a message, should NEVER automatically include the "Sender" host-phrase in the reply's address list;
- If the "Reply-To" field exists, then the reply should go ONLY to the addresses indicated in that field and not to the addresses indicated in the "From" field.

(Extensive examples are provided in Section V.) This recommendation is intended only for originator-fields and is not intended to suggest that replies should not also be sent to the other recipients of this message. It is up to the respective mail handling programs to decide what additional facilities will be provided.

3. Receiver Fields

a. To

This field contains the identity of the primary recipients of the message.

b. cc

This field contains the identity of the secondary recipients of the message.

b. Bcc

This field contains the identity of additional recipients of the message. The contents of this field are not included in copies of the message sent to the primary and secondary recipients. Some systems may choose to include the text of the "Bcc" field only in the author(s)'s copy, while others may also include it in the text sent to all those indicated in the "Bcc" list.

B. REFERENCE SPECIFICATION FIELDS

1. Message-ID

This field contains a unique identifier (the phrase) which refers to THIS version of THIS message. The uniqueness of the message identifier is guaranteed by the host which generates it. This identifier is intended to be machine readable and not necessarily meaningful to humans. A message identifier pertains to exactly one instantiation of a particular message; subsequent revisions to the message should each receive a new message identifier.

2. In-Reply-To

The contents of this field identify previous correspondence which this message answers. Note that if message identifiers are used in this field, they must use the mach-id specification format.

3. References

The contents of this field identify other correspondence which this message references. Note that if message identifiers are used, they must use the mach-id specification format.

4. Keywords

This field contains keywords or phrases, separated by commas.

C. OTHER FIELDS AND SYNTACTIC ITEMS

1. Subject

The "Subject" field is intended to provide as much information as necessary to adequately summarize or indicate the nature of the message.

2. Comments

Permits adding text comments onto the message without disturbing the contents of the message's body.

3. Extension-field

A relatively limited number of common fields have been defined in this document. As network mail requirements dictate, additional fields may be standardized. The authors of this document will regulate the publishing of such definitions as extensions to the basic specification.

4. User-defined-field

Individual users of network mail are free to define and use additional header fields. Such fields must have names which are not already used in the current specification or in any definitions of extension-fields, and the overall syntax of these user-defined-fields must conform to this specification's rules for delimiting and folding fields. Due to the extension-field publishing process, the name of a user-defined-field may be pre-empted.

D. DATES AND TIMES

If included, day-of-week must be the day implied by the date specification.

Time zone may be indicated in several ways. The military standard uses a single character for each zone. "Z" is Greenwhich Mean Time; "A" indicates one hour earlier, and "M" indicates 12 hours earlier; "N" is one hour later, and "Y" is 12 hours later. The letter "J" is not used. The other remaining two forms are taken from ANSI standard X3.51-1975. One allows explicit indication of the amount of offset from GMT; the other uses common 3-character strings for indicating time zones in North America.

V. EXAMPLES

A. ADDRESSES

1. Alfred E. Neuman <Neuman at BBN-TENEXA>

2. Neuman@BBN-TENEXA

These two "Alfred E. Neuman" examples have identical semantics, as far as the operation of the local host's mail sending (distribution) program (also sometimes called its "mailer") and the remote host's FTP server are concerned. In the first example, the "Alfred E. Neuman" is ignored by the mailer, as "Neuman at BBN-TENEXA" completely specifies the recipient. The second example contains no superfluous information, and, again, "Neuman@BBN-TENEXA" is the intended recipient.

3. Al Neuman at BBN-TENEXA

This is identical to "Al Neuman <Al Neuman at BBN-TENEXA>". That is, the full phrase, "Al Neuman", is passed to the FTP server. Note that not all FTP servers accept multi-word identifiers; and some that do accept them will treat each word as a different addressee (in this case, attempting to send a copy of the message to "Al" and a copy to "Neuman").

4. "George Lovell, Ted Hackle" <Shared-Mailbox at Office-1>

This form might be used to indicate that a single mailbox is shared by several users. The quoted string is ignored by the originating host's mailer, as "Shared-Mailbox at Office-1" completely specifies the destination mailbox.

4. Wilt (the Stilt) Chamberlain at NBA

The "(the Stilt)" is a comment, which is NOT included in the destination mailbox address handed to the originating system's mailer. The address is the string "Wilt Chamberlain", with exactly one space between the first and second words. (The quotation marks are not included.)

B. ADDRESS LISTS

```
Gourmets:  Pompous Person <WhoZiWhatZit at Cordon-Bleu>,
           Cooks:  Childs at WGBH, Galloping Gourmet at
                   ANT (Australian National Television);,
           Wine Lovers:  Cheapie at Discount-Liquors,
                         Port at Portugal;;,
Jones at SEA
```

This group list example points out the use of comments, the nesting of groups, and the mixing of addresses and groups. Note that the two consecutive semi-colons preceding "Jones at SEA" mean that Jones is NOT a member of the Gourmets group.

C. ORIGINATOR ITEMS

1. Author-sent

George Jones logs into his Host as "Jones". He sends mail himself.

```
From:  Jones at Host
```

or

```
From:  George Jones <Jones at Host>
```

2. Secretary-sent

George Jones logs in as Jones on his Host. His secretary, who logs in as Secy on Shost sends mail for him. Replies to the mail should go to George, of course.

```
From:    George Jones <Jones at Host>
Sender:  Secy at SHost
```

3. Shared directory or unrepresentative directory-name

George Jones logs in as Group at Host. He sends mail himself; replies should go to the Group mailbox.

```
From:  George Jones <Group at Host>
```

4. Secretary-sent, for user of shared directory

George Jones' secretary sends mail for George in his capacity as a member of Group while logged in as Secy at Host. Replies should go to Group.

```
From:   George Jones<Group at Host>
Sender: Secy at Host
```

Note that there need not be a space between "Jones" and the "<", but adding a space enhances readability (as is the case in other examples).

5. Secretary acting as full agent of author

George Jones asks his secretary (Secy at Host) to send a message for him in his capacity as Group. He wants his secretary to handle all replies.

```
From:     George Jones <Group at Host>
Sender:   Secy at Host
Reply-To: Secy at Host
```

6. Agent for user without online mailbox

A non-ARPANET user friend of George's, Sarah, is visting. George's secretary sends some mail to a friend of Sarah in computer-land. Replies should go to George, whose mailbox is Jones at Host.

```
From:     Sarah Friendly
Sender:   Secy at Host
Reply-To: Jones at Host
```

7. Sent by member of a committee

George is a member of a committee. He wishes to have any replies to his message go to all committee members.

```
From:     George Jones
Sender:   Jones at Host
Reply-To: Big-committee: Jones at Host,
                         Smith at Other-Host,
                         Doe at Somewhere-Else;
```

Note that if George had not included himself in the enumeration of Big-committee, he would not have gotten an implicit reply; the presence of the "Reply-to" field SUPERSEDES the sending of a reply to the person named in the "From" field.

8. Example of INCORRECT use

George desires a reply to go to his secretary; therefore his secretary leaves his mailbox address off the "From" field, leaving only his name, which is not, itself, a mailbox address.

```
From:   George Jones
Sender: Secy at SHost
```

THIS IS NOT PERMITTED. Replies are NEVER implicitly sent to the "Sender"; George's secretary should have used the "Reply-To" field, or the mail creating program should have forced the secretary to.

9. Agent for member of a committee

George's secretary sends out a message which was authored jointly by all the members of the "Big-committee".

```
From:   Big-committee: Jones at Host,
                       Smith at Other-Host,
                       Doe at Somewhere-Else;
Sender: Secy at SHost
```

D. COMPLETE HEADERS

1. Minimum required:

```
Date:  26 August 1976 1429-EDT
From:  Jones at Host
```

2. Using some of the additional fields:

```
Date: 26 August 1976 1430-EDT
From:George Jones<Group at Host>
Sender:Secy at SHOST
To:Al Neuman at Mad-Host,
          Sam Irving at Other-Host
Message-ID:  <some string at SHOST>
```

3. About as complex as you're going to get:

```
Date        :  27 Aug 1976 0932-PDT
From        :  Ken Davis <KDavis at Other-Host>
Subject     :  Re: The Syntax in the RFC
Sender      :  KSecy at Other-Host
Reply-To    :  Sam Irving at Other-Host
To          :  George Jones <Group at Host>,
               Al Neuman at Mad-Host
cc          :  Important folk:
               Tom Softwood <Balsa at Another-Host>,
               Sam Irving at Other-Host;,
               Standard Distribution::Include:
                </main/davis/people/standard at Other-Host,
                  "<Jones>standard.dist.3" at Tops-20-Host>,
               (The following Included Postal list is part
               of Standard Distribution.)
               :Postal::Include: Non-net-addrs@Other-host;,
               :Postal: "Sam Irving, P.O. Box 001, Las Vegas,
                         Nevada"  (So that he can stay
                         apprised of the situation)
Comment     :  Sam is away on business. He asked me to handle
               his mail for him.  He'll be able to provide a
               more accurate explanation when he returns
               next week.
In-Reply-To: <some string at SHOST>
Special (action):  This is a sample of multi-word field-
               names, using a range of characters.  There
               could also be a field-name "Special (info)".
Message-ID: <4231.629.XYzi-What at Other-Host>
```

APPENDIX

A. ALPHABETICAL LISTING OF SYNTAX RULES

```
address      =  host-phrase / quoted-string
             / (*phrase "<" #address ">" )
             / (*phrase ":" #address ";" )
             / (":" ("Include" / "Postal" / atom) ":" address)
ALPHA        =  <any TELNET ASCII alphabetic character>
atom         =  1*<any CHAR except specials and CTLs>

CHAR         =  <any TELNET ASCII character>
comment      =  "(" *(ctext / comment / quoted-pair) ")"
CR           =  <TELNET ASCII carriage return>
CRLF         =  CR LF
ctext        =  <any CHAR excluding "(", ")", CR, LF and
                including linear-white-space>
CTL          =  <any TELNET ASCII control character and DEL>

date         =  1*2DIGIT ["-"] month ["-"] (2DIGIT /4DIGIT)
date-field   =  "Date"       ":" date-time
date-time    =  [ day-of-week "," ] date time
day-of-week  =  "Monday"    / "Mon"  / "Tuesday"   / "Tue"
             /  "Wednesday" / "Wed"  / "Thursday"  / "Thu"
             /  "Friday"    / "Fri"  / "Saturday"  / "Sat"
             /  "Sunday"    / "Sun"
delimiters   =  specials / comment / linear-white-space
DIGIT        =  <any TELNET ASCII digit>

extension-field = <Any field which is defined in a document
                published as a formal extension to this
                specification>

field        =  field-name ":" [ field-body ] CRLF

fields       =  date-field  originator-fields  *optional-field
field-body   =  field-body-contents
                [CRLF LWSP-char field-body]
field-body-contents = <the TELNET ASCII characters making up the
                field-body, as defined in the following sections,
                and consisting of combinations of atom, quoted-
                string, and specials tokens, or else consisting of
                texts>
field-name   =  fnatom *(LWSP-char [fnatom])
fnatom       =  1*<any CHAR, excluding CTLs, SPACE, and ":">
```

```
host-indicator =  1*( ("at" / "@") node )
host-phrase =  phrase  host-indicator
hour        =  2DIGIT [":"] 2DIGIT [ [":"] 2DIGIT ]
HTAB        =  <TELNET ASCII horizontal-tab>

LF          =  <TELNET ASCII linefeed>
linear-white-space =  1*([CRLF] LWSP-char)
LWSP-char   = SPACE / HTAB

mach-id     =  "<" host-phrase ">"
mailbox     =  host-phrase /  (phrase mach-id)
message     =  fields *(CRLF *text)
month       =  "January"   / "Jan"  / "February"  / "Feb"
            /  "March"     / "Mar"  / "April"     / "Apr"
            /  "May"                / "June"      / "Jun"
            /  "July"      / "Jul"  / "August"    / "Aug"
            /  "September" / "Sep"  / "October"   / "Oct"
            /  "November"  / "Nov"  / "December"  / "Dec"

node        =  word / 1*DIGIT

optional-field  =
               "To"          ":" #address
            /  "cc"          ":" #address
            /  "bcc"         ":" #address
            /  "Subject"     ":" *text
            /  "Comments"    ":" *text
            /  "Message-ID" ":" mach-id
            /  "In-Reply-To""":" #(phrase / mach-id)
            /  "References" ":" #(phrase / mach-id)
            /  "Keywords"    ":" #phrase
            /  extension-field
            /  user-defined-field
originator-fields =
               (  "From"      ":" mailbox
                 ["Reply-To" ":" #address] )
            /  (  "From"      ":" 1#address
                  "Sender"    ":" mailbox
                 ["Reply-To" ":" #address] )

phrase      =  1*word

quoted-pair =  "\" CHAR
quoted-string =  <">  *(qtext / quoted-pair)  <">
qtext       =  <any CHAR except <">, CR, or LF and including
               linear-white-space>
SPACE       =  <TELNET ASCII space>
specials    =  "(" / ")" / "<" / ">" / "@"/ "," / ";" / ":"
            /  "\" / <">

text        =  <any CHAR, including bare CR and/or bare LF, but
               NOT including CRLF>
```

```
time          =  hour zone

user-defined-field = <Any field which has not been defined in
                  this specification or published as an extension to
                  this specification; names for such fields must be
                  unique and may be preempted by putlished
                  extensions>

word          =  atom / quoted-string

zone          = ( ("+" / "-") 4DIGIT )
              / ( ["-"] (1ALPHA
                / "GMT" / "NST"  / "AST" / "ADT" / "EST" / "EDT"
                / "CST" / "CDT"  / "MST" / "MDT" / "PST" / "PDT"
                / "YST" / "YDT"  / "HST" / "HDT" / "BST" / "BDT" ))

<">           =  <TELNET ASCII quote mark>
```

B. SIMPLE PARSING

Some mail-reading software systems may wish to perform only minimal processing, ignoring the internal syntax of structured field-bodies and treating them the same as unstructured-field-bodies. Such software will need only to distinguish:

- Header fields from the message body,
- Beginnings of fields from lines which continue fields,
- Field-names from field-contents.

The abbreviated set of syntactic rules which follows will suffice for this purpose. They describe a limited view of messages and are a subset of the syntactic rules provided in the main part of this specification. One small exception is that the contents of field-bodies consist only of text:

SYNTAX:

```
message      =  *field *(CRLF *text)

field        =  field-name ":" [field-body] CRLF

field-name   =  fnatom *( LWSP-char [fnatom] )

fnatom       =  1*<any CHAR, excluding CTLs, SPACE, and ":">

field-body   =  *text [CRLF LWSP-char field-body]
```

SEMANTICS:

Headers occur before the message body and are terminated by a null line (i.e., two contiguous CRLFs).

A line which continues a header field begins with a SPACE or HTAB character, while a line beginning a field starts with a printable character which is not a colon.

A field-name consists of one or more printable characters (excluding colon), each separated by one or more SPACES or HTABS. A field-name MUST be contained on one line. Upper and lower case are not distinguished when comparing field-names.

BIBLIOGRAPHY

ANSI. Representations of universal time, local time differentials, and United States time zone references for information interchange. ANSI X3.51-1975; American National Standards Institute: New York, 1975.

Bhushan, A.K. The File Transfer Protocol. ARPANET Request for Comments, No. 354, Network Information Center No. 10596; Augmentation Research Center, Stanford Research Institute: Menlo Park, July 1972.

Bhushan, A.K. Comments on the File Transfer Protocol. ARPANET Request for Comments, No. 385, Network Information Center No. 11357; Augmentation Research Center, Stanford Research Institute: Menlo Park, August 1972.

Bhushan, A.K., Pogran, K.T., Tomlinson, R.S., and White, J.E. Standardizing Network Mail Headers. ARPANET Request for Comments, No. 561, Network Information Center No. 18516; Augmentation Research Center, Stanford Research Institute: Menlo Park, September 1973.

Feinler, E.J. and Postel, J.B. ARPANET Protocol Handbook. Network Information Center No. 7104; Augmentation Research Center, Stanford Research Institute: Menlo Park, April 1976. (NTIS AD A003890).

McKenzie, A. File Transfer Protocol. ARPANET Request for Comments, No. 454, Network Information Center No. 14333; Augmentation Research Center, Stanford Research Institute: Menlo Park, February 1973.

McKenzie, A. TELNET Protocol Specification. Network Information Center No. 18639; Augmentation Research Center, Stanford Research Institute: Menlo Park, August 1973.

Myer, T.H. and Henderson, D.A. Message Transmission Protocol. ARPANET Request for Comments, No. 680, Network Information Center No. 32116; Augmentation Research Center, Stanford Research Institute: Menlo Park, 1975.

Neigus, N. File Transfer Protocol. ARPANET Request for Comments, No. 542, Network Information Center No. 17759; Augmentation Research Center, Stanford Research Institute: Menlo Park, July 1973.

Pogran, K., Vittal, J., Crocker, D. and Henderson, A. Proposed official standard for the format of ARPA network messages.

ARPANET Request for Comments, No. 724, Network Information Center No. 37435; Augmentation Research Center, Stanford Research Institute: Menlo Park, May 1977.

Postel, J.B. Revised FTP Reply Codes. ARPANET Request for Comments, No. 640, Network Information Center No. 30843; Augmentation Research Center, Stanford Research Institute: Menlo Park, June 1974.

Mark Crispin (SU-AI)
RFC 734, NIC 41953 (Oct. 7, 1977)

SUPDUP Protocol

INTRODUCTION

This document describes the SUPDUP protocol, a highly efficient display Telnet protocol. It originally started as a private protocol between the ITS systems at MIT to allow a user at any one of these systems to use one of the others as a display. At the current writing, SUPDUP user programs also exist for Data Disc and Datamedia displays at SU-AI and for Datamedias at SRI-KL. The author is not aware of any SUPDUP servers other than at the four MIT ITS sites.

The advantage of the SUPDUP protocol over an individual terminal's protocol is that SUPDUP defines a "virtual" or "software" display terminal that implements relevant cursor motion operations. The protocol is not built on any particular display terminal but rather on the set of functions common to all display terminals; hence it is completely device-independent. In addition, the protocol also provides for terminals which cannot handle certain operations, such as line or character insert/delete. In fact, it is more than this. It provides for terminals which are missing any set of features, all the way down to model 33 Teletypes.

The advantage over the TELNET protocol is that SUPDUP takes advantage of the full capabilities of display terminals, although it also has the ability to run printing terminals.

It is to be noted that SUPDUP operates independently from TELNET; it is not an option to the TELNET protocol. In addition, certain assumptions are made about the server and the user programs and their capabilities. Specifically, it is assumed that the operating system on a server host provides all the display-oriented features of ITS. However, a server may elect not to do certain display operations available in SUPDUP; the SUPDUP protocol is far-reaching enough so that the protocol allows terminals to be handled as well as that host can handle terminals in general. Of course, if a host does not support display terminals in any special way, there is no point in bothering to implement a SUPDUP server since TELNET will work just as well.

A more complete description of the display facilities of SUPDUP and ITS can be found by FTPing the online file .INFO.;ITS TTY from ARPANET host MIT-AI (host 206 octal, 134 decimal). For more

information, the mailing address for SUPDUP is "(BUG SUPDUP) at MIT-AI". If your mail system won't allow you to use parentheses, use Bug-SUPDUP@MIT-AI.

BACKGROUND

The SUPDUP protocol originated as the internal protocol used between parts of ITS, and between ITS and "intelligent" terminals. Over the network, a user host acts like an intelligent terminal programmed for ITS.

The way terminal output works in ITS is as follows: The user program tells the system to do various operations, such as printing characters, clearing the screen, moving the cursor, etc. These operations are formed into 8-bit characters (using the %TD codes described below) and stored into a buffer. At interrupt level, as the terminal demands output, characters are removed from the buffer and translated into terminal dependent codes. At this time padding and cursor motion optimization are also done.

In some cases, the interrupt side does not run on the same machine as the user program. SUPDUP terminals have their "interrupt side" running in the user host. When SUPDUP is run between two ITS's, the SUPDUP user and server programs and the network simply move characters from the buffer in the server machine to the buffer in the user machine. The interrupt side then runs on the user machine just as if the characters had been generated locally.

Due to the highly interactive characteristics of both the SUPDUP protocol and the ITS system, all transactions are strictly character at a time and all echoing is remote. In addition, all padding and cursor control optimization must be done by the user.

Because this is also the internals of ITS, the right to change it any time if necessary to provide new features is reserved by MIT. In particular, the initial negotiation is probably going to be changed to transmit additional variables, and additional %TD codes may be added at any time. User programs should ignore those they don't know about.

The following conventions are used in this document: function keys (ie, keys which represent a "function" rather than a "graphic character") are in upper case in square brackets. Prefix keys (ie, keys which generate no character but rather are held down while typing another character to modify that character) are in upper case in angle brackets. Hence "<CONTROL><META>[LINE FEED]" refers to the character generated when both the CONTROL and META keys are held down

while a LINE FEED is typed. Case should be noted; <CONTROL>A refers to a different character from <CONTROL>a. Finally, all numbers which do not explicitly specify a base (ie, octal or decimal) should be read as octal unless the number is immediately followed by a period, in which case it is decimal.

INITIALIZATION

The SUPDUP server listens on socket 137 octal. ICP proceeds in the normal way for establishing 8-bit connections. After the ICP is completed, the user side sends several parameters to the server side in the form of 36.-bit words. Each word is sent through the 8-bit connection as six 6-bit bytes, most-significant first. Each byte is in the low-order 6 bits of a character. The first word is the negative of the number of variables to follow in the high order 18. bits (the low-order 18. bits are ignored), followed by the values of the TCTYP, TTYOPT, TCMXV, TCMXH, and TTYROL terminal descriptor variables (these are the names they are known by at ITS sites). These variables are 36.-bit binary numbers and define the terminal characteristics for the virtual terminal at the REMOTE host.

The count is for future compatability. If more variables need to be sent in the future, the server should assume "reasonable" default values if the user does not specify them. PDP-10 fans will recognize the format of the count (ie, -count,,0) as being an AOBJN pointer. At the present writing there are five variables hence this word should be -5,,0.

The TCTYP variable defines the terminal type. It MUST be 7 (%TNSFW). Any other value is a violation of protocol.

The TTYOPT variable specifies what capabilities or options the user's terminal has. A bit being true implies that the terminal has this option. This variable also includes user options which the user may wish to alter at his or her own descretion; these options are included since they may be specified along with the terminal capabilities in the initial negotiation. See below for the relevant TTYOPT bits.

The TCMXV variable specifies the screen height in number of lines.

The TCMXH variable specifies the line width in number of characters. This value is one less than the screen width (ITS indicates line overflow by outputting an exclamation point at the end of the display line before moving to the next line). NOTE: the terminal must not do an automatic CRLF when a character is printed in the rightmost

column. If this is unavoidable, the user SUPDUP must decrement the width it sends by one.

NOTE: Setting either the TCMXV or TCMXH dimension greater than 128. will work, but will have some problems as coordinates are sometimes represented in only 7 bits. The main problems occur in the SUPDUP protocol when sending the cursor position after an output reset and in ITS user programs using the display position codes ^PH and ^PV.

The TTYROL variable specifies the "glitch count" when scrolling. This is the number of lines to scroll up when scrolling is required. If zero, the terminal is not capable of scrolling. 1 is the usual value, but some terminals glitch up by more than one line when they scroll.

Following the transmission of the terminal options by the user, the server should respond with an ASCII greeting message, terminated with a %TDNOP code (%TD codes are described below). All transmissions from the server after the %TDNOP are either printing characters or virtual terminal display codes.

The user and the server now both communicate using the intelligent terminal protocol (described below) from the user and %TD codes from the server. The user has two commands in addition to these; they are escaped by sending 300 (octal). If following the escape is a 301 (octal), the server should attempt to log off the remote job (generally this is sent immediately before the user disconnects, so this logout procedure should be done regardless of the continuing integrity of the connection). If the character following the escape is a 302 (octal), all ASCII characters following up to a null (000 octal) are interpreted as "console location" which the server can handle as it pleases. No carriage return or line feed should be in the console location text. Normally this is saved away to be displayed by the "who" command when other users ask where this user is located.

TTYOPT FUNCTION BITS

The relevant TTYOPT bits for SUPDUP usage follow. The values are given in octal, with the left and right 18-bit halves separated by ",," as in the usual PDP-10 convention.

Bit name	Value	Meaning
%TOALT	200000,,0	characters 175 and 176 are converted to altmode (033) on input.
%TOERS	40000,,0	this terminal is capable of selectively erasing its screen. That is, it supports the %TDEOL, the %TDDLF, and (optionally) the %TDEOF operations. For terminals which can only do single-character erasing, see %TOOVR.
%TOMVB	10000,,0	this terminal is capable of backspacing (ie, moving the cursor backwards).
%TOSAI	4000,,0	this terminal has the Stanford/ITS extended ASCII graphics character set.
%TOOVR	1000,,0	this terminal is capable of overprinting; if two characters are displayed in the same position, they will both be visible, rather than one replacing the other. Lack of this capability but the capability to backspace (see %TOMVB) implies that the terminal can do single character erasing by overstriking with a space. This allows terminals without the %TOERS capability to have display-style "rubout processing", as this capability depends upon either %TOERS or [%TOMVB and not %TOOVR].
%TOMVU	400,,0	this terminal is capable of moving the cursor upwards.
%TOLWR	20,,0	this terminal's keyboard is capable of generating lowercase characters; this bit is mostly provided for programs which want to know this information.

Bit name	Value	Meaning
%TOFCI	10,,0	this terminal's keyboard is capable of generating CONTROL and META characters as described below.
%TOLID	2,,0	this terminal is capable of doing line insert/delete operations, ie, it supports %TDILP and %TDDLP.
%TOCID	1,,0	this terminal is capable of doing character insert/delete operations, ie, it supports %TDICP and %TDDCP.
%TPCBS	0,,40	this terminal is using the "intelligent terminal protocol". THIS BIT MUST BE ON.
%TPORS	0,,10	the server should process output resets instead of ignoring them. IT IS HIGHLY RECOMMENDED THAT THIS BIT BE ON; OTHERWISE THERE MAY BE LARGE DELAYS IN ABORTING OUTPUT.

The following bits are user option bits. They may be set or not set at the user's discretion. The bits that are labelled "normally on" are those that are normally set on when a terminal is initialized (ie, by typing [CALL] on a local terminal).

Bit name	Value	Meaning
%TOCLC	100000,,0	convert lower-case input to upper case. Many terminals have a "shift lock" key which makes this option useless. NORMALLY OFF.
%TOSA1	2000,,0	characters 001-037 should be displayed using the Stanford/ITS extended ASCII graphics character set instead of uparrow followed by 100+character. NORMALLY OFF.
%TOMOR	200,,0	the system should provide "**MORE**" processing when the cursor reaches the bottom line of the screen. **MORE** processing is described in ITS TTY. NORMALLY ON.
%TOROL	100,,0	the terminal should scroll when attempting output below the bottom line of the screen instead of wrapping around to the top. NORMALLY OFF.

INPUT -- THE INTELLIGENT TERMINAL PROTOCOL

NOTE: only the parts of the intelligent terminal protocol relevant to SUPDUP are discussed here. For more information, read ITS TTY.

CHARACTER SETS

There are two character sets available for use with SUPDUP; the 7-bit character set of standard ASCII, and the 12-bit character set of extended ASCII. Extended ASCII has 5 high order or "bucky" bits on input and has graphics for octal 000-037 and 177 (see the section entitled "Stanford/ITS character set" for more details). The two character sets are identical on output since the protocol specifies that the host should never send the standard ASCII formatting characters (ie, TAB, LF, VT, FF, CR) as formatting characters; the characters whose octal values are the same as these formatting characters are never output unless the user job has these characters enabled (setting %TOSAI and %TOSA1 generally does this).

Input differs dramatically between the 7-bit and 12-bit character sets. In the 7-bit character set, all characters input whose value is 037 octal or less are assumed to be (ASCII) control characters. In the 12-bit character set, there are 5 "bucky" bits which may be attached to the character. The two most important of these are CONTROL and META, which form a 9-bit character set. TOP is used to distinguish between printing graphics in the extended character set and ASCII controls. The other two are reserved and should be ignored. Since both 7-bit and 12-bit terminals are commonly in use, 0001, 0301, and 0341 are considered to be <CONTROL>A on input by most programs, while 4001 is considered to be downwards arrow.

MAPPING BETWEEN CHARACTER SETS

Many programs and hosts do not process 12-bit input. In this case, 12-bit input is folded down to 7-bit as follows: TOP and META are discarded. If CONTROL is on, then if the 7-bit part of the character specifies a lower case alphabetic it is converted to upper case; then if the 7-bit part is between 077 and 137 the 100 bit is complemented or if the 7-bit part is 040 the 040 bit is subtracted (that's right, <CONTROL>? is converted to [RUBOUT] and <CONTROL>[SPACE] is converted to [NULL]). In any case the CONTROL bit is discarded, and the remainder is treated as a 7-bit ASCII character. It should be noted that in this case downwards arrow is read by the program as standard ASCII <CONTROL>A.

Servers which expect 12-bit input and are told to use the 7-bit character set should do appropriate unfolding from the 7-bit character set to 12-bit. It is up to the individual server to decide upon the unfolding scheme. On ITS, user programs that use the 12-bit character set generally have an alternative method for 7-bit; this often takes the form of prefix characters indicating that the next character should be "controllified" or "metized", etc.

BUCKY BITS

Under normal circumstances, characters input from the keyboard are sent to the foreign host as is. There are two exceptions; the first occurs when an octal 034 character is to be sent; it must be quoted by being sent twice, because 034 is used as an escape character for protocol commands. The second exception occurs when %TOFCI is set and a character with non-zero bucky bits is to be sent. In this case, the character, which is in the 12-bit form:

Name	Value	Description
%TXTOP	4000	This character has the [TOP] key depressed.
%TXSFL	2000	Reserved, must be zero.
%TXSFT	1000	Reserved, must be zero.
%TXMTA	400	This character has the [META] key depressed.
%TXCTL	200	This character has the [CONTROL] key depressed.
%TXASC	177	The ASCII portion of the character

is sent as three bytes. The first byte is always 034 octal (that is why 034 must be quoted). The next byte contains the "bucky bits", ie, the %TXTOP through %TXCTL bits, shifted over 7 bits (ie, %TXTOP becomes 20) with the 100 bit on. The third byte contains the %TXASC part of the character. Hence the character <CONTROL><META>[LINE FEED] is sent as 034 103 012.

OUTPUT RESETS

The intelligent terminal protocol also is involved when a network interrupt (INR/INS) is received by the user program. The user program should increment a count of received network interrupts when this happens. It should not do any output, and if possible abort any

output in progress, if this count is greater than zero (NOTE: the program MUST allow for the count to go less than zero).

Since the server no longer knows where the cursor is, it suspends all output until the user informs it of the cursor position. This also gives the server an idea of how much was thrown out in case it has to have some of the aborted output displayed at a later time. The user program does this when it receives a %TDORS from the server. When this happens it should decrement the "number of received network interrupts" count described in the previous paragraph and then send 034 followed by 020, the vertical position, and the horizontal position of where the cursor currently is located on the user's screen.

OUTPUT -- DISPLAY PROTOCOL (%TD CODES)

Display output is somewhat simpler. Codes less than 200 octal are printing characters and are displayed on the terminal (see the section describing the "Stanford/ITS character set"). Codes greater than or equal to 200 (octal) are known as "%TD codes", so called since their names begin with %TD. The %TD codes that are relevant to SUPDUP operation are listed here. Any other code received should be ignored, although a bug report might be sent to the server's maintainers. Note that the normal ASCII formatting characters (011 - 015) do NOT have their formatting sense under SUPDUP and should not occur at all unless the Stanford/ITS extended ASCII character set is in use (i.e., %TOSAI is set in the TTYOPT word).

For cursor positioning operations, the top left corner is (0,0), i.e., vertical position 0, horizontal position 0.

%TD code	Value	Meaning
%TDMOV	200	General cursor position code. Followed by four bytes; the first two are the "old" vertical and horizontal positions and may be ignored. The next two are the new vertical and horizontal positions. The cursor should be moved to this position. On printing consoles (non %TOMVU), the old vertical position may differ from the true vertical position; this can occur when scrolling. In this case, the user program should set its idea of the old vertical position to what the %TDMOV says and then proceed. Hence a %TDMOV with an old vpos of

20. and a new vpos of 22. should always move the "cursor" down two lines. This is used to prevent the vertical position from becoming infinite.

%TDMV1	201	An internal cursor motion code which should not be seen; but if it is, it has two argument bytes after it and should be treated the same as %TDMV0.
%TDEOF	202	Erase to end of screen. This is an optional function since many terminals do not support this. If the terminal does not support this function, it should be treated the same as %TDEOL. %TDEOF does an erase to end of line, then erases all lines lower on the screen than the cursor. The cursor does not move.
%TDEOL	203	Erase to end of line. This erases the character position the cursor is at and all positions to the right on the same line. The cursor does not move.
%TDDLF	204	Clear the character position the cursor is on. The cursor does not move.
%TDCRL	207	If the cursor is not on the bottom line of the screen, move cursor to the beginning of the next line and clear that line. If the cursor is at the bottom line, scroll up.
%TDNOP	210	No-op; should be ignored.
%TDORS	214	Output reset. This code serves as a data mark for aborting output much as IAC DM does in the ordinary TELNET protocol.
%TDQOT	215	Quotes the following character. This is used when sending 8-bit codes which are not %TD codes, for instance when loading programs into an intelligent terminal. The following character should be passed through intact to the terminal.

%TDFS	216	Non-destructive forward space. The cursor moves right one position; this code will not be sent at the end of a line.
%TDMV0	217	General cursor position code. Followed by two bytes; the new vertical and horizontal positions.
%TDCLR	220	Erase the screen. Home the cursor to the top left hand corner of the screen.
%TDBEL	221	Generate an audio tone, bell, whatever.
%TDILP	223	Insert blank lines at the cursor; followed by a byte containing a count of the number of blank lines to insert. The cursor is unmoved. The line the cursor is on and all lines below it move down; lines moved off the bottom of the screen are lost.
%TDDLP	224	Delete lines at the cursor; followed by a count. The cursor is unmoved. The first line deleted is the one the cursor is on. Lines below those deleted move up. Newly-created lines at the bottom of the screen are blank.
%TDICP	225	Insert blank character positions at the cursor; followed by a count. The cursor is unmoved. The character the cursor is on and all characters to the right on the current line move to the right; characters moved off the end of the line are lost.
%TDDCP	226	Delete characters at the cursor; followed by a count. The cursor is unmoved. The first character deleted is the one the cursor is on. Newly-created characters at the end of the line are blank.
%TDBOW	227	Display black characters on white screen. HIGHLY OPTIONAL.
%TDRST	230	Reset %TDBOW and such any future options.

STANFORD/ITS CHARACTER SET

This section describes the extended ASCII character set. It originated with the character set developed at SAIL but was modified for 1968 ASCII.

This character set only applies to terminals with the %TOSAI and %TOFCI bits set in its TTYOPT word. For non-%TOSAI terminals, the standard ASCII printing characters are the only available output characters. For non-%TOFCI terminals, the standard ASCII characters are the only available input characters.

PRINTING CHARACTERS

The first table describes the printing characters. For output, the 7-bit code is sent (terminal operations are performed by %TD codes). For input, the characters with values 000-037 and 177 must have the %TXTOP bit on to indicate the graphic is intended rather than a function or ASCII control.

Value	Character
4000	centered dot
4001	downward arrow
4002	alpha
4003	beta
4004	logical AND
4005	logical NOT
4006	epsilon
4007	pi
4010	lambda
4011	gamma
4012	delta
4013	uparrow
4014	plus-minus
4015	circle-plus
4016	infinity
4017	partial delta
4020	proper subset (left horseshoe)
4021	proper superset (right horseshoe)
4022	intersection (up horseshoe)
4023	union (downward horseshoe)
4024	universal quantifier
4025	existential quantifier
4026	circle-x
4027	double arrow
4030	left arrow

Value	Character
4031	right arrow
4032	not-equal
4033	lozenge (diamond)
4034	less-than-or-equal
4035	greater-than-or-equal
4036	equivalence
4037	logical OR
0040	first standard ASCII character (space)
..	. . .
0176	last standard ASCII character (tilde)
4177	integral

FUNCTION KEYS AND SPECIAL CHARACTERS

In addition, the following special characters exist for input only. These characters are function keys rather than printing characters; however, some of these characters have some format effect or graphic which they echo as; the host, not the SUPDUP program, handles any such mappings.

Value	Character	Usual echo	Usual Function
0000	[NULL]		
0010	[BACK SPACE]		text formatting
0011	[TAB]		text formatting
0012	[LINE FEED]		text formatting
0013	[VT]		text formatting
0014	[FORM]		text formatting
0015	[RETURN]		text formatting
0032	[CALL]	uparrow-Z	escape to system
0033	[ALTMODE]	lozenge or $	special activation
0037	[BACK NEXT]	uparrow-underscore	monitor command prefix
0177	[RUBOUT]		character delete
4101	[ESCAPE]		local terminal command
4102	[BREAK]		local subsystem escape
4103	[CLEAR]		
4110	[HELP]		requests a help message

BUCKY BITS

For all input characters, the following "bucky bits" may be added to the character. Their interpretation depends entirely upon the host. <TOP> is not listed here, as it has been considered part of the character in the previous tables.

<CONTROL> is different from ASCII CTRL, however, many programs may request the operating system to map such characters to the ASCII forms (with the <TOP> bit off). In this case <META> is ignored.

Value	Key
2000	Reserved
1000	Reserved
0400	<META>
0200	<CONTROL>

ACKNOWLEDGEMENTS

Richard M. Stallman (RMS@MIT-AI) and David A. Moon (Moon@MIT-MC) of the MIT-AI and MIT-MC systems staff wrote the source documentation and the wonderful ITS terminal support that made this protocol possible. It must be emphasized that this is a functional protocol which has been in operation for some years now.

In addition, Moon, Stallman, and Michael McMahon (MMcM@SRI-KL) provided many helpful comments and corrections to this document.

For further reference, the sources for the known currently existing SUPDUP user programs are available online as:

```
[MIT-AI] SYSENG;SUPDUP >        for the ITS monitor,
[SU-AI] SUPDUP.MID[NET,MRC]     for the SAIL monitor,
[SRI-KL] <MMcM>SD.FAI           for the TOPS-20 monitor.
```

The source for the known currently existing SUPDUP server program is:

```
[MIT-AI] SYSENG;TELSER >        for the ITS monitor.
```

These programs are written in the MIDAS and FAIL dialects of PDP-10 assembly language.

Mark Crispin (SU-AI)
RFC 736, NIC 42213 (Oct. 31, 1977)

TELNET SUPDUP Option

1. Command name and code.

   SUPDUP 21

2. Command meanings.

   IAC WILL SUPDUP

      The sender of this command REQUESTS permission to, or confirms that it will, use the SUPDUP display protocol

   IAC WON'T SUPDUP

      The sender of this command REFUSES to use the SUPDUP protocol.

   IAC DO SUPDUP

      The sender of this command REQUESTS that the receiver use, or grants the receiver permission to use, the SUPDUP protocol.

   IAC DON'T

      The sender of this command DEMANDS that the receiver not use the SUPDUP protocol.

3. Default.

   WON'T SUPDUP

   DON'T SUPDUP

   i.e., the SUPDUP display protocol is not in use.

4. Motivation for the option.

Since the publication of RFC 734, I have been requested to design an option to the TELNET protocol to provide for SUPDUP service. This option allows a host to provide SUPDUP service on the normal TELNET socket (27 octal) instead of 137 (octal) which is the normal SUPDUP ICP socket.

5. Description of the option.

A user TELNET program which wishes to use the SUPDUP display protocol instead of the NVT terminal service should send an IAC DO SUPDUP. If the server is willing to use the SUPDUP display protocol, it should respond with IAC WILL SUPDUP; otherwise it should refuse with IAC WONT SUPDUP.

For hosts which normally provide SUPDUP terminal services, the server can send IAC WILL SUPDUP upon ICP which the user may then accept or refuse.

If the SUPDUP option is in effect, no further TELNET negotiations are allowed. They are meaningless, since SUPDUP has its own facilities to perform the functions that are needed. Hence, octal 377 will become an ordinary transmitted character (in this case an invalid %TD code) instead of an IAC.

Following the mutual acceptance of the SUPDUP option, the SUPDUP negotiation proceeds as described in RFC 734.

< AJOURNAL, 42395.NLS.1, >, 17-Nov-77 21:34 XXX ;;;; .HJOURNAL="DVN 17-Nov-77 21:32 42395"; Title: .H1="ROUGH Draft Conceptual Design for User's Views and Commands in Innovations in Engineering Publication."; Author(s): Dirk H. Van Nouhuys/DVN; Distribution: /EKM([ACTION]) JHB([ACTION]) INPUBS([INFO-ONLY]) DCE([INFO-ONLY]) KIRK([INFO-ONLY]) HGL([INFO-ONLY]) JBP([INFO-ONLY]) DSM([INFO-ONLY]) ; Sub-Collections: SRI-ARC INPUBS; Clerk: DVN; .IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PES;

This is a conceptual sketch of how to implement (29248,1c1a) and (29248,1c2c3,). It is intended mostly as a straw person for us to work against, and probably asks for more than we can do. It's mostly in the form of a rough command syntax with notes because that's the way I found it easiest to think.

Reader stuff

Sketch of Subsystems and Commands

Browse subsystem

Brief Description: This subsystem allows the user to read specific issues of specitic publications, or a range of issues, with clipt or full views, and filtered by keyword or by previous reading history. The system will tell her at login time if there are any unread issue or new items that pass her keywords.

Fetch Datalog LSEL (through) OK/LSEL OK

These LSEL's Accept Dates. I imagine each date's issue being one file. When the user asked for a span of dates an Include mechanism would get her all the right stuff as filtered.

Fetch Aerospace LSEL (through) OK/LSEL OK

Fetch EEIE LSEL (through) OK/LSEL OK

Fetch Tables (of Contents) (through) OK/LSEL OK

Show Headers (only)/ Oneline (only)/Full OK

Plugin Keywords All OK/BUG BUG (finished) ANSWER OK

This command activates the keyword filter, if the option is All, the the filter uses all her keywords. If the users choose bug and has Fetched her list of keywords she may select those she wants to use this time as a series of groups.

Unplug Keywords OK

Turns off the keyword filters

Trace (my reads) OK

Causes the system to display all the items in the feteched domain that the user has previously read in full view Filtering for Keyword should take pace before filtering for trace.

Smooth (my trace) OK

turns off the Trace function

Fetch Keywords OK

(loads and displays the plex where the keywords live)

Add Keyword LSEL OK

To add to your list of keywords.

Delete Keyword BUG BUG

To remove keywords, the bugs spacify a group.

Sort Keywords OK

(alphabetacally)

Jump (to) BUG
These Jump commands do not allow you to change Viewspecs.
Jump (to) Content LSEL OK
Jump Next OK
Jump Back OK
Goto Exchange
Help
Goto Note
Logout
Problems and issues: Do we want to allow them to do Boolean stuff with the keywords?

Note Subsystem
Brief description: This subsystem allows users to write messages (or other notes), and read and edit them primitively.
Jump (to) BUG
Logout
Jump (to) Content LSEL OK
Jump Next OK
Jump Back OK
Goto Exchange
Goto Browse
Help
Jot Item (in scratch file) LSEL DSEL OK
= Insert Statement
Erase Item (from scratch file) DSEL OK
Erase Item Since LSEL OK
The system would warn people when they log in that they were approaching the limit of their file size and they could throw away old branches this way. The LSEL would take a date or time and date. Logout would call updating and trimming functions.
Repace Word DSEL OK
This would be the editing command. the "Word" would be a visible but would handle spaces and punctuation the way "word" does.
Problems: The editing is very primative, that is some one who wants to change a whole note has to retype it, but I think we want to keep this simple. Maybe we don't want to let them writ at all except the test of a message.

Exchange Subsystem
Brief description: This system allows the users to publicize their interest in subjects and exchange messages with people who share their interests. It allows them to record their interest in things or hide it or hide from everyone but their organization. Messages are delivered to a "messages" branch in the notes file and read and erased with that subsystem. When the user logs in, she is notified if there is a message for her.
Send Note (item) OK
(To:) LSEL
(and cc) LSEL
(Subject:) LSEL
(Message) MSGITEM
---Status of the distribution is displayed here--- (Send the message? (Type n to add to list)) ANSWER OK
MSGITEM = Message CONTENT or

Statement/Branch/Group/Plex DESTINATION VIEWSPECS OK
The first 2 LSEL's would accept the names of users or the number (or whatever we have in the way of a unique identifier) of an item in which case the message would go to all the people who had marked their reads public, or a keyword, in which case the message would go to everyone who had that keyword.

Mark (my reads) Private/ Public/ Group/ BUG/All OK.
The user would have one of the publications loaded to use this command; The dates and filters then in force would determine its span. This command allows a user to control whether the send message command can deliver messages on the basis of whether this user has read that item in full view. Group means the people in the organization she works for.

Goto Note
Goto Browse
Help
Logout

Problems: There are obviously some problems in data base manipulation and delivery time if the system is to translate names, reads of items, and keywords into delivery to individuals which involves the question of whether we want individual users to have directories etc.

Login must be very simple, e, g. =
@ INPUBS Username<CR>

Background Concepts

Necessary Views
ct
w
i
K
Title only Only
All items containing the current keywords.
All the items I have read before.

The system will record all the users commands and what publication files he seen and which she has studied with a filtered or a full view.

Some one will have to find a way to comprehend all this data. I assume we will have different publications in different files, probably in several files, probably one file per issue. We need an include function to see the several files of each publication as one, and something that shows the title and date of the publications up in the corner while the user is looking at different items. title and date of publications.

If a user does not log in for a couple of days it would be nice to have an automatic system print out the names of new items that pass her keywords and mail them to her.

Publisher Stuff

This is much easier to think about, the hard part is training and working out procedures They need the full range of INLS, plus off-line input and printout, plus an easy way to get the offline stuff in and out of NLS. That includes special training in whatever terminal and local storage we get them and in the output processor. In addition they need a formatter for their publication.

< AJOURNAL, 42403.NLS.1, >, 18-Nov-77 15:03 XXX ;;;; .HJOURNAL="RJC 18-Nov-77 10:09 42403"; Title: .H1="TIME Magazine Article"; Author(s): Roberta J. Carrier/RJC; Distribution: /DLS([ACTION]) ; Sub-Collections: NIC; Clerk: RJC; .IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PES; Origin: < CARRIER, TIMES.NLS.1, >, 18-Nov-77 08:47 RJC ;;;; .LBS=1; .PES; .PN=0; .NUMDASH=0; ####;

THE OFFICE OF TOMORROW CAN BE HERE TODAY!.CENTER;.GCR=3;
BY DR. VINCENT E. GIULIANO.CENTER;.GCR=3;
MOST OF WHAT DR. GIULIANO ENVISIONS IS ALREADY POSSIBLE, INDEED IN PLACE IN SOME LOCATIONS. THE REST IS JUST AROUND THE CORNER. HE SAYS NOW IS THE TIME TO START YOUR OFFICE INFORMATION SYSTEMS BALL ROLLING - OR YOU MAY BE PLAYING CATCH-UP IN THE FUTURE..GCR=3;
Dr. Vincent Giuliani is a member of the staff of Arthur D. Little Inc. He has been involved with word processing methods since 1952 and has authored some 60 publications and reports, 12 films plus videotape and mixed media presentations on the subject..PES;
Present approaches to office automation, spearheaded by word processing and limited use of specialized minicomputers, are just the beginning of an avalanche that's going to drastically change office systems over the next 10 to 20 years.

A WHOLE HOST OF NEW TECHNOLOGIES

The changes required go far beyond word processing as it is now envisioned. They involve totally integrated systems for written information entry, editing, storage, retrieval and transmission. The Office of Tomorrow technology will be based upon a massive extension of existing data processing networks, with an office systems terminal placed on everyone's desk - executive and secretary alike. Office correspondence, management information, public and commercial data base information will all be handled within a single data and communications network through those terminals.

New technologies such as low cost, flat video display terminals that weigh less than two pounds and fit in a briefcase will facilitate this chain of development. This "office in the briefcase" will make one's files and information management resources available wherever there is a telephone. Massive stakes are involved, for the national economy, for large companies, for office equipment manufacturers and ultimately for those who do white-collar or office work.

The change process is underway now propelled by social, economic and technical driving forces. What is uncertain is the nature and timing of the transition, exactly what technology will be available at what cost and when.

A VIDEO TERMINAL ON EVERY DESK...

The office information terminal is the most visible symbol of the Office of Tomorrow, so consider its role in a scenario which will be true in most offices in the not too distant future.

There is one terminal on every desk for input, output, for word processing, data processing, data base access, personal file searching and all the other aspects of personal information management. The terminal looks a lot like one of today's video display computer terminals, but will probably have a flat display that can handle alphanumeric and limited graphics. Most likely, it will be the size of a normal sheet of office paper. It will have a conventional typewriter keyboard plus special function keys for things like paging backwards or forwards through text, generating

hard-copy printouts, etc. The terminal will be networked with local and distant minicomputers and maxicomputers, and with other terminals via a completely switchable digital network comparable to, or possibly an extension of today's telephone network. The terminal will embody a very powerful processor on a chip (one that has a capability that would have cost a million dollars a few years ago) and have a significant local memory.

...WITH ACCESS TO MILLIONS OF OTHER TERMINALS

The terminal will provide access to literally millions of like terminals, each with its own internal computer on a chip, as well as to thousands of more specialized and more powerful computers which manage larger data bases. The networked terminal system will offer capabilities for:

Instantaneous electronic aid. When you type a message along with a destination code (which could be a telephone number anywhere) and hit the "release key," the message appears on the electronic "mailbox" of the recipient, i.e., a light will flash on his terminal saying that new mail has arrived, which can be read at his or her convenience.

Electronic information retrieval. The network will switch you to the data base that has the answers you need, in your office or in Zurich. This means access to your company's management information data bases, as well as access to government and commercial on-line data base services - New York Times Data Bank, the Lexis bank of legal case history information, and dozens more like them.

A hierarchical file structure for personal, group, project and corporate correspondence and reports. Some of your personal files will be kept locally, at the site of the terminal itself, perhaps on a removable magnetic "floppy disk."

Automatic full text indexing. Your local office computer will automatically index a letter or report by all the words contained in it.

Powerful search and retrieval commands will be available (something like, "get me the last letter that mentions the Smith Account"). The system will not be fussy, and will try searching even when the information is ambiguous or incomplete.

Multiple privilege levels for access. Various passwords and checks will insure privacy. Access privileges will vary by file, i.e., you may give someone else permission to read an item or part of it, and revoke that permission at will.

Personal scheduling and reminder aids. Your local office computer will "learn" your habits - with your help, and prompt you accordingly. It will, through automatic programming, learn about when you are likely to be forgetful, and will prod you gently to meet important deadlines. It will even remember birthdays.

Executive-secretary communications. Terminals possibly 4,000 miles apart will link you and your secretary. Long-haul telecommunications will be via public "value added" data networks which draw on earth satellites for low cost interconnection.

Interfacing with mail and paper systems. The old paper-based systems will be around for a long time. The office information system will have optional character recognition to handle incoming typed mail as well as printer and envelope addressing capabilities to generate outgoing mail when necessary.

Twenty-four hour availability. The office information system never sleeps.

Computer-voice conferencing. While you talk with a colleague by

phone, you will also be able to flash messages back and forth on your terminals, and modify these jointly as your discussion progresses.

SCIENCE FICTION? NO, THERE ARE INSTALLATIONS IN PLACE

There are a few installations where selected number of managers and professionals have had access to network terminal-based systems that have done most of the above for several years now. One place where this is happening is in the research laboratories of certain of the big office equipment manufacturers. However, the exact nature of their activities is a closely held proprietary secret. Perhaps the best known laboratory for the Office of Tomorrow is a research-oriented government project which provided the kinds of service described above (along with more conventional computer capabilities) to a few hundred users in the MIT university community. In the early 1970's the project became part of a far more extensive and ambitious government-sponsored undertaking, which networked several dozen computer facilities at various universities and research laboratories throughout the U.S. Each such utility has its own community of users, and the system routinely delivers tens of thousands of electronic mail messages between them daily.

WE ALREADY KNOW MUCH OF WHAT THE OFFICE OF TOMORROW WILL BE LIKE TODAY

To the extent these projects and similar experiments are examples of the Office of Tomorrow's information system, some inferences can be drawn from experience.

USERS LOVE IT

Users love it; would not want to give it up.

Messages tend to be shorter and crisper than in normal correspondence, since there is a possibility of several rounds of electronic mail interaction in a single day.

Professionals don't mind doing some typing. This, perhaps, has to do with the shorter messages and the ease of correction and editing.

Work can take place 16 hours a day, 7 days a week. Many individuals take terminals home, on trips or on vacations.

The technologies developed at great cost are now widely available. For instance, advanced multi-terminal systems are already in operation in over 100 newspapers. These systems capture the story as it is written by the reporter or comes off the press wire, and enable the page and section editors, the rewrite man and the daily editors to revise or further edit from terminals without further retyping or paper movement. TIME - the magazine which you are now reading - uses one form of such a system.

THE PACE IS ACCELERATING

The pace of new product announcements that will lead to the all-electronic office is steadily accelerating. Hardly a month goes by without new display or shared-logic word processor announcements, new high-speed printer developments, etc. The major manufacturers have yet to fire their really big guns, however, in terms of integrated office information systems that make for less, not more paper. In 3 - 5 years, however, it is likely that the present pace of change will seem slow and secretary-oriented word processing will seem almost commonplace.

WILL THE POST OFFICE GO THE WAY OF THE HORSE & BUGGY?

Cost is one of the major continuing driving forces for office automation. It stems from rising costs of paper systems and old ways of doing business associated with them, and continually dropping costs of telecommunications and electronic systems. The

major U.S. paper based institution is the U.S. Postal Service. A limit seems to have been reached, both human and technological, in the handling of mail. Despite continuing automation, postage costs in the last ten years have outstripped inflation by a large factor and will continue to go up - and service seems to be deteriorating. Electronic alternatives are increasingly providing a faster, cheaper, and better way to handle most of the communications now sent through the mail.

$250,000 COMPUTER POWER FOR ONLY A FEW DOLLARS A CHIP

A computer chip central processor unit two millimeters by two millimeters in size - 4 will fit on a fingernail - goes for less than $12 today. Ten years ago a machine of this power cost $250,000. In a couple of years the cost will be down to a dollar or so. Already, the connectors required to interface the chip with the other equipment cost far more than the processor itself. The long period of cost decline in electronic information technology started in the '40's and appears to still have a long way to go. In five or ten years, enough solid state "bubble" memory (information is stored in tiny magnetic "bubbles") to store a full page of text can be expected to cost far less than a penny.

PERIPHERAL COSTS WILL DECLINE

What will cost money in the very near future is not electronic hardware technology. It is the packaging of hardware into systems, the software (programmed instruction sets) required to drive the hardware, the electro-mechanical peripherals, such as printing or typing heads, and the marketing of systems that will continue to be expensive. With the development of bubble memories replacing rotating disks and new flat video terminals replacing moving typing heads, these peripheral costs will steadily decline in the next few hears. Long haul communications costs are already relatively negligible in big computer networks. We have grown used to seeing an automobile rental form completed by a computer thousands of miles away. Satellites will reduce communication costs further, although communications will remain very big business.

WHY NOT AUTOMATION ALREADY?

Given these powerful driving forces and the fact that the technology is already here, the question remains, why don't we now have highly sophisticated all-electronic offices?

Here are some of the reasons:

1. The way corporations are set up. Office automation cuts across several lines of authority. It involves technology, office services, financial planning and, depending on the company, many other areas of responsibility. Frequently no one officer is clearly in charge.
2. Inertia. Organizations and human habits would be profoundly affected by instantaneous communications, executives manning their own files from wherever they may be, Executives do not want to lose their secretaries; many executives see typing as below them.
3. Long-range vision called for. Many companies have too narrow and immediate a view of economics. Will an automated office show payout in the next year's P and L?
4. Technology still new. Certain of the necessary software and systems components are comparatively new to the market (e.g., full text-oriented data base management software).

5. Increased executive productivity is a benefit of the Office of Tomorrow not yet fully understood. So far, most of the focus has been on the near-term benefits of word processing, secretarial automation. How the system will increase the productivity of executives is still hard to grasp for managements rooted in traditional office systems.

WHAT CAN A COMPANY DO NOW?

Begin a prototype now. Large companies, having high stakes in the area of white collar productivity, should begin a prototype office information system project now. This is not so much in the interest of short-term cost savings (which can be very real, in fact), but more in the interest of getting the organizational learning and sophistication required to enable good decisions to be made when, 2 - 4 years from now, the big manufacturers wheel out their "do everything" integrated systems.

Doing it right requires learning. Major companies should responsibly start now with a prototype project.

Start with a full capability terminal-based office information system. This first system should not be just a secretary-oriented word processing system no matter how sophisticated. It should be a full capability terminal-based office information system that links a limited number of top level decision-making executives and their secretaries.

Use familiar components. The interim system must be made up of computer components familiar to EDP people. No technological break-throughs are required, just further evolution and software specialization of what exists.

THE NEAR FUTURE WILL BRING

Integrated office EDP systems with 1,000-4,000 terminals in an office complex.

Shift of information from paper to electronic form. Information "float" eliminated, i.e., present 1-3 days delay in correspondence handling eliminated.

Mail room, postage, filing, copying, secretarial search, message center costs reduced significantly.

Most business mail will short-circuit the post office via electronic networks.

More and more executives doing more and more of their information management.

Role of secretary evolving toward that of a junior professional.

Only top executives will have secretaries in the traditional role.

Messages shorter; almost everyone types; typing is not stigmatized.

New and more efficient keyboards being introduced.

Central facilities using computer stenotype translation, limited computer-voice recognition employed for voice-to-digital translation.

16-hour span of flexible working time more commonplace; much executive information management while at home, on travel, etc.

More and more mobile executives, not rooted in any fixed location.

Corporate systems linked via public digital networks to each other, to government and commercial information banks.

WILL THE OFFICE SURVIVE?

One final question. Will the office survive the onslaught of the new technology? The answer is definitely yes - as a place where human interactions take place, where trust is established, deals made,

interactions with colleagues enjoyed, tough problems solved, where joint work takes place around terminal-inlaid conference tables, where personnel work stations are located. The environment of the Office of Tomorrow is another question. One thing is certain, however, it will be better engineered for human communications, more humane and individually customized, and less structured by a need for manufacturing, moving, and filing large quantities of paperwork.

SEMINAR ON THE AUGMENTED KNOWLEDGE WORKSHOP
STANFORD RESEARCH INSTITUTE
Menlo Park, Room J2022
*** Agenda: 28 November - 2 December 1977 ***

MONDAY

9:00 Overview of the week's activities (Bair)

*Introduction

*Introduction of participants and their organizations

10:00 Background (Management)

*SRI Services and activities related to the AKW Concept

*The future of the AKW and NLS

10:30 Guest Speaker (Alan Purchase, SRI)

*Economic and design considerations for Office Information Systems

11:45 Lunch

1:00 An Approach to the Office of the Future (Bair)

*Problems and needs

*Potential solutions

*Problems with the solutions

2:30 System Architecture (Bair)

*The computer environment in an AKW

*DEX: Deferred EXecution
*TNLS: Teletype oNLine System
*DNLS: Display oNLine System

3:00 Experience with NLS (Weinberg)

*Use of the interface devices

*Command syntax and command word alternatives

*Displaying information by pointing and controlling the view

*The organization of information in NLS (structure, files)

TUESDAY

9:00	Application Community	(Norton)
	*Current client community	
	*The "architect" community	
10:00	Case Study: Complete Office Automation	(Lieberman)
	*National Science Foundation	
11:00	Application Principles	(Bair)
	*Human-computer communication design model	
	*NLS system design: human-computer communication optimization	
12:00	Lunch	
1:15	Application Principles	(Bair)
	*Information structured for logical power and flexibility	
	*Composing and editing textual information	
	*Information structuring, filtering, sorting, and keying	
2:15	Experience with NLS	(Weinberg)
	*Intra-file editing	
	*Multifile editing	
	*Entering text: structuring information	
	*Links	

WEDNESDAY

9:00 Application Principles (Bair)

*Principles for successful system implementation and usage

*Steps of innovation transfer

*Elements of the innovation transfer strategy

10:30 Case Study: Telecommunication Support of a Distributed Organization

*Army Development & Readiness Command (DARCOM) (Lieberman)

*Design of the ELITE System for project manager support (Maynard)

12:00 Lunch

1:15 Applications: Emphasis on Communication (Bair)

*Sending, receiving, and managing communications

*Automatic handling of communications

*Demonstration of shared screen teleconferencing

2:30 Experience with NLS (Weinberg)

*Sendmail, Journal, and Sndmsg

*Subsystems

THURSDAY

9:00 Application: Document Production and Control Systems (Vannouhuys)

*The document production cycle

*The ideal system: features and capabilities

*The prototype system: NLS

*Printing and COM (Computer Output to Microform)

10:45 Case Study: Document Production Application (Weinberg)

*Air Force Data Services Design Center (Gunter AFS)

12:00 Lunch

1:15 Application Principles and Subsystems (Norton)

*Distributed editing

*Special purpose subsystems for editing and other applications

*Demonstration of NLS Proof and Graphics subsystems

2:30 Experience with NLS (Weinberg)

*Output Processor and Directives

*Automatic Format, Letter, and Modify programs

*Filtered viewing of information: Content Analyzer

*Typewriter NLS session

FRIDAY

9:00	AKW Overview	(Engelbart)
	*The evolution of large and complex systems	
10:30	Application Review	(Bair)
	*Impact of computer message systems	
	*NLS impact from empirical studies	
	*Applications models	
12:00	End of Formal Seminar	

SEMINAR ON THE AUGMENTED KNOWLEDGE WORKSHOP

Major Themes from the Seminar on the Office of the Future from SRI International

Definition of the Augmented Knowledge Workshop (AKW)

The special name, "Augmented Knowledge Workshop," has been given to our approach to the office of the future. In this approach, the users are more important than the technology. The technology includes the hardware, software, and data networks that are the basis of the second computer revolution. The user system includes the organization, work flow, management, innovation transfer, attitudes, policies, and productivity changes. The result of this approach, the Workshop, is a complete environment where the latest technology is available to be used with new management techniques and working methods. Since there will continue to be new technology, techniques and methods, the Workshop must evolve to take advantage of these innovations.

The Impact of the Augmented Knowledge Workshop on Individuals, Groups, and Organizations

Much has been learned in the fifteen years of operating an AKW about how to build a Workshop, how to help the new users of the technology, and what are the effects on the users and their organizations. The Workshop must be built set-by-step, following a carefully planned strategy that includes persons assigned to special roles such as that of the "architect." Comprehensive and intensive training, documentation, assistance, and planning-consultation must be continually provided. Several studies of the effects of the Workshop show that increased efficiency can be achieved in interpersonal communication, document production, information management, and other areas. For example, scientists and engineers can collaborate from several different locations and create a shared workspace. The studies also show that the steps outlined above for building and providing continuing assistance need to be followed.

An Approach to the Design of the Office Information System

The technology system in an AKW is designed for the optimum communication between the user and the machine. The design of this interface between user and machine addresses several problems inherent in the Workshop, including: the wide spectrum of users, the variety and diversity of services that need to be provided, heavy usage, different kinds of hardware and software, and rapid changes in the technology and work methods. The solution is a natural command language for all users, services, and technologies. This language is used at television like displays which enable the user to point to information in special function "windows." The language and windows are

J H Bair

---

maintained in local machines that can automatically make connections to the different hardware and software the user needs. This solution to the problems inherent in the Workshop has certain tradeoffs, for example, increased complexity.

NLS (oNLine System): A Prototype of the Office Information System Possible Now

NLS is the software component of the technology system in an AKW. The approach to building this large software system has resulted in a comprehensive set of computer-based tools that can improve the power people have to handle information efficiently. As an operational office automation system, NLS is now being applied to document production, electronic communication, personal and organizational information processing, and software engineering. A number of organizations have implemented NLS for applications such as project coordination, decision formulation and communication, software development, remote collaboration, the generation and filing of letters, reports, and proposals, and the organization, scheduling and coordination of users' daily work routines. NLS is currently available through a new utility service.

J H Bair

BUILDING THE PROJECT SYSTEM IN THE NLS ENVIRONMENT: An Evaluation

Lawrence L. Garlick

December 16, 1977

Augmentation Research Center
SRI International
Menlo Park, California

Introduction

This document is an evaluation of the appropriateness of using NLS and its associated software development tools to build the Project System. The criteria used in this evaluation are the ability to satisfy the specific design requirements of the Project System and the ability to develop user-oriented systems that are large, reliable, flexible, responsive, and easy to maintain.

The Project System is a tool for planning and managing projects. Using the familiar medium of reports, project leaders and upper management plan and track the progress and resources of each project. The system generates reports suitable for each level of the management structure. The Project System also provides access paths to the project data that is dependent on the orientation of the user.

For details about the design of the SRI implementation of this system, see (office-1,wey-arc,sri-final,).

Meeting Capabilities Requirements

This section describes how NLS was used to satisfy the functional and implementational requirements of the Project System. It summarizes the issues and methods used in building a report framework. The problems of information representation are discussed, followed by techniques for integrating information structures and reports.

Information Structure

One of the problems that has continually plagued the designers of this system is the fuzzy distinction between information structuring and data base management. While NLS is not a general data base management system, it has many ways in which to structure information. Unlike a data base management system, the information is stored primarily as structure and text.

NLS is well known for its hierarchical representation of documents. As a result, NLS is sometimes viewed as being inflexible with more general information structures. In addition to the hierarchical tree structure, arbitrarily complex structures (graphs) can be supported through NLS links.

An important consideration is not only whether a particular structure can be supported, but how the performance of the system will be affected if a particular structure is chosen. The most efficient NLS address is relative addressing by structure, specifically the "down" or "successor" from the current location. But the next most efficient is statement name look-up, which uses a hashing algorithm on an inverted statement name list. The point to be made is that efficient mechanisms do exist to support arbitrary information structure and that the information does not have to be hierarchical to be handled efficiently by NLS.

Data Base Management Support

NLS is not a data base management system. The Project System data base was implemented by storing all non-textual elements as strings and simulating a data base management system. This has several advantages but is highly inefficient.

The advantages stem from the fact that the data base can be viewed as text. All the standard NLS commands are

available for manipulation of the data base. This was extremely useful for the implementation of the Project System during system debugging and the initial loading of the data base before data input capabilities were available. To reduce the probability of data being manipulated by accident, file access protection is used.

The inefficiencies arise mainly from number to string conversion and searching for fields (data items) in statements. In rough measurements performed during development, about 75% of the time spent generating a new report was attributed to data base retrievals. These inefficiencies could be overcome if a real DBMS is used.

Textual Information

The bulk of the information handling in the Projects System is the generation, display, and manipulation of reports. The editing of free-text reports requires flexible commands to manipulate textual and structural entities. Formatting of reports from data base elements also requires flexible text manipulation facilities.

The development environment includes string handling capabilities at two distinct levels, the L10 language and the procedures that implement the NLS system. The L10 language has powerful string construction and pattern matching capabilities built into it. Since the Project System is run as an NLS subsystem, both high-level routines and low-level support routines that support the NLS text editing functions are available.

In addition to the text manipulation capabilities, there must be some method of filing and retrieving reports that is efficient and dynamic (since the reports are themselves variable in length and constantly changing). Using NLS hierarchy, it is a simple and efficient to file and retrieve reports that correspond to levels of the R&D organizational structure. An entire report can be stored as a plex anywhere in an NLS file and can be displayed from that file with the appropriate viewspecs.

Numerical Methods

L10 does not support floating point data types or arithmetic. Thus, all floating point arithmetic must be done through procedure (subroutine) calls. ARC has two sets of mathematics packages available, one for single-precision and one for double-precision floating point.

For the Project System, a single-precision arithmetic package was used. This is simple since L10 allows the programer to drop into assembly language at any time. Each floating point procedure takes arguments and returns a floating point result.

For example,

```
a _ fpadd(b, c);
```

means add floating point numbers b and c and deposit into a. Single precision floating point numbers are contained in one PDP-10 word, so simple variables are used to hold them. L10 does not have data type checking so it all works.

To use double precision floating point, addresses of two-word numbers would be manipulated. The previous example would be,

```
fpadd($da, $db, $dc);
```

which means deposit the double precision result of adding numbers located at db and dc into a two-word array at da.

All that is lost by using L10 is the extra time it takes to call a procedure to do the arithmetic. No loss in capability occurs.

Report Generation

NLS is ideal for handling textual information, no matter what the source of the information. Once a report has been generated by the Project System, it becomes a standard NLS structure, like any other document. Since NLS is very good at manipulating documents, the manipulation of reports in the Project System is simple and efficient. Generation of the reports themselves is an independent matter.

The Project System report file holds three classes of reports: free-text, forms, and project listings. Each of these are handled differently with regard to generation of initial reports, updating of the reports, and display of the reports.

Free-text reports

Free-text reports are standard NLS files that are generated partially by the system from the project data base and partially by the user through editing commands. They are stored as NLS statements, with one special feature--the heading information generated from the project data base cannot be bugged when displayed in the Project System. Free-text reports are handled simply, using standard NLS editing commands.

The header information cannot be bugged because a special property on the appropriate header statements was set. The NLS statement property is a powerful feature that allows attachment of non-textual information to a statement. In other applications, properties are used to store graphics diagrams and formatting information.

Forms reports

Forms reports are very special types of NLS structures. They are initially generated through the interpretation of a report template. The generated structure has no explicit hierarchy, rather it has properties that define the position and format of data on the screen and the location in the data base from which a particular field came. The form is displayed through a special portrayal generator and edited through special Project System procedures. As a field is edited, the data base location property is used to modify the corresponding data base element(s).

Considerable development went into the template interpretation system to provide a flexible mechanism for generating complex report forms from a data base or directly from user input. The result of this development is that users can alter the format of forms reports through simple NLS editing of a template.

The Resource Data Input Sheet, Section Summary, and Department Summary reports were all prepared using form templates. Once the data base design was complete, it took only a couple of hours to prepare templates for the Resource Data Input Sheet, the Section Summary, and Department Summary reports.

Project listings

Project listings are generated from the project data base and stored as standard NLS files. Project listings are generated via "hard coding" methods. Procedures were written to extract and format data base information according to some fixed, specified format. These reports are the hardest to modify with respect to content and format.

Due to some recent improvements in the template interpretation that supports conditionals, these reports could now be produced using the forms approach.

New versions of project listings are generated on demand, either by the user or by a special generation program run automatically in the off hours.

Contention for Reports and Data

Contention for NLS files is handled by a partial copy mechanism that allows many readers and one writer. The modifications made by the writer are invisible to the readers until the file is updated by the writer. This method is used by the Project System to lock a file during editing.

Contention exists at two other levels in the Project System. First, any time a report is displayed for input/update, it must be locked since only one user can update the contents of a report. This is accomplished by setting a locked flag in the report file for that report and copying the report to a user file. When the user no longer wishes to modify the report, a Leave command is given and the old report in the report file is replaced by the new one in the user file and the report file is updated.

The second level of contention is for resource data elements in the data base file. It should be a simple readers and writers problem, but NLS's solution to contention for a file incurs heavy overhead. This overhead is experienced each time the project file is locked. A mechanism similar to that used for locking reports could have been used for locking project records, were it not for the fact that summary elements must be updated also. Since department summary elements must be updated, locking also has to occur on the department summary record. For the pilot, since the summary records are part of the project file and there

is essentially only one department, this is equivalent to locking the entire project file.

Usefulness

Can the Project System, as implemented by SRI, perform in a manner that is satisfactory? Can it be altered easily? Can it be trusted to maintain the integrity of its data? This section will answer these and related questions.

Response Time

A major complaint from the client has been overall NLS responsiveness. The Project System is no exception, since it runs as part of the NLS environment. Fortunately, since moving to the Office-1 computer there has been a vast improvement in system responsiveness.

Some points to keep in mind are 1) the SRI-KL computer has had a poor performance record and has rarely provided responsive service to any of its users except during off hours, 2) the Project System runs under NLS 9 and thus will be less responsive than NLS 8.5 until NLS 9.0 is released to the general utility clients, 3) until the Office-1 computer becomes more loaded, all systems will provide more than adequate responsiveness, and 4) the Project System performs some very time-consuming functions, such as the total regeneration of a report (however, these functions are usually performed automatically at night).

Flexibility

User interface

It is extremely easy to change the user interface for the Project System. The interface is written in CML, which is a high-level representation of the command alternatives. The entire user command language was prepared prior to implementation of the supporting execution code so that the client could become familiar with what the system would "look" like, long before the system was completed.

The interface includes menu selections, which can be changed just by editing an NLS file (menu file). The dynamic nature of the menu alternatives requires that changes be made easily and that they be incorporated rapidly. Throughout the development and review of the system, the data base administrator, NOT a programmer, made changes to the menu file.

Entire grammars can be written in a modular fashion using CML rules, which are similar to macros. The Project System was written in this manner. This greatly expedited the delivery of the Client and R&D paths once the Project Reporting path was completed. The two additional paths were added in approximately one person-week.

Project System Backend

The Project System backend, the execution portion, is written in L10. Included in this document is a detailed section on Code Development that discusses the issue of flexibility and ease of modification.

Report Templates

It is extremely important to be able to compose new reports and variations on existing reports. The design of the template forms capability (described above) allows the format of the report to be designed with NLS text editing. The field designators can usually be placed in an NLS file exactly where they would appear on the screen in the final generated report.

If a new field is added to a report then some knowledge of the data base is required. If the field is generated from a data base element that already exists, then a simple retrieval rule is added to the template. This is a reasonable task for the data base administrator.

If the field is to represent a piece of information that does not exist in the data base, then changes to the data base and data dictionary must occur. It is possible for the data base administrator to add the data base element and dictionary entries, but only if sufficient training has been given. At the time this document was written, the data base administrator did not have this training.

Reliability

Unlike many other text processing systems, NLS reliably saves all edits made to files, even across crashes. Since the Project System uses the core procedures of NLS to make all edits to reports and data base files, that same reliability is achieved.

Even in the case where links are inserted in files for cross-file references, safeguards have been set up to keep the data base links consistent. One such safeguard is the data base verification program that runs each time the computer is restarted. The verification program guarantees that the data base will be in a logically consistent state, without backing up to checkpoints of the data base.

A mechanism similar to the NLS partial copy mechanism is used to lock a report during updating. This mechanism has built in redundancy to allow for system crash/restart or user interruption of a session. The system keeps two generations of all files online. All files are copied to tape every night as a final safety measure.

The Development Environment

Communication

Sending Messages

The message sending features of TENEX/TOPS-20 were used frequently by both SRI and the client during the development cycle. Design issues were often discussed, events announced, and feedback given. It was not necessary to locate a person by phone when an instantaneous reply was not required. Conference calls were often replaced by simple messages with multiple copies. Some of the users catalogued their messages to give a record of their dialogue with others.

Linking

When immediate feedback was required, a more personal form of communication was used--linking terminals. If the respondent is logged in, then an immediate conversation can occur using linking.

Shared files

As collaboration on design and implementation strategies increased, the need to deposit ideas into a common document was felt. By providing access to common files, several people could merge information without ever exchanging pieces of paper or making phone calls. This technique, coupled with sending messages, lead to several joint efforts, such as user documentation and the implementation design. This document was developed as a shared file.

Code development

A large set of software development tools was used for programming the backend of the Project System. The tools include a high-level language, a programmer's editor, interactive compilers, incremental loaders, and a powerful debugger.

The executing portion of the Project System was written in L10, a high-level, procedure-oriented, algorithmic language. It is a generally accepted fact that modifications to software systems are easier when the system is written in a high-level language. The following are a few of the unique L10 features that make programming

large systems even easier: string manipulation, list structures, signals for exception handling, and coroutines.

Once the proper algorithms and programming constructs are selected, the source code can be constructed. At this time a powerful text editor can reduce program composition time considerably. The NLS structure makes programs very easy to read; moreover, it assists the programmer as a thinking tool. NLS encourages well-structured, top-down design of systems. Through the use of level clipping, the programmer can at any moment view and manipulate the organization of large sections of the program. ARC's programmer's editor includes all the NLS Base editing commands, along with commands to insert special L10 constructs, such as IF-THEN-ELSE, CASE, WHILE statements and entire procedure skeletons. This encourages proper use of well-structured and well-documented procedures.

When testing an interactive system, tools for interactive modification of the system should be available. ARC's programming tools include compilers available in an interactive mode and a powerful interactive, incremental loading capability. Using these, a programmer can alter a single procedure in a subsystem by doing a "procedure replacement, " which amounts to compiling the procedure, loading the procedure, and resolving all calls to the old procedure so that the new procedure is called. This technique saves countless hours of development time that would be spent compiling modules, reloading entire systems, and bringing them into the user program buffer.

Debugging

ARC has developed a very sophisticated interactive debugger called DAD. DAD supports high-level symbolic debugging of multiple processes from a single terminal. A few of its features are breakpoints, interrupts, dynamically modifying code and data, and support of many L10 data structures. DAD was used extensively in all phases of development and testing.

One of the more unique features of DAD is the ability to debug an errant process that is being run outside the debugging environment. This feature was demonstrated when a bug was detected by a remote user during testing. The process was interrupted and the

job was detached by the user. An ARC programmer attached to the job, spliced DAD in to monitor the process, and found the bug causing the problem.

The Operating System--Advantages of TENEX/TOPS-20

The TENEX/TOPS-20 operating system is an extremely flexible system for interactive computing. NLS makes use of most of that flexibility. The most visible features are the file system, terminal support, and mail support. TOPS-20 provides for a multi-tiered file system that gives two tiers for use by NLS, the user name and the file name. The support of very long file names makes organization of files in one's directory straightforward. The file name fillout capabilities save considerable type-in time. File access protection is supported in a way that is suitable for almost all applications, even though it is not totally general.

TOPS-20 handles all terminals as if they were general purpose devices that can be linked to any source and/or destination user or process. This allows users to link to one another, so that the output of each terminal goes to both users.

TOPS-20 supports its own messaging system, which is interfaced to NLS via the Message Subsystem. This allows the user to send messages and read messages without ever leaving NLS. As is mentioned elsewhere, messaging was used a great deal in developing the Project System.

Costs

Design of Implementation (initial version)	.35 person-month
Consulting and Presentations	.9 person-month
Programming and System Documentation	6 person-months
(Includes final version of Design Document)	
User Documentation	.55 person-month
This document	.25 person-month

A Project System vs. Plain Old NLS

Why develop a special system to do what could be done with standard NLS files, using links and level and line clipping? What does the Project System do that is important to the client? These questions can be answered by restating three of the design goals of the Project system: 1) to support the novice user, 2) to limit the actions available to the user, and 3) to provide current human and dollar resource information.

Many NLS subsystems have been developed with the more skilled NLS user in mind. They really represent a method of extending the power of the NLS user. However, in this case the client needed a system that could be tested in a pilot mode with users not necessarily skilled at NLS. Thus, the Project System was designed to perform many functions that are available to the general NLS user, but it provides them in a system that is much easier to use.

The Project System was designed for maximum utility, but it exerts many controls that prevent damage to the integrity of the data base, the reports, and the report framework. While in the Project System, a user has enhanced capabilities with respect to the files, i.e., he or she may modify them. To prevent accidental modifications, the system restricts the functions available to the user to only the necessary commands.

NLS provides enormous flexibility in formatting documents. If many Project System users used NLS to compose reports, standardization of report format would be a problem. The Project System completely controls all formatting of reports, except for the body of the free-text reports.

The Project System performs many calculations on human and dollar resource data that otherwise would have to be done by users. This includes both dollar calculations from human resource data and the summarizing of all data. A desirable side effect is that the system provides a simple way to centralize calculation formulas and salary rates.

Problem Areas

During the design and implementation phases of the Project System, several problems arose that are worth examining. They are categorized here under hardware, design, milestones, and feedback problems.

Hardware Problems

Due to the poor performance of the SRI-KL computer and the subsequent switch to the Office-1 computer, there were considerable delays. NLS 9 and thus the Project System were only marginally usable on the SRI-KL. At least half a week of development time was lost bringing up NLS 9 on Office-1.

The initial users of the Project System were to access the system from Seattle, but the communications lines were so bad that the system was completely unusable.

Requirements Specification Problems

The client presented a detailed design document to SRI that was to be the basis for the implementation of the Project System. This was inappropriate. A functional requirements document was needed and from this a detailed design would have been produced.

In trying to provide the detailed design themselves, the client designers were at a distinct disadvantage. They were attempting to produce a detailed design for the Project System without becoming fully familiar with the NLS environment. This resulted in a design that was modelled around a data base management system, rather than a design that was oriented toward existing NLS capabilities.

The client and SRI cooperated more closely on the design of the Client Path and R&D Path (formerly the Management System). While the NLS capabilities were better understood during this phase of design, many design decisions concerning this phase were not made until very late in the contract.

Milestone Problems

Several milestones were not met during the contract period. The first was the delivery of the detailed design, which was approximately two weeks late. The most

notable milestone slippage was the delivery of the template capability for report generation. SRI was approximately one month late with the template capability. This prevented delivery of a system that would support the input of resource data and the display of all the resource data reports, and the file system was not ready for use during the one production cycle that occurred during this pilot study.

Feedback Problems

The NLS environment offers a wide range of tools that encourage communication. Even though sending messages, linking, and shared files were used extensively, feedback on the Project System has not been adequate. Hesitancy by the client to demonstrate anything but the final system is partially responsible for this lack of feedback. Interestingly enough, once the final system was delivered, feedback decreased severely. As a result, the Project System is perhaps too much a product of a system designer/implementer at SRI.

Conclusions

This document discussed the issues that are pivotal in arguing for and against the appropriateness and effectiveness of the Project System as implemented under NLS. In the document, the weaknesses and strengths of NLS were indicated, especially as they relate to the requirements specified by the client.

It is important to examine the history of the Project System implementation, as has been done in this document. In addition, it is interesting to look at the future in terms of how the development methods will affect the expansion and modification of the system. As a result of modular programming techniques and the development of a powerful report generation capability, the Project System is in an excellent position to evolve into a large-scale system to support R&D project management and planning. While efficiency was not the prime consideration in its implementation, flexibility and maintainability were.

It is also desirable that the Project System interface to existing and future systems. To strengthen the project tracking capabilities of management, interface to the client's accounting system is important. To give project management more detailed human resource information, interface to some kind of organization chart system seems appropriate. NLS, in general, and the Project System, in particular, have been built to permit interfaces to external information sources. Implementation of these interfaces will be much easier than with many other systems.

K. Harrenstien
RFC 742, NIC 42758 (Dec. 30, 1977)

NAME/FINGER

Introduction

This note describes the Name/Finger protocol. This is a simple protocol which provides an interface to the Name and Finger programs at several network sites. These programs return a friendly, human-oriented status report on either the system at the moment or a particular person in depth. Currently only the SAIL (SU-AI), SRI (SRI-(KA/KL)), and ITS (MIT-(AI/ML/MC/DMS)) sites support this protocol, but there are other systems with similar programs that could easily be made servers; there is no required format and the protocol consists mostly of specifying a single "command line".

To use via the network:

ICP to socket 117 (oct), 79 (decimal) and establish two 8-bit connections.

Send a single "command line", ending with <CRLF>.

Receive information which will vary depending on the above line and the particular system. The server closes its connections as soon as this output is finished.

The command line:

Systems may differ in their interpretations of this line. However, the basic scheme is straightforward: if the line is null (i.e. just a <CRLF> is sent) then the server should return a "default" report which lists all people using the system at that moment. If on the other hand a user name is specified (e.g. FOO<CRLF>) then the response should concern only that particular user, whether logged in or not.

Both ITS and SAIL sites allow several names to be included on the line, separated by commas; but the syntax for some servers can be slightly more elaborate. For example, if "/W" (called the "Whois switch") also appears on the line given to an ITS server, much fuller descriptions are returned. The complete documentation may be found at any time in the files ".INFO.;NAME ORDER" on MIT-AI, "FINGER.LES[UP,DOC]" on SU-AI, and "<DOCUMENTATION>FINGER.DOC" on

SRI-KL, all freely accessible by FTP (with the exception of SRI-KL, where TOPS-20 requires the "anonymous" login convention).

Allowable "names" in the command line should of course include "user names" or "login names" as defined by the system, but it is also reasonable to understand last names or even full names as well. If a name is ambiguous, all possible derivations should be returned in some fashion; SAIL will simply list the possible names and no more, whereas an ITS server will furnish the full standard information for each possibility.

Response to null command line - "default" listing:

This is a request for a list of all online users, much like a TOPS-10 or TENEX "systat". To fulfill the basic intent of the Name/Finger programs, the returned list should include at least the full names of each user and the physical locations of their terminals insofar as they can be determined. Including the job name and idle time (number of minutes since last typein, or since last job activity) is also reasonable and useful. The appendix has examples which demonstrate how this information can be formatted.

Response to non-null command line - "name" listing:

For in-depth status of a specified user, there are two main cases. If the user is logged in, a line or two is returned in the same format as that for the "default" listing, but showing only that user. If not logged in, things become more interesting. Furnishing the full name and time of last logout is the expected thing to do, but there is also a "plan" feature, wherein a user may leave a short message that will be included in the response to such requests. This is easily implemented by (for example) having the program look for a specially named text file on the user's directory or some common area. See the examples for typical "plans".

Implementation miscellany:

Anyone wishing to implement such a server is encouraged to get in touch with the maintainers of NAME by sending a message to BUG-NAME @ MIT-AI; apart from offering advice and help, a list of all sites with such servers is kept there. It is also suggested that any existing programs performing similar functions locally (i.e. not as net servers) be extended to allow specification of other sites, or names at other sites. For example, on ITS systems one can say ":NAME<cr>" for a local default listing, or ":NAME @SAIL<cr>" for SAIL's default listing, or ":NAME Foo@MC<cr>" to ask MIT-MC about Foo's status, etc.

It should be noted that connecting directly to the server from a TIP or an equally narrow-minded TELNET-protocol user program can result in meaningless attempts at option negotiation being sent to the server, which will foul up the command line interpretation unless the server knows enough to filter out IAC's and perhaps even respond negatively (IAC WON'T) to all option commands received. This is a convenience but is not at all required, since normally the user side is just an extended NAME/FINGER type program.

And finally a little background:

The FINGER program at SAIL, written by Les Earnest, was the inspiration for the NAME program on ITS. Earl Killian at MIT and Brian Harvey at SAIL were jointly responsible for implementing the protocol just described, and Greg Hinchliffe has recently brought up a similar server for SRI-KA and SRI-KL.

EXAMPLES

NOTE: It is possible for some lines of the actual output to exceed 80 chars in length. The handling of such lines is of course dependant on the particular user program; in these examples, lines have been truncated to 72 chars for greater clarity.

Three examples with a null command line:

Site: MIT-AI
Command line:

```
-User-    --Full name--          Jobnam Idle TTY -Console location-
XGP     O Xerox Graphics Printer XGPSPL      T24 Datapoint Near XGP (9TH)
FFM     U Steven J. Kudlak       HACTRN      T41 Net site CMU-10A
KLH     + Ken Harrenstien        F           T42 Net site SRI-KL
___013  -  Not Logged In         HACTRN 1:26.T43 DSSR UNIX x3-6048 (MIT-*
CWH     U Carl W. Hoffman        E         4.T50 919 Very Small Data Bas*
CARL    A Carl Hewitt            HACTRN 5:03.T52 813 Hewitt x5873
APD     M Alexander Doohovskoy   XGP    1:52.T54 912 9th Floor Lounge x6*
JJK     T James Koschella        E           T55 824 Hollerbach, Levin, *
KEN     L Kenneth Kahn           E           T56 925 Moon (Tycho under) *
```

Site: SAIL
Command line:

```
     Person              Job Jobnam Idle     Terminal
DAN  Dan Sleator          46 MACLSP       DM-3          150/1200 modem 415 49*
DEK  Don Knuth             3 E         3. tv-55  205   Library
                          20 PI        2  TV-55  205   Library
ES   Gene Salamin         44 SD MC        TV-40  223a  Farmwald
JJ   Jerrold Ginsparg     11 TELNET       DM-0          150/1200 modem 415 49*
JMC  John McCarthy         1 FINGER     . detached
                          12 E         2. IML-15        McCarthy's house
KRD  Randy Davis          42 AID       7  TV-52  203   Allen
LES  Les Earnest          23 TEMPS     2. DM-1          150/1200 modem 415 49*
ME   Martin Frost         17 E         3  tv-46  220   Filman, Frost
                          31 E            TV-46  220   Filman, Frost
PAM  Paul Martin           9 E            TV-106 251C  King, Levy, Martin
ROD  Rod Brooks           37 MACLSP    3  TV-117 250C
RWG  Bill Gosper          30 SD MC        TV-34  230e  Robinson
                                          TV-67  213   Kant, McCune, Steinbe*
RWW  Richard Weyhrauch    39 E            TV-42  214   Weyhrauch
SYS  system files          6 FINGER       PTY122       job 5 Arpanet site AI*
```

```
Site: SRI-KL
Command line:

Thursday, 15-Dec-77 01:21:24-PST  System up  3 Days, 22:20:52  28 Jobs
Drum   0%  Load avs 0.26 0.23 0.31    14 Act, 10 Idle,  8 Det

User     Personal Name       Job Subsys 15m%  TTY  Room    Console Location
BLEAN    Bob Blean            37 EXEC    0.0   41  K2007   Blean
KLH      Ken Harrenstien      83 TELNET  1.6   12  J2023   Spaceport
KREMERS  Jan Kremers          48 TECO    0.0  121  Dialup  326-7005 (300 Ba*
MAINT    Digital Equipment    54 SNDMSG  0.5   43  K2035   Melling
MCCLURG  Jim McClurg          40 EXEC    0.0   26  PKT
MMCM     Michael McMahon      31 EXEC    1.5  122  Dialup  326-7006 (300 Ba*
MOORE    J Moore              52 TV      0.2  124  Dialup  326-7008 (300 Ba*
PATTIS   Richard Pattis       19 LISP    0.8   11  ARC
PETERSO  Norman Peterson      33 EXEC   25:12 234          (RAND-TIP)
STONE    Duane Stone          34 TELNET 3:51  240          (RADC-TIP)
                              27 EXEC   7:11  232          (SRI-KL)
TORRES   Israel Torres        64 BSYS    0.0   76  K2079   TI by tape drives
                              68 EXEC   1:15  104  K2029   Operators' Office
```

Examples with names specified:

```
Site: MIT-AI
Command line: klh

KLH     + Ken Harrenstien          Last logout 10/16/77 13:02:11  No plan.
```

```
Site: MIT-MC
Command line: cbf

CBF     M Charles Frankston        Not logged in.  Plan:
  I'll be visiting another planet til about December 15.  If anyone
  wants to get a hold of me transmit on some fundamental wavelength
  (like the radius of the hydrogen atom).
```

```
Site: MIT-MC
Command line: smith

BRIAN   A Brian C. Smith           Last logout 11/24/77 08:02:24  No plan.
DBS     T David B. Smith           Last logout 12/03/77 11:24:01  No plan.
BPS     T Byron Paul Smith         Not logged in.  No plan.
GRS     U Gary R. Smith            Last logout 12/12/77 18:43:19  No plan.
JOS     S Julius Orion III Smith   Last logout 11/29/77 06:18:18  No plan.
$PETE   M PETER G. SMITH,          Not logged in.  No plan.
IAN     L Ian C. Smith             Not logged in.  No plan.
AJS     D Arnold J. Smith          Last logout 12/09/77 14:31:11  No plan.
```

```
Site: SU-AI
Command line: smith

"SMITH" is ambiguous:
RS  Bob Smith
DAV Dave Smith
JOS Julius Smith
LCS Leland Smith
```

```
Site: SU-AI
Command line: jbr

     Person              Job Jobnam Idle Line  Room   Location
JBR Jeff Rubin            16 COPY    27. TV-43  222    Rubin
                                        TV-104 233    hand-eye table


Site: SU-AI
Command line: bh

     Person                   Last logout
BH  Brian Harvey         22:49 on 14 Dec 1977.  Plan:
    ↑008-Oct-77  2156   BH  ↑Y12257 (1-Jul-78)
Weekdays during the day I'm usually unreachable; I'm either at S.F.
State or at Benjamin Franklin JHS in San Francisco, but neither place
is recommended for leaving messages.  Evenings and weekends I'm
generally home (55) 751-1762 unless I'm at SAIL.  I log in daily from
home.


Site: SRI-KL
Command line: greg

GREG (Greg Hinchliffe) is on the system:

Job Subsys  # Siz  Runtime     1m%  15m%  TTY  Room   Console Location
62  EXEC    1   0  0:00:10.6        0.8  235         (SUMEX-AIM)

Last login: Mon 12-Dec-77, 15:05, from SUMEX-AIM (Host #56.)
GREG has no new mail, last read on Mon 12-Dec-77 15:10
```

AF Software Eng, Std.

PROPOSED AIR FORCE SOFTWARE ENGINEERING STANDARD

1 General Introduction 1

The historical attributes of the software acquisition process addressed in this presentation is the very high cost of software development and the difficulty of maintaining or modifying the software after it's operational. 1a

The history of government software system acquisitions is liberally spread with financial horror stories. Projects die in midstream when costs from overruns and unforseen developments occur. And even those projects that finally fly, are often burdened with costs that are prohibitive. Historically, the systems acquisition organizations focussed on the hardware portion of the system and the software just happened incidently. It is only recently that we have become aware of the preponderance of the software costs and the immediate need for doing something about them. 1b

We, at RADC, have compiled a Software Development Standard in an attempt to provide a partial solution to the problem. No single document can completely cure the situation, but we feel that the use of this standard will move us a giant step forward in the right direction. 1c

This Standard will attempt to present a comprehensive set of engineering techniques, procedures, conventions, and restraints that we believe are necessary for the generation of high quality, highly reliable software. The standard's contents consists of items that are thought of, collectively, as Modern Programming Practices. 1d

A software document of this kind presents a new way of doing business for the government. Traditionally, the government would specify all requirements of the hardware in the system and leave the software to the discretion of the contractor. One or two sentences describing the functions of the software were considered sufficient to deal with the software aspect of a system. 1e

The imposition of a specification of this nature changes all that radically. We feel confident that we know enough about software development to mandate that a set of tools and procedures be applied to an ongoing effort. We know that

telling the contractor how he should design his programs in general, how he should restrain code and the selection of the language used, how he should keep track of the software development status, etc. will result in more reliable timely software. Moreover, hardware produced under such restraints will prove more amenable to debugging, modification, and eventual expansion without having to start all over again. All of the above will result in reduced costs. 1f

It often happens that smaller than system projects which combine hardware and software are managed by an engineer who might be fully competent in radar technology, for instance, but is totally innocent of the correct way to buy software. Having a Standard from which he can select appropriate procedures and tools for use in his effort will be of great help. 1g

We, as a laboratory with several sections specializing in the state of the art in software engineering are in a position to know how a software system can be properly developed. And putting all the various aspects of the task in one document provides a focus for modern programming technology. 1h

Since this Standard appears now, long after many software houses have built up impressive inventories of tools and procedures in-house or on other government contracts, provisions have to be made for allowing their use wherever possible in satisfying the mandated requirements. This is simply done by stating the requirements as functionally as possible. Any proposed tool that will satisfy the required functions is considered acceptable. 1i

2 Pertinent System Life Cycle Factors 2

The proposed AF Software Engineering Standard (AFSES) will cover that portion of a system life cycle beginning early in the requirements phase when the software portion of the system has been defined and the requirements for the software are being generated. It affects elements in the design, coding, test, and integration continuing on through the deployment phase. 2a

Although this defines the area over which the Standard is actually operative, its effect and resultant benefits extend on through the operation and maintenance phases of the system. Ease of maintaining and ease of modifying the software resulting from techniques mandated by the Standard

continue to pay dividends throughout the life of the system. Attributes inherent in the design and coding of the software system produced yield cost and time advantages in every phase of the life cycle subsequent to their implementation. 2b

Ongoing studies are being conducted at RADC on the applicability and validity of various software engineering methods, procedures, and tools. Data repositories are being developed to serve as historical data from which insights can be gained into fruitful areas of research. 2c

3 Maintaining the Competitive Environment 3

In the overall world of system acquisition, the ever-present element of maintaining a free competitive environment is nowhere more complex than the software portion of the system. How do you mandate engineering requirements, requirements that tell the contractor how to do things, without inhibiting his unique skills and inventiveness? 3a

Furthermore if he cannot propose the superior fruits of his individual skills freely, how can he produce a winning proposal? If certain tools and procedures developed by a rival firm are mandated, why should he be burdened with the task of responding to a requirement that is clearly the invention of some rival corporation? Contractors are very reluctant to produce and use a software tool invented by a rival software house. 3b

The problem is, "How much and what kind of direction should be given the contractor?" 3c

The two opposite poles of this question are:
1. Leave the contractor a free hand to propose his own solution and methods for fulfilling the system requirements.
2. Specify exactly what and how the contractor should produce the system. 3d

Between these two diametrically opposed poles lies a wide field of choice. History has clearly shown that leaving the contractor a free hand and awarding the contracts to the lowest bidder is often the most expensive and time-consuming way to go. On the other hand, specifying everything not only inhibits the contractor's special talents and inventiveness, it also transfers the responsibility for the success of the software to the government. 3e

Many approaches to handling this question have grown up and are in use at present. One typical approach is to suggest the techniques and procedures and let the contractor propose in detail the items you would like to see in the proposal. For example, here is an item from a current procurement document which guides the preparation of proposals; "Describe all software programming languages, practices, standards, methods and conventions." 3f

The procuring agency wants the contractor to propose a higher order language, and he hopes the language suggested will be one of the languages officially approved by his agency. He hopes the contractor will propose top down design and implementation. He is looking for structured code or a very good reason why it shouldn't be offered. This leaves it all up to the contractor to realize what is implied. If all the contractors bidding on the job guess right and they all bid and price the same items, it will be entirely up to the evaluators to pick the best proposal based on the current state of the art and the peculiarities of the subject system. But it's not very likely that they all will guess right and forcing bidders to guess is a sure invitation to confusion at best. 3g

Another problem is how to impose requirements without provoking legitimate objections from the contractors. Why should a contractor explicitly propose all the latest guarantees of timely, low-cost software development when many of them cost more at the beginning? It may push the proposal out of the running if the other contractors merely give lip service to the choice buzz words which will allow them to price the work as usual. This will let overruns and extended debugging as the project develops be paid for in engineering changes and adjustments where the costs don't show up on the competitive bids. 3h

The only solution seems to be to require each item specifically so that the contractor must reapond to it in detail. If the specified items have to be met by all the bidders, the costs remain competitive and the evaluators can examine the proposals for depth of understanding and clever useful items over and above those required. To overcome the problems of requiring devices developed by rival software houses, the requirements must be expressed in terms of their bare functions. 3i

This aspect of procurement which tends to keep the original

development contract cost down regardless of total life cycle costs is a source of resistance from systems procuring offices down to the project manager of a small procurement. Money is allotted to the software portion of the effort and the normal attitude is to get as much software for the available dollar as possible and never mind spending anything extra on the support environment. The fact that money spent up front results in very real life cycle cost savings doesn't fit comfortably in the traditional scheme of doing business. A certain amount of education must take place before such resistance is overcome. 3j

However, more and more responsible administrators in the System Project Offices (SPO) are coming to realize that it's worth it to pay more at the beginning to implement Modern Programming Practices thus upgrading the quality and reliability of the system. They are increasingly willing to specify the items that promise better, less troublesome software systems. 3k

4 Components of the Standard and Their Use 4

The proposed Standard is composed of two principal parts. The first part consists of a set of sections stating the items required. They are carefully worded for as-is insertion in the various documents of a Request For Proposal (RFP) and are intended to go on contract. This part is referred to as the Specification part of the Standard. The second part consists of a Users Guide which contains instructions for using the standard and peripheral information useful to the SPO and the contractor. Thus the Specification contains the sections to be put on contract and the Users Guide contains instructions for tailoring the sections selected and criteria for selecting appropriate sections. Most of the Users Guide content is intended to support the software acquisition agency rather than the contractor. 4a

The Standard is designed as a looseleaf document to allow for constant updating as research projects surface new techniques, devices, or procedures suitable for inclusion in the document. The entire concept is evolutionary to accomodate the most recent products in the software industry that will aid in producing more reliable and maintainable software. It will also serve as an effective device for transferring the latest technology since an items appearance in the Specification will be immediately accessible to any

user of the Standard. When the user references the Standard on his next procurement effort, he will see the new item in it and will decide whether or not it's appropriate for his present effort. 4b

Besides improving the process of developing the software itself, the Standard will provide items that will improve visibility so necessary for effective management and control of the process throughout the development phase. The Program Support Library, for instance, provides management with updated status information on the developing software throughout the effort. 4c

In an effort to avoid the common pitfalls of restrictive government specifications, the Standard contains two statements of overall philosophy. The first is that that "justifiable negotiated waivers or substitutions of functionally equivalent devices or solutions shall be entertained." This is intended to avoid the problem of what to do about proposing a device that a contractor already uses that does the same job as the item stated in the Specification. Reinventing the wheel on every proposal can prove costly and contractors could justifiably complain that the inventor of the device specified had an unfair advantage. 4d

The second statement reads "The overall intent of this Specification is to mandatee requirements as a minimum and shall not preclude exceeding this minimum." This is meant to encourage the contractor to propose functions over and above the stated requirements. 4e

Taken together, the two statements provide the contractor with room for original or individual attributes he may have at his disposal. 4f

The main body of the Specification part of the Standard consists of properly worded sections that can be used in the various documents that form a Request For Proposal (RFP). In addition to the document itself, there are presently eight appendices which contain expanded information relating to certain sections. One example of this is App. 80 which details procedures for handling classified material. 4g

The general headings in the Specification are as follows:

5.1 Programming Languages
5.2 Software Development Tools and Procedures
5.3 Programming Standards and Conventions

5.4 Management Aids and Quality Control Data
5.5 Software Security Procedures
5.6 Documentation Standards
5.7 Operating Systems and Utilities Services
5.8 National Software Works (NSW)
Appendices 10,20,30,...,80 4h

Several sections are divided into sub-sections which allow the user to select the appropriate sub-section for his project depending on size, cost, criticality, or any other considerations. Each succeeding sub-section increases the scope of the requirements and includes the requirements of the preceeding section. 4i

For example 5.4.2 is the section dealing with Program Support Libraries. Under it there are

5.4.2.1 Manual PSL - The simplest form.
5.4.2.2 Basic PSL - which requires some software but the smallest subset of functions.
5.4.2.3 Full PSL With Management Data Collection and Reporting and all functions pertinent to managing a software development effort. 4j

Each succeeding sub-section requires more functions, is more costly, and provides a more comprehensive tool. The user chooses the proper sub-section for his project and disregards the others in the same section. This makes use of the document without tailoring impossible because all sub-sections, taken together, would pose apparent contradictions, or at least, redundancies. 4k

The Standard is continually being updated as research produces new or better items for use in developing more reliable software. As new items are developed, thoroughly tested, and approved, they will become candidates for insertion into the Specification. Therefore, the document is considered open-ended in the sense that it is in a perpetual state of evolution. It would be fatal to ever cast it in stone because that would freeze modern software engineering technology at some particular time frame. 4l

The accompanying document, the Users Guide, contains instructions with examples on how to use the Standard. It also contains information on each section giving its rationale and in which document of the RFP the section should

be placed. Any comments or information pertinent to some aspect of the Specification document that would not be approriate on contract, goes into the Users Guide. 4m

5 Management and Technical Performance 5

One of the major benefits derived from the use of this Standard is the greatly improved management visibility throughout the development phase of the project. 5a

Management visibility is a growing concern in the field of software procurement. One indication of this is the recent development of the Computer Program Development Plan (CPDP) in the Air Force. This is a document that is included in the Request For Proposal (RFP) package. The CPDP queries the bidder on every aspect of software management. The bidder is forced to respond in detail stating how he will manage the software development. Many of the responses to the requirements of the subject Standard will be included in the CPDP. 5b

Both the CPDP and the subject Standard are intended to augment rather than supplant the traditional management procedures of system software procurements. All the standards and Air Force regulations governing acquisition and support procedures such as AF Regulation 800-14, Acquisition and Support Procedures for Computer Resources in Systems, and MIL STD 483 Configuration Management Practices for Systems, Equipment, Munitions, and Computer Programs and its backup MIL STD 490, Specification Practices, are still in force. 5c

All the configuration management and quality assurance procedures with their reviews and audits throughout the life cycle of the system are still intact. Serious thought has been given to how the new requirements would blend and mesh with the traditional activities of a procuring agency. We have been able to demonstrate that no serious disruptions result from the imposition of our new Standard. 5d

Furthermore, the Standard's requirements are not intended to negate any useful tools and procedures the software houses formerly used. For example, the Program Support Library is meant to augment any procedures a software house traditionally used. Many times, the traditional tools fulfill all or most of the required functions. In such cases, all that would be required would be the addition of any missing functions to the existing tools. 5e

In response to concerns expressed by small software houses, RADC conducted a study on the impact of modern program engineering technology on the small contractor. The results of this study showed that even the smallest software houses using a reputable time-sharing service could easily adapt to the new requirements without lessening the constraints or the severity of the requirements in any way. 5f

6 Expected Benefits/Implications for the Software Industry 6

Although it may appear, at first glance, that industry would display great resistance to the imposition of a Standard on them, the fact is that software houses are not really averse to the idea. Many of the responsible houses know that the tools and procedures required in the document are useful in the orderly development of a software system and this Standard gives them a chance to propose them and cost them in the proposal. Another definite benefit is that the more explicitly you can tell the contractor what you want done and how he can best do it, the more secure he feels. Many projects that get in severe difficulties are the result of the contractor guessing wrongly what the government agency really wanted. 6a

As further research produces new tools and procedures, they will be included in the standard for use on procurements. In spite of its necessarily incomplete form, the Standard has been in use for over a year at RADC. Coordination with the office that produced the Specification is mandatory at RADC whenever a procurement contains software. It has been used unofficially on several large-scale weapons system acquisitions. The Electronic Systems Division (ESD) at Hanscom Air Force Base near Boston is currently preparing a version of this Standard that will be appropriate for use by their System Program Offices (SPOs). Our intentions are that the standard will eventually be coordinated with the other defense community agencies and will be proposed as a MIL-STD for use throughout DoD. 6b

(J44751) 25-May-78 12:00;;;; .HJOURNAL="ESV 25-May-78 12:00 44751";
Title: .H1="ELITE FY-78 Budget"; Author(s): E. S. (Ed) Von Gehren/ESV;
Distribution: /JCB3([ACTION]) ; Sub-Collections: NIC; Clerk: ESV;
.IGD=0; .SNF=HJRM; .RM=HJRM-7; .PN=-1; .YBS=1; .PES;

Revised FY-78 Budget (21 April '78)

	Old Budget Amt	Rqmt	Funded*
Obligated**			
Terminal Equip	22K	38K	38K
(to-be-entered)			
ALMSA TDY/per diem	18K	18K	18K
(to-be-entered)			
Computer Resources(9CRUs)	97K	100K	100K
(to-be-entered)			
Equip Maint	6K	6K	6K
(to-be-entered)			
Commo (Dev Contr #1)	60K	99K	99K
(to-be-entered)			
SRI Spt (Dev Contr #2) 120K	(80K deferred until FY79)		
Misc Contr (Dev Contr #3)	25K	25K	12K***
(to-be-entered)			
Line Conditioning	2K		
NLS Training	6K	(6K)	
TYMESHARE-ARC (Reach-Thru)	2K	(2K)	

Development Contract #1 - Design Reach-Thru capability (TYMESHARE-ARC)
Development Contract #2 - Implement Reach-Thru Capability (80K in FY79)(TYMESHARE-ARC)
Development Contract #3 - NMA Contract and other misc contracts to be awarded.
* - Funded items - Monies budgeted and available.
** - Obligated - Monies spent or firmly obligated.
*** - NMA contract support will be covered (and funded) under an LSSA contract.

```
00000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000011111111112222222222333333333344444444445555555555666666666677777777778
12345678901234567890123456789012345678901234567890123456789012345678901234567890
 *START* User FEINLER  Job  NLS Seq. 4090 Date 21-Nov-78 19:33:54 Monitor SRI-KL


FFFFFFFFFF    EEEEEEEEEE      IIIIII     NN      NN    LL            EEEEEEEEEE
FFFFFFFFFF    EEEEEEEEEE      IIIIII     NN      NN    LL            EEEEEEEEEE
FF            EE                II       NN      NN    LL            EE
FF            EE                II       NN      NN    LL            EE
FF            EE                II       NNNN    NN    LL            EE
FF            EE                II       NNNN    NN    LL            EE
FFFFFFFF      EEEEEEEE          II       NN  NN  NN    LL            EEEEEEEE
FFFFFFFF      EEEEEEEE          II       NN  NN  NN    LL            EEEEEEEE
FF            EE                II       NN    NNNN    LL            EE
FF            EE                II       NN    NNNN    LL            EE
FF            EE                II       NN      NN    LL            EE
FF            EE                II       NN      NN    LL            EE
FF            EEEEEEEEEE      IIIIII     NN      NN    LLLLLLLLLL    EEEEEEEEEE
FF            EEEEEEEEEE      IIIIII     NN      NN    LLLLLLLLLL    EEEEEEEEEE
```


START User FEINLER Job NLS Seq. 4090 Date 21-Nov-78 19:33:54 Monitor SRI-KL

START User FEINLER Job NLS Seq. 4090 Date 21-Nov-78 19:33:54 Monitor SRI-KL

START User FEINLER Job NLS Seq. 4090 Date 21-Nov-78 19:33:54 Monitor SRI-KL

START User FEINLER Job NLS Seq. 4090 Date 21-Nov-78 19:33:54 Monitor SRI-KL

START User FEINLER Job NLS Seq. 4090 Date 21-Nov-78 19:33:54 Monitor SRI-KL

START User FEINLER Job NLS Seq. 4090 Date 21-Nov-78 19:33:54 Monitor SRI-KL

20 Nov 1978 1353-PST PETERSON at USC-ISIE: Input and Output of Sequential Text
Distribution: VANNOUHUYS AT SRI, peterson
Received at: 20-Nov-78 13:52:32-PST

Dirk,

We have a n AGILE terminal, which we bought from Martin Hardy for use as a COPYPORT device. It has a micro, and most of its tunctions are programmable, from plot to type. We could program simple things such as characters per inch, lines per inch, sub and supercript, etc., if NLS would pass the special characters into a sequential file. As it stands, we use XED on the Sequencial File.

Realizing that it might be a good idea to interest someone with money in our problem, I sumarized all of the problems which I can remember into a short paragraph in my October status Report to Dick Robinson at RADC. Here is that paragraph:

In the course of investigating the usefulness of the NLS system as an adjunct to NSW application activities, TRW has found that some NLS peculiarities cause problems in transferring files from NLS to other tools, and from other tools to NLS. In particular, the Copy Sequential command will not preserve program structure if blank statements are imbedded in the code. Also, the Output Terminal File command cannot create a file which is directly suitable for input to a compiler.

Following is a list of these and other additional capabilities which TRW recommends for inclusion in NLS.

Output Processor

Enable a "pure" program source file to be created by Output Terminal File. Current version inserts at least one file identifier line which confuses compilers.

Enable Output Processor to create blank page (or skip page number) on next page. Cancel Footer (return to simple page number) on next page would also be of use. These will enable insertion of figures, tables and footnotes into NLS prepared reports, without iteration to determine "next page".

Enable Output Processor to act on .Grabranch=n; which would prevent the printing of a "widow" heading line at the bottom ot a page. Current .Grab; acts only within statement; new one would act within a branch.

Input Sequential

Enable Copy Sequential to retain blank statements but not allow them to affect level of following statements. Present practice reverts to top level at each blank statement.

Other

Fix LPOUTPUT to handle all ASCII characters. TAB and ESC are particularly necessary to our usage, but others would be

required to support effective control of printers for such special activities as sub and super script, characters per inch, lines per inch, etc..

If I got any of them wrong, please let me know!

Regards,

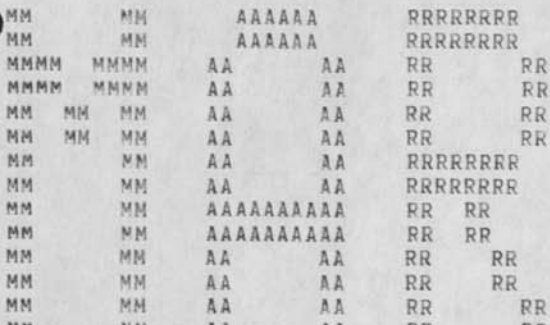
Norm

PS: Thanks for the grammar lesson.

I'll ftp the files to ISIE and try to archive them tonight.

I hope that my attempt to interest RADC is good thing from your point of view.

START User MARIMONT Job CHTREE Seq. 4091 Date 21-Nov-78 20:06:39 Monitor SR



START User MARIMONT Job CHTREE Seq. 4091 Date 21-Nov-78 20:06:39 Monitor SR

START User MARIMONT Job CHTREE Seq. 4091 Date 21-Nov-78 20:06:39 Monitor SR

START User MARIMONT Job CHTREE Seq. 4091 Date 21-Nov-78 20:06:39 Monitor SR

START User MARIMONT Job CHTREE Seq. 4091 Date 21-Nov-78 20:06:39 Monitor SR

START User MARIMONT Job CHTREE Seq. 4091 Date 21-Nov-78 20:06:39 Monitor SR

Draft of New Identsystem Design

Action recipients, please comment on the general features of the new ident system described here. thanks.

Introduction 1

This document contains design notes for a new ident system, to go along with the new MAIL subsystem. This document is not complete at this time but contains enough description of the major features we have in mind to start a discussion. Please comment. 1a

Requirements 2

Multiple Ident Systems 2a

It will be possible to have many ident systems. Each will have a name. To completely specify who an ident represents, it will be necessary to also specify the ident system name if it is different from your own. 2a1

Ident uniqueness. 2a2

Idents will be unique in a given ident system, but not necessarily over all ident systems. 2a2a

Ident duplication. 2a3

Idents may be defined in more than one ident system. These may represent same or different individuals. Or the same individual may be represented in more than one ident system by different idents. 2a3a

Multiple Hosts 2b

Across hosts. 2b1

A single ident system can span several hosts. A single host can support multiple ident systems. 2b1a

Changes. 2b1b

When a change is made in an ident system on one host, the change will autmatically be made in same ident system on other hosts that support that ident system, wherever at all possible. 2b1b1

Support. 2b1c

Ident system X is supported on hosts A and B. That means on hosts A and B there are lookup facilities and database info files. On host C there are no lookup facilities for X and no database. Interrogations may be made from C to

either host A or B (or a designated one of the host set) and results delivered back to C in some fashion. 2b1c1

Mail addressing. 2b1d

It will be possible for a user on host C to deliver a mail item to a user in ident system X without getting query results from host (A or B), but by only specifying the user's ident and ident system name. 2b1d1

Protections 2c

Between Ident Systems 2c1

Each ident system will have its own protection settings on its ident databases. This means that ident system A could be made private and individuals from other ident systems could not find out any ident information from A. This would of course mean that no MAIL could be sent from X to A. In this case a public "in" mailbox ident could be created, and mail could be sent to that ident from outside. 2c1a

Journal Access 2c2

Each journal will have the ability to restrict reading and or writing within the journal. This will be done by the journal and will use ident systems as one way of restricting access. 2c2a

Active and Inactive Idents 2d

There will be an inactive database for each ident system which will contain all expired idents. This will be used to determine who was associated with any given ident at any given date and time in the past. This gives us the capability to reuse idents that are not active. No two idents within the same ident system will be active at any given time. 2d1

Idents as AUGMENT Signatures 2e

Limit of 4 letter/digits. 2e1

Ident Resolution 2e2

Over Ident Systems 2e2a

There will be a way to find out which ident system a given signature in a given file is defined in. Such info

need not be printed with each ident, but will be available via some facility. 2e2a1

Over Individuals 2e2b

There will be a way to find the individual from the statement signature. The date and time in the signature will be resolved to the individual who had that ident on the given date. 2e2b1

Compatability. 2e3

The new ident system will not make old files or their idents obsolete in any way. 2e3a

Larger Ident Systems 2f

One ident system will accept as many idents as reasonably possible. 2f1

Knowledge of Existance 2g

In general, hosts will be required to know of the existance of ident systems that are not supported on that host, and also know the/a "home" host for that ident system. 2g1

Ident system directory. 2g1a

There will be a database that is available to both programs and human users, that contains the names of known ident systems and associated home host names and/or addresses necessary for forwarding mail and making enquiries. 2g1a1

New Administative Subsystem 2h

Ident System Creation 2h1

It will be possible to create a new ident system (that is supported over a given set of hosts). It wil be possible to create an "empty" ident system, or create one with initial entries supplied from an AUGMENT file. 2h1a

Database modification. 2h2

It will be possible for a person or persons to add new idents, delete old idents, and change user information associated with an ident. This will be done by a special "database administrator" individual. 2h2a

Easy to use. 2h2b

The maintenance system will be easy to use and will do automatic verification, etc. Use will be restricted to administrative personell only, but not necessarily system personell. 2h2b1

Insert. 2h2c

It will be possible to formulate and insert a new ident. It will be required that the new ident be completely specified before any entry is made. Any corrections to existing idents (that are necessary due to an addition) will be made automatically. The formulated ident information will be maintained while updating. If errors exist the user will have the ability to edit the entry and try updating again. 2h2c1

Groups. 2h2c1a

If the new ident is a group, all associated idents will be shown to be in the group. 2h2c1a1

Organizations. 2h2c1b

If the new ident is an organization, all associated idents will be shown to be in the organization. 2h2c1b1

Individuals. 2h2c1c

If the new individual is in a group and/or organization, those entities will be changed to show that they include the individual. 2h2c1c1

Office/Role idents. 2h2c1d

Role idents will be treated just like individuals, but will refer to a "real" individual that fills the given role. (FEED is a role ident). 2h2c1d1

Program idents. 2h2c1d2

We will include the concept of a "program" ident-- for programs that can send and receive mail. The purpose and exact use of this is not entirely clear at this time. 2h2c1d2a

Delete. 2h2d

It will be possible to delete iderts from the system. Automatic correction of other ident information will be made. It is acceptable to run a utility after many deletes are made, in order to maintain efficiency of information. 2h2d1

Change. 2h2e

It will be possible to change ident information. Changes will be edited using AUGMENT. Any updates to existing idents will be made automatically. 2h2e1

Login Creation 2h3

There will be an administrative subsystem which will accept the necessary information about a user and will reach thru to the TENEX subsystems that create a login directory and everything else that has to be done to add a new user to the system. This may be postponed until the major part of the ident system is complete. 2h3a

Methods of IDENT lookup 2i

The ident system machinery will allow for the lookup of an ident "record" of information in the following ways: 2i1

By ident. 2i1a

The ident and identsystem name may be supplied. That will uniquely determine an ident record. This must be fast. It will be used by the MAIL system. 2i1a1

By login-name. 2i1b

Given a login-name and host name, the ident record can be located. It is requried that this type of lookup be very fast when performed on the host where the login-name is valid -- i.e. on the same host as "host name". This will be used by AUGMENT an entry time. 2i1b1

By Soundex. 2i1c

This method allows the specification of a last name that "sounds like" the desired name. The result is a list of names of individuals (or organizations possibly) that meet the Soundex test for similarity. This feature will replace the lastname... lookup of the partial lastnames. 2i1c1

By Lastname. 2i1d

The complete last name can be supplied and a list of records for individuals with that last name will be given. 2i1d1

By "other" tests. 2i1e

At some future time we may include other lookup techniques. These will not necessarily be as fast at the ones given above. 2i1e1

In addition it there will be a way to find out which is the "primary" host for a given ident system, and which hosts support that ident system (i.e. which hosts can do a search in that ident system). 2i2

Program Interface 2j

The methods of lookup listed above will be available to programs that run under AUGMENT. These may have to see that certain support code modules are loaded, etc. This type of interface to the facility will be used to implement the following two "interfaces". 2j1

User Interface 2k

The various lookup mechanisms above will be available to users as AUGMENT commands, probably via the MAIL subsystem. 2k1

Ident Server 2l

A possibly limited set of lookup mechanisms will be available as an "ident server". This will be similar to the "who" facilityon SRI-KL and may be used by remote users or remote programs. 2l1

One Ident Per Login 2m

There will be the restriction impossed that only one ident will be associated with each login name. 2m1

Conceptual Framework 3

Conceptually there are the following "objects" in the new ident system: 3a

A set of AUGMENT service hosts, connected via networks to each other and possibly non-AUGMENT hosts. (It will be possible to have AUGMENT hosts that are not connected to networks however). 3a1

For each AUGMENT host: 3a1a

A directory of ident systems known to that host. 3a1a1

For each such ident system, the "master" host is listed, followed by all the hosts that support that ident system. 3a1a1a

A set of ident systems. 3a1a2

For each ident system, a directory with the following files: 3a1a2a

Database files (AUGMENT files containing ident records) 3a1a2a1

NAME-1.AUG 3a1a2a1a

NAME-2.AUG 3a1a2a1b

... 3a1a2a1c

NAME-n.AUG 3a1a2a1d

The inverted -by-ident- lookup file. 3a1a2a2

IDENTS.INFO 3a1a2a2a

Given an ident, this will get you (the login name if on this host, else the host at which individual receives mail) and a link into the NAME-n files. 3a1a2a2a1

The inverted -by-login- lookup file. 3a1a2a3

LOGIN.INFO 3a1a2a3a

Given a login-name, this will get you the ident. 3a1a2a3a1

The inverted -by-Soundex- lookup file. 3a1a2a4

SOUNDEX.INFO 3a1a2a4a

Given a lastname attempt, this will give you a list of links into the name files. 3a1a2a4a1

Database Structure 3b

Active Idents 3b1

Structure 3b1a

The active database for each ident system will contain much of the same information but the structure will be different. The ident database file will appear to have just the name associated with the ident and no information. The information will be in properties. It will be readable through some kind of filter mechanism. 3b1a1

New Categories 3b1b

The following additional categories will be added to each ident: Creation Date and Expiration Date. The creation and expiration date will be used to decide who an ident belongs to durring any given time. This will be needed since idents can be reused. 3b1b1

New Ident Types 3b1c

There will be two new types of idents, role and program idents. Role idents will include such things as FEEDBACK and OPERATOR. 3b1c1

Program Idents? 3b1c2

Inactive Idents 3b2

The inactive idents database will look exactly the same as the active database except that the expiration date will be filled in. 3b2a

Ident Systems 3b3

There will be an inverted table that contains each ident systems home host and the hosts the system is supported on. There will be no readable file containing this information. However this information will be attainable in the user subsystem. 3b3a

"Information" Files, Ident Files 3c

We will implement a very fast general symbol lookup facility, called here "information files". This will be used for lookup by ident and lookup by login name. This will also be used for Soundex lookup. It will be used for other less critical things as well, such as the names of journals residing on a host. 3c1

An information file will contain a hash table, indexed by the hash of the given string. The hash table entry will point to a "data" portion of the file, usually a string for each entry. Given a string and the name of the file, a procedure can return the data string by hitting only 2-4 pages of the file. 3c2

For lookup by ident, the idents will be hashed and the data strings will be pointers to the AUGMENT branch with the ident info. These pointers should be quite compressed, like a number and an sid (.e.g 1-0345 for <OAD-IDENTS>OAD-1.AUG, 0345). There should also be enough information to allow the AUGMENT mailer to deliver mail, given the ident, so it doesn't have to get into the ident file itself. For lookup by login name, the login name will be hashed and the data string will be the ident and possibly the AUGMENT branch. 3c3

Ident Lookup 3d

The BE ident support procedures have to be changed to use the information file approach to find an ident record. Note that this will allow an ident system have its data in more than one AUGMENT file. 3d1

Login Name Lookup 3e

The getuid procedure in the FE has to be changed to use the new lookup by login name process. Does the BE look up the ident by login name as well?? It should get the ident from the FE. 3e1

Ident Database 3f

Last Name and Properties 3f1

The database will have the ident name (e.g. (Andrews)) and associated with it properties. These properties will contain all the information about the named individual, group or organization. There will be a structure associated with these properties so information access will be as direct as possible. 3f1a

System Database 3g

Hashed Table w/Ident System, Master Host and Hosts Supported On 3g1

Addressing Database Information 3h

All database addressing will be done through hash tables (inverted files). The result of the probe into the hast table

will be a pointer into the AUGMENT file. Property traversing will return the specific information desired. 3h1

Database Template Definition 3i

The ident record will be kept in properties in a fixed format. The format will be determined by a template. 3i1

Ident Database Management - Administrative Subsystem 3j

Adding Idents 3j1

A template will be displayed on the screen and filled in by the user. This will really be a separate file that gets integrated into the data base at completion of fillin. Integration will be done by the system and will include entering this template into the proper place in the file as will as altering all other existing entries as required. For example, when an individual is entered, the organization to which this individual belongs must add this individual as a member. In addition all the work that is done now by a JCP hack to create a new login directory and ident will be done by a new facility. This will include entering info into the login table and associating the ident etc. 3j1a

Deleting Idents 3j2

This will simply mean getting rid of this entry from the data base and deleting the ident from all other places it exists. It will also mean placing this entry in the inactive database and entering the expiration date. All the above will be done by the system. 3j2a

Modifying Idents 3j3

To modify an entry, the entry will appear on the screen and the user will be free to edit the entry as desired. This will really be a file with a copy of the actual entry. Upon completion of editing the system will modify the "real" data base as necessary including all other affected entries as in the ADD capability. 3j3a

General Information 3j4

No information will be lost while integration is taking place. If complications arise error messages will be produced and the user will be able to continue editing the last template to correct the errors. 3j4a

The following information will be needed to execute any of the Database Management funcitons:
Ident System + Ident + Ident Kind + Function. 3j4b

Ident Database Creation 3k

Ident System Database Management 3l

Adding Systems or Hosts 3l1

Deleting Systems or Hosts 3l2

Modifying Systems or Hosts 3l3

Ident System Database Creation 3m

User Information Retrieval Subsystem 3n

Finding Idents 3n1

Given the following information a single ident will be returned if an exact match is found otherwise a list of possible names will be returned:
Ident System + Ident Name 3n1a

Finding Associated Idents 3n2

Given the following information an ident list with all the associated idents will be returned:
Ident System + Ident
In the case on an Individual Ident all the organization and group idents associated with this ident will be returned. In the case of a group ident the coordiator, members and subcollection idents will be returned. In the case of and organization ident the coordinator, members groups and subcollection idents will be returned. 3n2a

Finding Information 3n3

Given the following information the address of the information desired will be returned:
Ident System + Ident + Category 3n3a

Finding Which Ident System 3n4

Given a name or ident and and ident system to try finding the ident in, TRUE or FALSE will be returned indicating success or failure. 3n4a

Algorithms 3o

Hashing 3o1

The hashing algorithm will be the hash entity MOD prime number table size. Collisions will be resolved by adding another prime number to the previous result until the entry is found, the whole table is searched or nothing exists in the table. 3o1a

Inverted Tables 3p

Ident Systems Table 3p1

Fields: system, master host, supported host 3p1a

Hashed On: ident system name 3p1b

Lastname Table 3p2

Fields: individual, group or organization name, address in ident database 3p2a

Hashed On: last name 3p2b

Letter Index Table 3p3

Fields: address of first individual with last name beginning with this letter 3p3a

Soundex Name Table 3p4

Fields: soundex key, address of related names in link names table 3p4a

Hashed On: soundex key (computed from a name) 3p4b

Soundex Linked Names Table 3p5

Fields: address of record in ident database, link to next name with same soundex 3p5a

Login Names Table 3p6

Fields: ident, link to ident record 3p6a

Hashed On: login name 3p6b

Ident Table 3p7

OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFF
OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFF
OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFFICE-2 OFF

LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN
LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN
LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN LEHTMAN

HGL75166 HGL75166 HGL75166 HGL75166 HGL75166 HGL75166 $H
HGL75166 HGL75166 HGL75166 HGL75166 HGL75166 HGL75166 $H
HGL75166 HGL75166 HGL75166 HGL75166 HGL75166 HGL75166 $H

Monday, August 18, 1980 10:16:56-PDT Morday, August 18, 1980 10:16:56-PDT Mo
Monday, August 18, 1980 10:16:56-PDT Monday, August 18, 1980 10:16:56-PDT Mo
Monday, August 18, 1980 10:16:56-PDT Monday, August 18, 1980 10:16:56-PDT Mo