RATS Programmer's Handbook

Introduction

RATS (RISOS ARPA Terminal System) is the operating system used on the RISOS PDP11/45. This document assumes familiarity with the PDP11/45 assembly language (*).

Part one of this document describes the environment that the RATS supervisor provides to user programs. It includes a general overview of the system, and detailed descriptions of all of the supervisor calls.

Part two describes the system from the point of view of a user logging in on a terminal.

(*) See, for example:

PDP11/45 Processor Handbook

PAL-11 Assembler Programmer's Manual

Page 2

Part One

The fundamental object in the RATS system is a <process>. The notion of a process should be familiar to users of multiprogrammed computer systems. In RATS, a process consists of:

Eight general registers, including a stack pointer (R6) and a program counter (R7). (The alternate set of hardware registers R0 through R5 which exists in the PDP11/45 is not available to the user programmer.)

A process status word (PS), containing the four condition codes and the "T" bit. (Other parts of the hardware PS are not available to the user programmer.)

An address space, divided into I-space and D-space.

(See the PDP11/45 processor handbook for a discussion of I- and D-spaces.) Each of I- and D-spaces is divided into 8 segments of 8192 bytes (20000 octal) each. Each of these segments has its own set of properties, described below.

A <C-list>, which is a directory containing all of the capabilities of the process. A <capability> is a pointer to an <object>. Examples of objects are files and processes. Capabilities provide the only access to objects.

A number of state variables, discussed in following sections.

User processes all run in user mode. The following paragraphs describe the user process's environment in detail. Certain instructions (such as HALT) are said to be <illegal>. Illegal instructions cause the offending process to stop running, and another process (which is "responsible" for the offending process) is notified of the error. This mechanism is described in more detail in the section on processes.

HALT is an illegal instruction.

WAIT should not be used, since it wastes processor time which could be used by other processes.

RTI may be used wherever convenient. It will not affect the processor mode, register set, or processor priority of the PDP11/45. RTT should not be used, since it may upset the operation of debugging

programs.

SPL and RESET do nothing in user mode.

MTPI, MTPD, MFPI, and MFPD should not be used.

BPT, or opcode 3, should not be used. It is reserved for use by debugging programs.

IOT is presently illegal. Suggestions for its use will be entertained.

EMT is used for supervisor calls and is described more

fully below.

TRAP performs a trap to a user routine. The PS and PC are pushed onto the user's stack and a new PC and condition codes and T-bit are loaded from location 34 of the user's I-space. The process remains in user mode.

All other instructions behave pretty much as advertised in the PDP11/45 processor handbook.

Capabilities

First, some general remarks about capabilities. A <capability> identifies ("points to") some <object> in the system. The types of objects are:

file
semaphore
process
directory
master entry
slave entry
entered process
supervisor

Capabilities exist only in <directories>. A capability is identified by specifying a directory and an <index> within the directory. An index can be any 16-bit number; hence a directory can contain at most 2**16 distinct capabilities.

Every process has a directory associated with it, called its <C-list> (short for capability list). A process can manipulate objects only through capabilities in its C-list. The C-list therefore defines the access privileges of the process.

The EMT instruction (pronounced "emit") is used to perform operations related to objects. When a process executes an EMT, the supervisor examines the word on the top of the process's stack. That word is used as an index specifying a capability in the process's C-list. This capability specifies the object on which some operation is to be performed; it is the capability <invoked> by the EMT. The particular operation to be performed depends on the type of object and the contents of the low byte of the EMT. EMT codes

in this handbook are in octal notation. These operations are described separately for each type of object.

If the top of the stack does not exist or there is no capability at the specified index, the EMT is illegal. (See above for treatment of illegal intructions.)

Parameters relating to the operation may be passed and/or returned on the stack. The word following the EMT, called supplies information relating to these parameters. In this handbook, parameters control words are given as two octal bytes. When any EMT is executed, the stack must contain (from the top): (1) the word specifying the index of the capability being invoked; (2) if bit 7 of the parameter control word is on, a word specifying an index in the C-list; (3) other parameters, equal in number to the contents of bits 6-0 of the parameter control word.

These parameters are referred to by their offset (in octal) from the original SP; e.g. the C-list index parameter would be word 2(SP). There is presently an upper limit of 24 parameters, excluding the C-list index parameter.

After the operation is performed, returned parameters, equal in number to the contents of bits 14-8 of the parameter

control word, are pushed onto the stack. These parameters are referred to by their offset from the final SP. There is presently a limit of 24 parameters.

When the EMT completes, the process resumes execution at the instruction following the parameter control word. Unless otherwise specified, EMT's complete immediately.

The EMT instruction clears all condition codes unless specified otherwise in the descriptions of individual operations.

Every capability has an eight-bit byte associated with it, called the <attribute> field. Each bit in this field defines a permission for certain operations on the object. If

the attribute is present (the bit on), the operation is allowed. Details may be found in the description of each particular operation.

Since directories may contain capabilities to other

directories, it is apparent that the directory structure is analogous to a directed graph. There can be several capabilities referring to a single object; in that case, access to the object is shared among all of the owners of the capabilities. Furthermore, no single owner can cause the object to be deleted, for then the others would have capabilities for a nonexistent object. It is a general rule that an object will continue to exist as long as it is possible to reference it, or in other words, as long as there are any capabilities referring to it. A process may release its own capability to an object, but the object itself is not deleted until the last capability to it is released. This fact is important to note, because often it is desired to delete an object which is consuming resources (such as disk space or an I/O device), and a capability tucked away in an obscure place can be a hindrance.

Deletion of a directory causes deletion of all the capabilities it contains, which may in turn cause deletion of other objects. A process is treated as having one capability, to its C-list. Deletion of a process therefore causes deletion of its C-list (which is a directory) only if there are no other capabilities to that directory.

The directory structure may have directed cycles; for

example, a directory may contain a capability to itself. In such a case, it is possible that capabilities to an object exist even though there is no path from the root of the directory structure to the object. The rule is: an object is deleted if and only if there is no path of capabilities from the root of the directory structure to the object.

Page 6

File

Files are the only form of on-line storage. (Off-line storage, such as magnetic tape, paper tape, and punched cards, is discussed elsewhere.) Files reside on disk and/or in core and are the only access to disk and core. (In this context, "core" means all memory which is directly addressable on the PDP11's Unibus, whether ferrite core or solid-state.)

In many computer systems, files reside on disk or other secondary memory, and are explicitly copied into the user's core or primary memory when needed. In RATS, files are the user's primary memory. In this respect they resemble closely the concept of segments in the Multics system. (The

convention has been established that a PDP11 segment is an 8192-byte area of the address space, and we follow that convention here.) A process references a file by referencing a location in its own address space. If the portion which was referenced is on disk, the supervisor will move it into core, moving part of some other file from core to disk if necessary to make room in core. This activity, known as paging, is completely invisible to the user, except as it affects speed of execution. The entire address space thus appears to be in core at all times.

A file may be from 0 through 2**32 - 1 bytes long. (In practice, available storage places a more severe upper limit on the length of a file.) The address of a byte within a file is therfore two 16-bit words. Since the PDP11 processor generates addresses which are only 16 bits long, a mechanism is needed to map these addresses into file addresses. This mechanism will now be described.

The PDP11/45 processor's address space of 2**16 bytes is divided into eight segments of 2**13 bytes each.

Segment number	contains addresses (in octal) from	through

0	0	17777	289
1	20000	37777	290
2	40000	57777	291
3	60000	77777	292
4	100000	117777	293
5	120000	137777	294
6	140000	157777	295
7	160000	177777	296

297

Page 7 297a

298

299

300

Furthermore, addresses are distinguished as to whether 301
they refer to I-space or D-space. A memory reference is a 302
reference to I-space if it is a fetch of an instruction, index 303
word, immediate operand (such as N in TST #N), or absolute 304
address (such as A in TST @#A); otherwise it is a reference to 305
D-space. Since any address may be in either I-space or 306
D-space according to the context in which it is used, there 307
are eight I-space segments and eight D-space segments. In 308
this handbook, the following convention for numbering segments 309
is used; segments 0-7 are in I-space, and segments 10-17 310
(octal) are in D-space. 311

312

Each of the 16 segments independently refers to some portion of some file. (Or, a segment may refer to no file.) We say that a segment is <attached> to a portion of a file; this means that references to the segment are in fact references to that portion of the file. The length of the portion may range from 1 to 20000 bytes (octal). The portion may be attached at the lower end of the segment, so that the lowest address of the segment corresponds to the lowest address of the portion, in which case we say the segment expands upward; or the portion may be attached at the upper end of the segment, so that the highest address of the segment corresponds to the highest address of the portion, in which case we say the segment expands downward.

A segment may or may not have write access.

The portion may begin at any file address subject to the following restrictions, which are divided into two cases. If the segment expands upward, then the file address of the portion is the address of the first location in the portion, and (1) The file address must be a multiple of 100 (octal). (2) If the segment does not have write access, the file address plus the length of the portion must not exceed the length of the file. If the segment has write access, the length of the file will be increased if necessary to satisfy

the preceding condition. (3) The portion must not cross a 20000-byte boundary; in other words, the greatest integer in (file address/20000) must equal the greatest integer in ((file address + length of portion - 1)/20000).

If the segment expands downward, then the file address of the portion is one plus the address of the last location in the portion, and (1) The file address must be a multiple of 100 (octal). (2) If the segment does not have write access, the file address must not exceed the length of the file. If the segment has write access, the length of the file will be increased if necessary to satisfy the preceding condition.

(3) The portion must not cross a 20000-byte boundary; in other words, the greatest integer in ((file address - length of portion)/20000) must equal the greatest integer in ((file address - 1)/20000).

For example, suppose a process has locations 400 through 777 of a file attached to D-space segment 3, with upward expansion. Then when the process references location 60254,

it will be referencing location 654 of the file.

Attaching and detaching files does not imply any movement of the file between core and disk. Such movement occurs only when the segment which is attached to the file is referenced. Note that when a file is modified, there is no need to explicitly "write out" the modified version, because the modified version is the only version. The supervisor will move the modified portion onto secondary storage in the normal course of its paging operations.

Two or more processes may share a common file if each has a capability for the file.

A file capability has two possible attributes, namely <write> (bit 1 of the attribute field) and <D-space> (bit 0). The <set length> operation, and the <attach> operation specifying write access, are legal only if the invoked file capability has the write attribute. The <attach> operation specifying a D-space segment is legal only if the invoked file capability has the D-space attribute. A file capability lacking the write attribute is "read-only", and one lacking the D-space attribute is "execute-only".

The operations on a file will now be described.

EMT 3 <attach>

,BYTE 4,0

Attaches a portion of the file to a segment of the process executing the attach, unless an error condition occurs, as indicated by condition codes (see below). Any previous attachment to the segment is removed. Parameters for the attach are:

2(SP): bits 11-8 are the segment number (D-space is allowed only if the invoked capability has the D-space attribute.)

bit 3 is the expansion direction;
0 = upward, 1 = downward

bits 2-0 are the access control field;
2 = read-only, 6 = read/write
(Read/write is allowed only if the invoked capability has the write

attribute.)

4(SP): File address, least significant half.

6(SP): File address, most significant half.

10(SP): Length of the portion, in bytes.

Condition codes are set to indicate error conditions:

N and Z are set if the attach requests write access but the capability invoked does not have the write attribute, or the attach specifies a D-space segment but the capability invoked does not have the D-space attribute, or the attach requests read-only access for a portion extending beyond the end of the file.

N and V are set if the attach requests write access for a portion extending beyond the end of the file, and the supervisor is not able to increase the length of the file because of insufficient secondary storage.

N and C are set if the file address, portion length, or access control field is invalid.

EMT 4 <read length>

,BYTE 0,2

The length of the file, in bytes, is pushed onto the stack. 2(SP) is the most significant word, and (SP) is the least significant.

EMT 5 <set length>

,BYTE 2,0

The length of the file, in bytes, is set to the double-word value on the stack. 2(SP) contains the least significant word, and 4(SP) contains the most significant word. N and Z are set if the invoked file capability does not have the write attribute. N and V are set if the file is being lengthened and the supervisor has insufficient secondary storage.

Directory

A directory is a list of capabilities. A directory capability has three possible attributes, which specify

permissions for each of three operations, namely read (bit 0), append (bit 1), and delete (bit 2).

EMT 0 <retrieve>
,BYTE 201,0
The capability in the directory at the index in 4(SP) is copied into the invoking process's C-list at the index in 2(SP), unless any of the following error conditions occurs. If the destination index is not free, N and V are set. If the capability to be copied is an entered process capability, or if there are already 32767 capabilities referring to the object, N and C are set. If the invoked directory capability does not have the read attribute, N and Z are set.

EMT 1 <grant>
,BYTE 201,0
The capability in the invoking process's C-list at the index in 2(SP) is copied into the directory at the index in 4(SP), unless any of the following error conditions occurs. V and C are set as in <retrieve>, and N is also set in those cases. If the invoked directory capability does not have the append attribute, N and Z are set.

EMT 2 <delete>

,BYTE 1,0

The capability in the directory at the index in 2(SP), if any, is released. If the invoked directory capability does not have the delete attribute, C and N are set and the operation does not take place.

EMT 3 <attach>

,BYTE 4,0

Same as <attach> for files, except that write access is not permitted. This operation allows a process to determine the number, types, and indexes of the capabilities in the directory. The format of a capability is described in the listing of the RATS supervisor.

EMT 4 <read length>

,BYTE 0,2

Same as <read length> for files.

501

Page 12 501a

502

503

504

Process 504a

505

506

The basic facts about a process have been given above. A 507
process has a run indicator, which indicates whether the 508
process is allowed to execute instructions. The run indicator 509
is independent of state information indicating whether the 510
process is in a wait state. Only the event being waited for 511
can remove the process from a wait state. Thus, if the run 512
indicator is on, turning it off and then at some later time on 513
again will never disrupt the state of the process. 514

515

A process has two names associated with it for the 516
purpose of accounting. They are called the user-name and the 517
account-name. In most cases the two names will be the same. 518
(For cases in which they are different, see the section on 519
entries.) When a user is logged in, his process's user-name 520
and account-name are set equal to the name used at login. 521
Resources used by a process are accounted to its account-name. 522
A process's user-name only changes at login and logout. 523

When an exceptional condition occurs in a process, such as a HALT instruction, a <fault entry> occurs. See the section on entries for details of the entry mechanism. A fault number is passed to the process receiving the fault entry. The fault number indicates the type of fault:

Fault No.	Type of fault
0	HALT, odd address error
1	Reserved instructions
2	BPT with T-bit off
3	T-bit trap, or BPT with T-bit on
4	IOT
5	Other illegal instructions, such as invoking a capability with an invalid EMT code, invoking an entry with no masters, or invoking a nonexistent capability.
6 + X	Incorrect number of parameters passed. X, which is the high byte of the fault number, indicates the expected number of parameters. The stack is unchanged.
7 + X	Incorrect number of return parameters expected. X, which is the high byte of

the fault number, indicates the number returned. The stack is unchanged.

8 + X + N*16 Illegal memory reference. N is the number of the segment referenced. X indicates the type of violation:

X	Type of violation
1	Read-only
2	Segment length
3	Both read-only and segment length
4	Segment is not attached to anything
5	Segment is attached to a portion of a file which has been deleted

Other fault numbers are possible in conjunction with the "cause fault" operation on an entered process capability (q.v.).

For the proper operation of T-bit traps, it is necessary to distinguish two states of a process in which the values of

all registers (notably, the PC and PS) are the same. In both states, the PC contains the address of the next instruction. In one state, the process is about to execute that instruction; if the T-bit is on, a T-bit trap will occur at the end of that instruction. In the other state, if the T-bit is on, the process is about to perform a T-bit trap signaling the end of the previous instruction executed by the process. If the T-bit is off, no T-bit traps occur and the two states are indistinguishable; however, for consistency the two states are distinguished on the basis of what would happen if the T-bit were on.

To distinguish these two states, bit 8 of the PS is used. Termed the "RTT bit", it is on in the first state above and off in the second. This bit may be read and written along with the rest of the PS.

Due to deficiencies in the PDP11/45 hardware, the RTT bit may be accidentally cleared. Consequently, programs making use of T-bit traps should be prepared to receive spurious extra T-bit traps.

The operations on a process capability are:

EMT 0, 1, 2, 3, and 4

Same as for directory capabilities, substituting "invoked process's C-list" for "directory". The invoked process capability is considered to have full access attributes.

EMT 5 <wait>
.BYTE 200,1
Waits for a fault entry from the process. Similar to the <wait> operation on a master entry capability (q.v.). The

entered process capability which is created will have the fault entry attribute (bit 0).

EMT 6 <start>
.BYTE 0,0
Turns on the run indicator.

EMT 7 <stop>
.BYTE 0,0
Turns off the run indicator. Does not affect any other processes, not even ones owned by the invoked process.

EMT 10 <copy from>

,BYTE 2,0

Sets up an attachment in the invoking process at the segment specified by 2(SP) identical to that in the invoked process at the segment specified by 4(SP). The null attachment may be copied. Any previous attachment at the destination is removed. If the file or directory capability which was originally invoked to establish the source attachment did not have the D-space attribute, then the destination segment must not be in D-space; if not, C and N are set. If either segment number is > 17 (octal), V and N are set.

EMT 11 <copy to>

,BYTE 2,0

Similar to <copy from>. The attachment of the segment in 4(SP) in the invoking process is copied to the segment in 2(SP) in the invoked process. The process's run indicator must be off and the process must not be in a wait; if this is not the case, N and Z are set.

EMT 12 <read registers>

,BYTE 0,11

Pushes the PS and registers 7 through 0 of the process onto the stack. The run indicator must be off; if this is

not the case, N and Z are set and garbage is pushed. The PS and PC may indicate that the process is in supervisor mode. This means that the process was stopped while executing a supervisor call (e.g. EMT). There is no easy way to determine the location of the instruction which caused the supervisor call. If the process is restarted without modifying its registers, the supervisor call will proceed to completion normally.

EMT 13 <write registers>
.BYTE 11,0
Pops registers 0 through 7 and the PS off the stack. Only

the 4 condition codes, the T-bit, and the RTT bit of the PS are significant. The process's run indicator must be off and the process must not be in a wait; if this is not the case, N and Z are set. If the process was in supervisor mode, the supervisor call which was in progress is aborted.

Semaphore

For a discussion of what a semaphore is and how to use it, see "Cooperating Sequential Processes", by Dijkstra.

```
EMT 0 <p>
.BYTE 0,0

EMT 1 <v>
.BYTE 0,0

EMT 2 <read value>
.BYTE 0,1
```

Pushes onto the stack the value of the semaphore variable. If N processes are hung in the semaphore the value will be -N. Intended for debugging. A faster way to determine the value of the semaphore variable is to maintain an ordinary variable which is decremented every time a <p> is done and incremented on every <v>.

Entry

The entry facility allows the user to create programmed capabilities. Such capabilities, when invoked, can perform any desired function. For example, they can be made to simulate most of the other kinds of capabilities.

The entry facility comprises three kinds of capabilities: master entry, slave entry, and entered process. The process being serviced must own a slave entry capability, and the process which is performing the service must own a corresponding master entry capability.

The <Wait> operation on a master entry waits until a corresponding slave entry capability is also invoked. When this happens, the slave process is put in a wait and the master process is restarted. The master process is given an <entered process> capability (henceforth abbreviated EPC) referring to the slave process at the index specified in 2(SP).

The account-name of the master process is set equal to the account-name of the slave process. In effect, this

licenses the master to use the slave's account on his behalf. When the master returns control to the slave, the master's account-name is set equal to its user-name.

Upon successful completion of the <wait> operation, (SP) will contain the <transmitted information>. If the entry was caused by a process invoking a slave entry capability, the transmitted information has, in the low byte, the low byte of the slave's EMT, and, in the high byte, the attributes field of the invoked slave entry capability. If the entry is a fault entry, the transmitted information is the fault number.

The operations on a master entry capability are:

EMT 0 <create slave>
.BYTE 200,0
A slave entry capability corresponding to the invoked master entry capability is placed in the C-list at the index specified in 2(Sp). Its attribute field will be all ones. If the requested index is not free, N and V are set. If the slave entry capability cannot be created, N and Z are set.

EMT 5 <wait>
.BYTE 200,1

Wait for a slave process to enter. See description above.

2(SP) has the index for the entered process capability. If successful, the transmitted information is returned on the stack. The EPC which is created will have the EMT entry attribute (bit 1). If the specified index is not free, N and V are set and garbage is returned on the stack. If all slave capabilities for this entry have been deleted, N and Z are set and garbage is returned on the stack.

The following are the possible operations on an EPC. If the EPC does not have the EMT entry attribute, only the <restart> and <return> operations are allowed; if other operations are attempted, N and Z will be set. An EPC may not be <grant>ed or <retrieve>d.

EMT 0 <retrieve>
,BYTE 200,0
The capability in the slave process's C-list at the index specified by the C-list index parameter passed by the

slave is copied into the master's C-list at the index in 2(SP). If no C-list index parameter was passed, N and Z are set, V and C are set as in the directory operation <retrieve>, and N is also set in those cases.

EMT 1 <grant>

.BYTE 200,0

The capability in the master's C-list at the index in 2(SP) is copied into the slave process's C-list at the index specified by the C-list index parameter passed by the slave. Z, V, C, and N are set as for <retrieve> above.

EMT 2 <delete>

.BYTE 0,0

The capability in the slave process's C-list at the index specified by the C-list index parameter passed by the slave is released, N and Z are set as for <retrieve> above.

EMT 3 <read parameters>

.BYTE 0, C + X

In the parameter control word for this operation, 0 <= X <= 177, and C = 0 or 200. The X parameters passed by the slave, excluding the C-list index parameter, are copied

onto the master's stack. They will appear in the same order in the master's stack as they did in the slave's stack. If the number of parameters passed by the slave is not equal to X, or the C-list index parameter bit in the slave's parameter control word does not match C, N and V

are set, and, if X > 0, the slave's parameter control word is pushed onto the master's stack, followed by X-1 words of garbage.

EMT 4 <cause fault>
.BYTE 1,0
The slave process executes a fault entry. 2(SP) specifies the fault number. The slave's stack pointer will have its value at the time of the enter. The EPC is released from the master's C-list, and the master's account-name is set equal to its user-name.

EMT 5 <restart>
.BYTE 0,0
The slave process is restarted at the instruction which

caused the enter. The slave's stack pointer will have its value at the time of the enter. Hence, unless something has changed, it will reenter. Things that might have changed include the C-list, the T-bit, or the run indicator. The EPC is released from the master's C-list, and the master's account-name is set equal to its user-name.

EMT 6 <return>
,BYTE X+1,0

The slave process is restarted after the instruction which caused the enter. If the EPC has the fault entry attribute, the slave's PC is stepped one word. If the EPC has the EMT entry attribute, the slave's PC is stepped two words (to skip over the parameter control word). Hence, the slave's EMT will appear to complete. 2(SP) specifies condition codes that are to be set for the slave. If the EPC has the EMT entry attribute, X parameters are copied from the master's stack to the slave's stack. They will appear in the same order in the slave's stack as in the master's stack. If the number of return parameters requested by the slave is not equal to X, N and V are set, and an appropriate fault is caused for the slave. If the EPC does not have the EMT entry attribute, X must be zero. The EPC is released from the master's C-list, and the

master's account-name is set equal to its user-name. The slave process's RTT bit will be cleared, so that if its T-bit is on, a T-trap will occur. Exception: if the entry was a fault entry due to a T-bit trap, the RTT bit was already off; in this case, it is turned on, so that another T-bit trap will not occur until the next instruction has been executed. If the entry was a fault entry due to a HALT or odd address error, <return> should not be used; <restart> will cause execution to resume at

the address in the PC, but in the case of an odd address error, that is not necessarily the address of the next instruction.

Supervisor Capability

EMT 0 <release capability>

```
.BYTE 200,0
```
The capability at the index specified in 2(SP), if any, is removed from the C-list. The general remarks above about releasing capabilities apply.

```
EMT 1 <create file>
.BYTE 200,0
```
A new file of length zero is created and a capability for the file, with write and D-space attributes, is placed in the C-list at the index specified in 2(SP). If the file cannot be created, N and Z are set. If the requested index is not free, N and V are set.

```
EMT 2 <create directory>
.BYTE 200,0
```
A new directory is created and a capability for it, with full access attributes, is placed in the C-list at the index specified in 2(SP). The directory is initially empty. If the directory cannot be created, N and Z are set. If the requested index is not free, N and V are set.

```
EMT 3 <create process>
.BYTE 200,0
```
A new process is created and a capability for it is placed in the C-list at the index specified in 2(SP). That index

must initially contain a capability for a directory, which will be used as the C-list. The eight registers are initially zero, the PS is initially 174400, and the process has no attachments. The run indicator is off. The user-name and account-name of the process are set equal to the account-name of the process executing the <create process>. If the process cannot be created or 2(SP) does not refer to a directory capability with full access attributes, N and Z are set.

EMT 4 <create semaphore>
.BYTE 201,0
A new semaphore is created and a capability for it is placed in the C-list at the index specified in 2(SP). The initial value of the semaphore variable will be 4(SP), which must be positive or zero. If the semaphore cannot be created or 4(SP) is negative, N and Z are set. If the requested index is not free, N and V are set.

EMT 5 <create entry>
.BYTE 200,0

A new entry is created, and a master entry capability for it is placed in the C-list at the index specified in 2(SP). If the entry cannot be created, N and Z are set. If the requested index is not free, N and V are set.

EMT 6 <read segment properties>
,BYTE 1,2
Reads properties of the segment whose number is in 2(SP). If 2(SP) > 17 (octal), V and N are set and two words of garbage are pushed. Otherwise, pushes the length of the segment in bytes, and the expansion direction and access control field in the same format as the file operation attach. (If someone thinks the file number and file address are useful, they could be provided also.) If the segment is not attached to anything, two zero words are pushed.

EMT 7 <detach>
,BYTE 1,0
Removes the attachment (if any) to the segment whose number is in 2(SP). If 2(SP) > 17 (octal), V and N are set.

EMT 10 <remove attributes>

,BYTE 201,0

Removes attributes from the capability whose index is in 2(SP). Each bit in the low byte of 4(SP), if set, clears the corresponding bit in the attribute field of the capability. There is no error indication if an attribute to be removed is already gone.

EMT 11 <read calendar clock>

,BYTE 0,3

Returns on the stack, in 4(SP), the number of days since 1 January 1901, and a two-word number giving the number of 1/60th's of a second since last midnight. 2(SP) is most significant, and (SP) is least significant. Greenwich Mean Time is used.

EMT 12 <wait on calendar clock>

,BYTE 3,0

Waits until the calendar clock time is greater than or equal to the time on the stack. The format of the time is the same as for <read calendar clock>. If the given time is invalid, N and C are set.

938
938a
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961

Part Two

When a user sits down at a terminal which is connected to RATS, he is typing to a program called EXEC. EXEC is intended to be self-documenting; in most cases, typing "HELP" will give the user all the assistance he needs.

EXEC has a facility for allowing users to run their own programs. This facility is presently described in a separate memo. Certain features of the environment of a user process are described here.

The C-list initially contains:

Index	Capability
0	Supervisor
2	C-list directory
3	Public directory
4	Terminal Input
5	Terminal Output
6	Code file
7	Stack file

The supervisor capability is described in part one. The C-list directory capability is an ordinary directory capability which refers to the C-list. This is used to copy capabilities from one C-list index to another.

The code file is a file containing the program to be executed by the process. The process will initially be given an attachment to this file in segment zero of I-space. In the case of EXEC, this file is called the system file, and contains code for all permanent system programs. Normally, the code file will be read-only, to allow for reentrant programs.

The stack file is a file which can be used for the stack, variables, and all other storage which must be private to each process (or group of processes) executing the program.

Terminal Input Capability

A terminal input capability has the following operation:

EMT 0 <read>

.BYTE 203,1

Parameters are:

2(SP): Index of a capability for a file containing space for characters to be placed. Must have write and D-space attributes.

4(SP): Low file address of beginning of the space for characters.

6(SP): High file address of beginning of the space.

10(SP): Maximum number of characters to be put into the space.

This operation waits until at least one character which has not been read by a previous <read> has been input on the terminal. Then all input characters which have not been read by a previous <read> (up to the maximum) are put into the space provided, one character per byte. The space must not cross a 20000-byte boundary in the file. The number of characters read is returned on the stack. Zero is returned if there is any error in the parameters supplied. All RATS terminals are full duplex.

The format of the characters returned is as follows.

If bit 7 of the byte is zero, then bits 6-0 contain an

ASCII character. If bit 7 of the byte is one, then some error occured on this character, as given by other bits in the byte, as follows. Bit 6 is on if one or more characters were lost at this point in the input, because either the input buffer overflowed or (unlikely) the interrupt handler didn't respond to an interrupt in time. Bit 5 is on if a break was received (i.e. a character with no stop bit). Bit 4 is on if a character with bad parity was received.

Terminal Output Capability

A terminal output capability has the following operation:

```
EMT 0 <write>
,BYTE 203,0
```

Parameters are:

2(SP): Index of a capability for a file containing characters to be output. Must have D-space attribute.

4(SP): Low file address of beginning of character string
6(SP): High file address of beginning of character string
10(SP): Number of characters in the string

This operation outputs the character string on the terminal. The character string must not cross a 20000-byte boundary in the file. Each byte contains an ASCII character; bit 7 is ignored. This operation may wait some length of time before returning, if output buffers are full. The user need not be concerned with padding carriage returns, or other timing considerations. After the EMT returns, the character string may be overwritten without affecting the output. It is recommended that the character string not be longer than 100 characters, since once a string begins being output there is no way to stop it. If there is any error in the parameters supplied, N is set.

Public Directory

This directory contains capabilities for accessing I/O

devices and other resources of general utility. It contains:

Index	Capability
6	System file (read-only)
10	Phone Handler capability
11	Line Printer capability
12	Card Reader Handler capability (not implemented yet)
13	Paper Tape Reader Handler capability
14	Network Control Program capability
20	GETRUN Code File (read-only)

The system file is a file containing the RATS supervisor, the I/O handlers, and the EXEC. Starting addresses of each of the programs in the file can be found in the listing of the RATS supervisor.

The GETRUN Code File contains a program designed to load other programs. For more information see J. E. Donnelley.

Phone Handler Capability 1079a

1080

A phone handler capability has the following operation: 1081

1082

EMT 0 <call> 1083

,BYTE 211,1 1084

1085

The <call> operation allocates a phone line, dials 1086
the requested number (if possible), and sets the line 1087
operating at the requested baud rate. Available baud 1088
rates are 110, 134.5, 150, and 300 (asynchronous) and 2000 1089
(synchronous). 2000 baud transmission is not implemented 1090
yet. Parameters are: 1091

1092

2(SP): Index of a directory capability with append 1093
attribute. 1094

1095

4(SP): Index in that directory to receive a phone input 1096
capability (see below). 1097

1098

6(SP): Index in that directory to receive a phone output 1099
capability (see below). 1100

1101

10(SP): Baud rate at which phone line is to operate. For 1102
134.5 baud, 10(SP) should contain 134. 1103

12(SP): through 24(SP): phone number to be called, one digit per byte. Sucessive digits are in successive bytes (i.e. increasing addresses). Only the low 4 bits of each byte are significant. The number must be in standard form for direct distance dialing, that is: the digit 1; a 3-digit area code; a 3-digit prefix; a 4-digit extension.

Two capabilities for operating the phone line, described below, are returned. A parameter is returned to indicate the outcome of the request:

0 Successful. The number was dialed automatically, and carrier was established. Phone input and output capabilities are returned.

1 Busy. The number was dialed automatically, but carrier was not established within a reasonable time. This could be due to calling a phone which is busy, or which is not equipped with data communications equipment (e.g. a wrong number). It may be advisable to try the call again.

2 No phone line could be allocated. Try again after some other RATS user has deallocated a phone line.

3 Insufficient resources. The phone handler was unable to create a process, entry, or semaphore. Try again when system resources are less heavily loaded.

4 Error. An error in a passed parameter was detected (e.g. invalid baud rate or phone number).

X > 100 A return code greater than 100 (decimal) indicates that a phone line has been allocated but the number could not be dialed automatically. X is the last four digits of the phone line from which the call must be manually dialed. Phone input and output capabilities are returned.

If the call was dialed automatically, deleting the phone

output capability will hang up the phone. If the phone was dialed manually it must be hung up manually. Deleting both phone input and output capabilities results in the phone line being deallocated (i.e. available for other use).

Phone Input Capability

A phone input capability has the following operation:

EMT 0 <read>
,BYTE 203,1
Parameters are:

2(SP): Index of a capability for a file containing space for characters to be placed. Must have write and D-space attributes.

4(SP): Least significant word of the address in the file of the beginning of the space for the characters to be placed. Must be even.

6(SP): Most significant word of the file address.

10(SP): Size of the reserved space, in bytes. Must be

even. The space must not cross a 20000 byte boundary.

Page 29

The <read> operation waits until either (1) a character which has not been read by a previous <read> has been received from the phone line; (2) carrier detect changes; or (3) data set ready is off (indicating the phone is on-hook). It then returns, in the space provided, one or more words containing either a character or status information. It returns on the stack a parameter which is the number of bytes of the space which were actually used (i.e. 2 times the number of words returned). This parameter will be zero if there is an error in the passed parameters.

The meaning of the words returned is as follows. If bit 15 is off, then bits 7-0 contain a character which was received from the phone line, and bit 12 has the parity of bits 7-0. If bit 15 is on, the word contains status information, as follows:

Bit 14 is on if one or more characters were lost due to buffer overflow.

Bit 13 is on if a break was received. (Note: Some phone line interfaces cannot recognize breaks. On these lines, a break will be interpreted as a series of null characters.)

Bit 11 is on if the word contains status:

Bit 10 indicates the status of carrier detect

Bit 9 is on if an outgoing call is in progress. It is off if the phone line is on-hook or an incoming call is in progress.

Phone Output Capability

A phone output capability has the following operation:

```
EMT 0 <write>
,BYTE 203,0
```

Parameters are:

2(SP): Index of a capability for a file containing

characters to be transmitted. Must have the D-space attribute.

4(SP): Least significant word of the address in the file of the beginning of the string of characters. Must be even.

6(SP): Most significant word of the file address.

10(SP): Size of the string of characters, in bytes. Must be even. The string must not cross a 20000-byte boundary.

The <write> operation transmits the string of characters on the phone line. Each character occupies one word. If bit 15 is zero, bits 7-0 contain the character to be transmitted. If bit 15 is one, a break is sent for one character time.

Errors are indicated by returned condition codes. N and Z are set if no outgoing call is in progress on the

line (see above). (Some characters may have been transmitted successfully.) N and C are set if any passed parameter is in error.

Page 31

Line Printer Capability

The line printer on the RISOS system is a Versatec matrix printer, with 132 columns and 54 lines per page. The line printer capability has the following operation:

EMT 0 <print>
.BYTE 200,0
2(SP) has the index of a file to be printed. The entire file will be printed, preceded by a page containing the calling process's account-name. Each byte of the file is an ASCII character; bit 7 of each byte is ignored. Output is buffered, so the <print> operation returns immediately, but the file may not be printed for a while. After the <print> operation returns, the file may be overwritten without affecting the output.

If any parameter is in error, N and C are set. If the line printer handler's buffer is full or enough file space cannot be created, N and V are set. If the line printer is off-line, out of paper, etc., Z (but not N) is set, a message is printed on the operator's console, and the file will be printed when the printer becomes ready. After the printer is made ready, another <print> operation must be done to resume output. If the printer becomes not ready while a file is being printed but after the <print> operation has returned, a message is printed on the operator's console, but no error indication can be given to the user who did the <print>.

Card Reader Handler Capability

A card reader handler capability has the following operation:

EMT 0 <assign card reader>
,BYTE 200,0

This operation returns a card reader capability at

the C-list index in 2(SP). The possessor of a card reader capability has exclusive use of the card reader until he deletes the card reader capability. If the card reader is already assigned to someone else, N and Z are set. If the C-list index passed is not free, N and C are set.

Card Reader Capability

A card reader capability has the following operation:

```
EMT 0 <read>
.BYTE 200,0
```

This operation reads a deck of cards and returns a data file in the index in 2(SP). The file has the following format. For each card read, there is one word of data for each column (usually 80), followed by a word containing 040000 (octal). The column data is in the following format:

Bit	Card Zone
15-12	unused (zero)
11	12
10	11

9	10	1320
8	1	1321
7	2	1322
6	3	1323
5	4	1324
4	5	1325
3	6	1326
2	7	1327
1	8	1328
0	9	1329

1330

After all card data, there is a word containing the 1331
status of the reader when reading terminated. The 1332
following bits are significant: 1333

1334

 1334a

1335

1336

1337

Bit 15 = always on 1338
Bit 13 = input hopper empty (i.e. normal 1339
termination) or output stacker full 1340
Bit 12 = card reader check (e.g. card jam) 1341
Bit 11 = timing error (indicates that data was lost) 1342
Bit 8 = reader was off-line 1343

If not enough file space can be created for the deck, N and V are set and no file is returned. If the C-list index passed is not free, N and C are set.

Page 34

Paper Tape Reader Handler Capability

A paper tape reader handler capability has the following operation:

EMT 0 <assign paper tape reader>
.BYTE 200,0
This operation returns a paper tape reader capability at the C-list index in 2(SP). The possessor of a paper tape reader capability has exclusive use of the paper tape reader until he deletes the paper tape reader capability. If the paper tape reader is already assigned to someone else, N and Z are set. If the C-list index passed is not free, N and C are set.

Paper Tape Reader Capability

A paper tape reader capability has the following operation:

EMT 0 <read>
,BYTE 200,0

This operation reads a paper tape and returns a data file in the index in 2(SP). The file will contain one byte for each line of tape read. Leaders and trailers are not stripped. If not enough file space can be created for the tape, N and V are set and no file is returned. If the C-list index passed is not free, N and C are set.

Network Control Program Capability

The Network Control Program (NCP) handles all communication with the IMP. Familiarity with the ARPA network Host-Host Protocol (described in NIC #8246 and #7104) is assumed.

EMT 0 <reserve socket> 1392

.BYTE 202,0 1393

The socket number specified in 4(SP) (least 1394
significant) and 6(SP) (most significant) is reserved. A 1395
socket capability is returned at the index in 2(SP). If 1396
that index is not free, N and C are set. If the requested 1397
socket is already reserved, N and Z are set. If the NCP 1398
cannot create an entry or process or enough file space to 1399
handle the socket, N and V are set. 1400

1401

EMT 1 <reserve sockets> 1402

.BYTE 202,2 1403

This operation reserves 6(SP) consecutive socket 1404
numbers, beginning with an even socket number selected 1405
arbitrarily by the NCP. The first socket number reserved 1406
is returned in (SP) (least significant) and 2(SP) (most 1407
significant). Socket capabilities for each socket 1408
reserved are returned in the directory specified by the 1409
index in 2(Sp), at respectively consecutive indexes 1410
beginning with the index in 4(SP). 1411

1412

If no set of 6(SP) consecutive sockets is available, 1413
N and Z are set and garbage is returned. If the NCP 1414
cannot create enough entries, processes, and file space to 1415
handle the sockets, N and V are set and garbage is 1416

returned. If 6(SP) is equal to zero or greater than 256, or if 2(SP) does not refer to a directory capability with the append attribute, or if not all the specified indexes in the directory are free, N and C are set and garbage is returned.

Socket Capability

A socket capability provides the mechanism for performing certain operations relating to its associated socket number. Briefly, the operations are:

<listen> Attempts to establish a connection by waiting for a Request-For-Connection (RFC) and then returning a matching RFC.

<init> Attempts to establish a connection by sending an RFC and waiting for a matching RFC.

<close> Used to (1) abort a <listen> or <init>, (2)

initiate closing of a connection, or (3) acknowledge closing of a connection by the foreign host.

<inactivate connection> Used to effectively close a connection even if the foreign host is slow in responding to a CLS.

<read> Receives data over the connection.

<write> Sends data over the connection.

<send INR> Sends the control message INR on this connection.

<send INS> Sends the control message INS on this connection.

<wait for INS> Waits for the control message INS on this connection.

<wait for INR> Waits for the control message INR on this connection.

A local socket is defined to be "active" between the time a <listen> or <init> is executed and a <close> is executed. Exactly one <close> must be executed for each <listen> or

<init>, even if a connection is closed at the instigation of the foreign host.

There are two typical scenarios for establishing and breaking a connection. In case 1, a connection is established with <listen> or <init>. Sometime later the foreign host closes the connection. The local process must eventually acknowledge by doing a <close>.

In case 2, a connection is established with <listen> or <init> as before. Sometime later the local process does a <close> to close the connection. The foreign host eventually acknowledges the close by sending a CLS.

In case 2, if the local socket is receiving data, some data may arrive between the time the local process does a <close> (which causes a CLS to be sent) and the time the foreign host acknowledges the close. We wish to allow the local process to <read> this data if it chooses; at the same time we do not want to force the local process to wait for the

foreign host to acknowledge the close before establishing a new connection, since the local socket may be a scarce resource such as the logger socket.

Accordingly, we make the following definition. A connection (not to be confused with a local socket) is "active" between the time an RFC is sent and either a CLS is received or an <inactivate connection> is executed. A connection can remain active after the local socket has become inactive; this is to allow the foreign host time to process the CLS. During this time, <read>s may be done. Only one connection to a given local socket can be active at once.

A socket number is reserved as long as there are any capabilities referring to the socket. Deleting a socket capability (i.e. releasing all copies of it): (1) does a <close> if the socket is active; (2) does an <inactivate connection>; and (3) un-reserves the socket number. RFC's received by the NCP are queued if the local socket they refer to is reserved; otherwise they are refused.

Receive (even) Socket Capability Operations

EMT 0 <listen>

,BYTE 0,4 1514
The local socket must be inactive and there must be 1515
no active connection to this socket; if this is not the 1516
case, N and C are set and garbage is returned. The local 1517
socket is made active. The <listen> operation then waits 1518
until either an RFC for this socket has been received from 1519
any host, or a <close> is done. 1520
1521
In the former case, an RFC is returned, opening the 1522
connection. The NCP assigns a link automatically. The 1523
connection byte size is returned in (SP). The foreign 1524
host number is returned in 2(SP), and the foreign socket 1525
number in 4(SP) (least significant) and 6(SP) (most 1526
significant). 1527
1528
If a <close> is done before any RFC is received, N 1529
and Z are set and garbage is returned. 1530
1531
EMT 1 <init> 1532
,BYTE 3,1 1533
The local socket must be inactive, there must be no 1534
<send INR> in progress, and the foreign socket specified 1535
in 4(SP) and 6(SP) must be odd; if this is not the case, N 1536
and C are set and garbage is returned. If there is an 1537
1538

active connection to this socket, it is made inactive. The local socket is made active. An RFC is sent to the host in 2(SP) and socket number in 4(SP) (least significant) and 6(SP) (most significant). The NCP assigns a link automatically. The <init> operation then waits until one of the following occurs:

(1) A matching RFC is received, opening the connection. The connection byte size is returned in (SP).

(2) The RFC is refused, or the RFC could not be delivered because either the foreign host is dead or the foreign IMP cannot be reached. N and V are set and a code is returned in (SP) telling which happened:

-1: Refused

0: Foreign IMP cannot be reached

1: Foreign host is dead

(3) A <close> is performed. The RFC is aborted by sending a CLS. N and Z are set and garbage is returned. This operation is intended to provide the user with a facility for timing out RFC's; the NCP never aborts an RFC unless a <close> is done.

EMT 2 <close> 1555
.BYTE 0,0 1556
The local socket should be active. It is made 1557
inactive. If a <listen> or <init> is in progress, it is 1558
aborted (q.v.). If there is an active connection to this 1559
socket, a close is initiated. (The connection will remain 1560
active until the foreign host acknowledges the close or 1561
another <listen> or <init> is executed.) 1562
1563
If the local socket is not active, the <close> will 1564
apply to the next <listen> or <init>. If one such <close> 1565
has already been saved when a second is attempted, N is 1566
set. 1567
1568
EMT 3 <inactivate connection> 1569
.BYTE 0,0 1570
The local socket must be inactive and there must be 1571
no <send INT> in progress; if this is not the case, N and 1572
C are set. If there is an active connection to this 1573
socket, it is made inactive. Any messages arriving on the 1574
inactive connection will be discarded. 1575
1576
EMT 4 <read> 1577
.BYTE 203,1 1578
There must be no <listen>, <init>, or <read> already 1579

in progress; if this is not the case, or there is any error in the parameters supplied, N and C are set and zero is returned. The <read> operation waits until either

there is some data to be read or there is no active connection to this socket (e.g. the foreign host closed the connection). If there is data to be read, it is transferred to the file whose index is in 2(SP). The file capability must have the write and D-space attributes. The file address of the beginning of the area for the data is in 4(SP) (least significant) and 6(SP) (most significant). 10(SP) contains the number of 8-bit bytes in the area; it should be at least 1012 (decimal), in order to insure that all messages can be received. The area must not cross a 20000-byte boundary. The connection byte count for the data (i.e. the number of bits of data divided by the connection byte size) is returned in (SP). The connection byte count will be greater than zero. Regardless of the connection byte size, bits are stored in successive 8-bit bytes, high order bit first. Users are reminded of the principle of the Host-Host protocol that

no significance may be inferred from message boundaries by a receiving process.

If there is no active connection to this socket and no data to be read, N and Z are set and zero is returned.

EMT 5 <send INR>
.BYTE 0,1
The local socket must be active and there must be no <listen>, <init>, or <send INR> already in progress; if this is not the case, N and C are set and garbage is returned. If there is no active connection to this socket, N and Z are set and garbage is returned. Otherwise, an INR (INterrupt-by-Receiver) is sent on the connection. A code is returned in (SP) indicating the outcome of the transmission, as for <write> (q.v.).

EMT 6 <wait for INS>
.BYTE 0,0
(Not yet implemented.)

Send (odd) Socket Capability Operations

EMT 0 <listen>

,BYTE 1,4
The local socket must be inactive and there must be no active connection to this socket; if this is not the case, N and C are set and garbage is returned. The local socket is made active. The <listen> operation then waits until either an RFC for this socket has been received from any host, or a <close> is done.

In the former case, an RFC is returned, opening the connection. 2(SP) specifies the connection byte size to be used. The connection byte size is returned in (SP). The foreign host number is returned in 2(SP), and the foreign socket number in 4(SP) (least significant) and 6(SP) (most significant).

If a <close> is done before any RFC is received, N and Z are set and garbage is returned.

EMT 1 <init>
,BYTE 4,1
The local socket must be inactive, there must be no

<write> or <send INS> in progress, and the foreign socket specified in 6(SP) and 10(SP) must be even; if this is not the case, N and C are set and garbage is returned. If there is an active connection to this socket, it is made inactive. The local socket is made active. An RFC is sent to the host in 4(SP) and socket number in 6(SP) (least significant) and 10(SP) (most significant). 2(SP) specifies the connection byte size to be used. The <init> operation then behaves the same as <init> for receive sockets (q.v.).

EMT 2 <close>
.BYTE 0,0
Same as <close> for receive sockets (q.v.).

EMT 3 <inactivate connection>
.BYTE 0,0
The local socket must be inactive and there must be no <write> or <send INS> in progress; if this is not the case, N and C are set. If there is an active connection to this socket, it is made inactive.

EMT 4 <write>
.BYTE 203,1
The local socket must be active and there must be no

<listen>, <init>, or <write> already in progress; if this is not the case, or if there is any error in the parameters supplied, N and C are set and garbage is returned. If there is no active connection to this socket, N and Z are set and garbage is returned. Otherwise, data is sent over the connection. The data is taken from the file whose index is in 2(SP). The file capability must have the D-space attribute. The file address of the beginning of the data is in 4(SP) (least significant) and 6(SP) (most significant). 10(SP) contains the connection byte count, which may be zero.

There must be fewer than 8096 bits of data. The data must not cross a 20000-byte boundary. Regardless of the connection byte size, data bits are taken from successive 8-bit bytes, high order bit first. A code is returned in (SP) indicating the outcome of the transmission:

-1: Successful
0: Unsuccessful because foreign IMP cannot be reached
1: Unsuccessful because foreign host is dead

EMT 5 <send INS> 1702

,BYTE 0,1 1703

The local socket must be active and there must be no 1704
<listen>, <init>, or <send INS> already in progress; if 1705
this is not the case, N and C are set and garbage is 1706
returned. If there is no active connection to this 1707
socket, N and Z are set and garbage is returned. 1708
Otherwise, an INS (INterrupt-by-Sender) is sent on the 1709
connection. A code is returned in (SP) indicating the 1710
outcome of the transmission, as for <write> (q.v.). 1711

1712

EMT 6 <wait for INR> 1713

,BYTE 0,0 1714

(Not yet implemented.) 1715

1716

------- 1717

1718

GSG 13-OCT-74 11:17 24212

RATS Programmer's Handbook

(J24212) 13-OCT-74 11:17; Title: Author(s): Geoffrey S. Goodfellow/GSG ; Distribution: / JAKE LLL-RISOS GSG JED; Sub-Collections: NIC LLL-RISOS; Clerk: GSG;

DCE 13-OCT-74 20:11 24213

Meetings for all ARC on Monday 14 Oct

Jim and Dick, please coordinate and announce times as early Monday as possible.

DCE 13-OCT-74 20:11 24213

Meetings for all ARC on Monday 14 Oct

I'd like for every one in ARC who is interested to have a chance Monday to hear the relevant experiences of our six staff members who were away. RWW had scheduled a meeting of his Development group at 3 pm for this purpose -- I'd like for him to open it for all of interested ARC (if other business, for Dev-group only, please defer that part). I'd like for JCN to schedule an adjoining time to cover the items relevant to all regarding his travels. If Dick and Jim don't get out a general announcement, let's count on all who are interested meet in conf. room at 2:30 for first JCN's 10-day summary, then as much Dev-Group as deemed of general interest. 1

DCE 13-OCT-74 20:11 24213

Meetings for all ARC on Monday 14 Oct

(J24213) 13-OCT-74 20:11;;;; Title: Author(s): Douglas C. Engelbart/DCE; Sub-Collections: SRI-ARC; Clerk: DCE;

RLL 14-OCT-74 09:52 24214

bugs: illegal statemet return ring....

This occurred on Sunday, 13 Oct 74

RLL 14-OCT-74 09:52 24214

bugs: illegal statemet return ring....

Just got the same old bug 'illegal statement return ring in copysrring.' 1

This occurred after the Jump File Return command and several spaces (to step through the list). 1a

Other facts: I did not cycle around and I did use slit screens during the NLS session. 1b

Also, immediately after this, I find myself unable to jump to any spot on the screen. The JUmp command gives the message 'file numbers to not match in storesring'. This also happened for other jump commands. Had to do a reset. 1c

RLL 14-OCT-74 09:52 24214

bugs: illegal statemet return ring....

(J24214) 14-OCT-74 09:52;;;; Title: Author(s): Robert N. Lieberman/RLL; Distribution: /FDBK([ACTION]) JDH([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: RLL;

JBP 14-OCT-74 10:20 24215

RFC Number Assignment in new system

Using preview on Office--1 i was unable to preassign an rfc number, The sendmail command reserve rfc asks some questions, one of which is "insert the number list" which i believe is a strange way of asking if the number is to be inserted into the text of the document, It is strange that this question is ask even if the document is offline, further the question indicated that it can be answered yes or no but the no answer is not accepted,
--jon,

1

JBP 14-OCT-74 10:20 24215

RFC Number Assignment in new system

(J24215) 14-OCT-74 10:20;;;; Title: Author(s): Jonathan B. Postel/JBP; Distribution: /BUGS([ACTION]) ; Sub-Collections: SRI-ARC BUGS; Clerk: JBP;

NIC 14-OCT-74 17:55 24216

Notice of all-ARC meetings

I would like to suggest that we have more than one day's notice before an all-ARC meeting. Monday I was working at home and could easily have come in except I was not aware that there was going to be a meeting. Sorry I missed it. 1

NIC 14-OCT-74 17:55 24216

Notice of all-ARC meetings

(J24216) 14-OCT-74 17:55;;;; Title: Author(s): Elizabeth J. (Jake) Feinler, Michael D. Kudlick/NIC; Distribution: /SRI-ARC([INFO-ONLY]) ; Sub-Collections: NIC SRI-ARC; Clerk: NIC;

New NLS loaded Monday Oct 14, 1974 18:30

NEW NLS loaded OCT, 14, 18:30 1

I Loaded a new NLS Monday night, The following bugs were fixed 1a

tnls print bug which caused tenex to perform line folding 1a1

set protection bug 1a2

spelling of SYNTAX corrected 1a3

The following modifications were made: 1b

Viewspecs OFF (and Viewspecs ON) Commands removed from USEROPTIONS 1b1

Leveladjust prompting ON / OFF has been removed from USEROPTIONS 1b2

The Playback Record (of session) (from file) OLDFILELINK now asks (Simulate recorded timing?) , and expects an answer construct (Yes or NO) 1b3

The implemntation o Process Commands has been changed slightly to speed it up , No change to the User Interface. 1b4

The following files have changed since nic-nls was last copied 1c

Compile auxcod 1c1

Compile fdata 1c2

Compile frontend 1c3

Compile inpfbk 1c4

Compile 110data 1c5

Compile 110runtime 1c6

Compile psedit 1c7

Compile sdata 1c8

Compile select 1c9

Compile srecords 1c10

Compile syntax 1c11

New NLS loaded Monday Oct 14, 1974 18:30

Compile tsprt 1c12

Compile utilty 1c13

DSM 14-OCT-74 18:56 24217

New NLS loaded Monday Oct 14, 1974 18:30

(J24217) 14-OCT-74 18:56;;;; Title: Author(s): David S. Maynard/DSM; Distribution: /KIRK([ACTION]) CHI([INFO-ONLY]) JDH([INFO-ONLY]) DSM([INFO-ONLY]) EKM([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: DSM;

JAKE 14-OCT-74 19:55 24218

Ident system

A very important portion of the ident system, namely the interrogation system has not yet been implemented. In addition there is currently no protection at all on the ident system - anyone who can load it can write on it. It is virtually impossible to enter information in a meaninful way at this point and I feel I cannot handle the database under existing circumstances. Are there any plans to do anything about this?
If so please let me know. 1

JAKE 14-OCT-74 19:55 24218

Ident system

(J24218) 14-OCT-74 19:55; Title: Author(s): Elizabeth J. (Jake) Feinler/JAKE; Distribution: /DCE(fyi only) JCN RWW EKM; Sub-Collections: SRI-ARC; Clerk: JAKE;

JDH 15-OCT-74 12:18 24219

Insert Number List

The "insert the number list" option (well, it is supposed to be optional) is NOT for te purpose of putting the number in the document. It is merely a bookeeping aid to make it easy to maintain a list of numbers somewhere for your reference. 1

JDH 15-OCT-74 12:18 24219

Insert Number List

(J24219) 15-OCT-74 12:18;;;; Title: Author(s): J. D. Hopper/JDH; Distribution: /JBP([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: JDH;

DVN 15-OCT-74 13:42 24220

The Next Move In DPCS for Montgomery

Naturally I am interested in the possibilitiesof NLS publications services to the people in Montgomery, What is the next move? 1

DVN 15-OCT-74 13:42 24220

The Next Move In DPCS for Montgomery

(J24220) 15-OCT-74 13:42;;;; Title: Author(s): Dirk H. Van Nouhuys/DVN; Distribution: /RWW([ACTION]) EKM([INFO-ONLY]) DCE([INFO-ONLY] Do you agree we should give Nielsen an idént so he can get journal mail?) JOAN([INFO-ONLY] would you start a DPCS notebook like the DIRT notebook and put this item in it?) EKM([INFO-ONLY]) ; Sub-Collections: DPCS SRI-ARC; Clerk: DVN;

RLL 15-OCT-74 13:53 24221

Visit and demo: Marlene Beckman of LEAA (dept of Justice)

(DATE) 11 October 1974 1

(BY) Lieberman (RLL) 2

(ATTENDEES) Marlene Beckman (Idnum) - LEAA 3

(MEDIUM) Medium of contact FACE-TO-FACE 4

(WHERE) SRI-ARC 5

(ACTION-ITEMS) none 6

(DISTRIBUTION) DCE JCN 7

(REMARKS) 8

Marlene Beckman of the Dept of Justice, Law Enforcement Assistance Administration (LEAA) visited ARC on Friday the 11th. I gave her a brief demonstration (1 hour) of DNLS and introduced her to the community concepts. 8a

Her office is responsible for giving grants and contracts to various consultants and local governments for studies, developments, and operational systems dealing with law enforcement. 8b

Marlene's group is concerned with correctonal institutions. She was out here (Calif.) monitoring a grant to a consulting firm responsible for surveying the treatment of females in the prisons of 11 states. 8b1

The nature of LEAA appears to parallel that of ARPA but with much less money and, of course in a different field (also, more emphasis on "practical" stuff, i.e., what is now happening and funding the people to change it.) 8b2

Our interest might be in serving her office or one (or more) of their grantees. 8b3

Overall this is the first contact with just another arm of the federal government which we should let lay (until someone else pops up). 8b4

I think she was impressed with some (not all) the advantages of our augmented environment. 8b5

RLL 15-OCT-74 13:53 24221

Visit and demo: Marlene Beckman of LEAA (dept of Justice)

(J24221) 15-OCT-74 13:53;;;; Title: Author(s): Robert N. Lieberman/RLL; Distribution: /DCE([INFO-ONLY]) JCN([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: RLL;

RLL 15-OCT-74 13:57 24222

To be sent message for review; help from the KWAC.

Jim this is a draft of what I would like to send to all the architects. Please comment. I plan to send it out by Wednesday (tomorrow). Thanks

RLL 15-OCT-74 13:57 24222

To be sent message for review: help from the KWAC.

In order to better know the subjective views of OFFICE-1 users, I am asking you to help me by making notes whenever the system is not performing as you would like (and when it is performing). The time, date, and load average would help, but any info would be welcomed. 1

The superwatch graphs are averaged over a 15 minute period and seem not to indicate recent high averages. 1a

You might also pass along any such complaints from others in your respective groups (include time, date, load ave.). 1b

This is only a small (unscientific) "look see" in to what is happening. I will keep you posted on any results that may come out of it. Thanks Robert 1c

A copy to feedback would be really neat. 1d

RLL 15-OCT-74 13:57 24222

To be sent message for review: help from the KWAC.

(J24222) 15-OCT-74 13:57;;;; Title: Author(s): Robert N. Lieberman/RLL; Distribution: /JCN([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: RLL;

KEV 15-OCT-74 14:46 24223

Ants-Elf comparison study

shirley, 1

we can't find our copy of the ANTS-ELF study. I suggest you get in touch with Lt. William Carlson at ARPA (Carlson@ISI) directly and ask him for a copy. 1a

KEV 15-OCT-74 14:46 24223

Ants-Elf comparison study

(J24223) 15-OCT-74 14:46;;;; Title: Author(s): Kenneth E. (Ken) Victor/KEV; Distribution: /SWW([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: KEV;

JMB 15-OCT-74 15:27 24224

On anthropomorphisms

Re item <jjournal, 24190,>

JMB 15-OCT-74 15:27 24224

On anthropomorphisms

I use anthropomorphisms like Dirk's example
-- "After (to follow), NLS expects you to..." -- or
-- "TYPEIN wants the name of the file..." --
rather than
-- "Upon seeing (to follow) print out, you are expected to..." -- or
-- "In place of TYPEIN, type the name of the file" --
in constrained situations where long and/or awkward phrases would be tedious for the reader, like in the Help database. Help is the only place I remember using such expressions, where we need to reduce awkward, impersonal, and seemingly technical descriptions as much as possible. 1

A "short and sweet" style seems called for, in my opinion, in these cases. Recommendations of TERSE, friendly, and CLEARER substitutes for these usages, in context, would prompt me to change the expression where I've used it. 2

JMB 15-OCT-74 15:27 24224

On anthropomorphisms

(J24224) 15-OCT-74 15:27;;;; Title: Author(s): Jeanne M. Beck/JMB;
Distribution: /DIRT([ACTION]) ; Sub-Collections: SRI-ARC DIRT;
Clerk: JMB;

KIRK 15-OCT-74 17:24 24225

Jump File return across horizontal split makes 'fst entry nonexistant'

I had done several cross-file edits and was trying to update the file in the bottom window but got "fst entry nonexistant" message. When I deleted the bottom window and reloaded the file, the update command worked ok.

1

KIRK 15-OCT-74 17:24 24225

Jump File return across horizontal split makes 'fst entry nonexistant'

(J24225) 15-OCT-74 17:24;;;; Title: Author(s): Kirk E. Kelley/KIRK;
Distribution: /BGS([INFO-ONLY]) CHI([INFO-ONLY]) ;
Sub-Collections: SRI-ARC; Clerk: KIRK;

KIRK 15-OCT-74 20:04 24226

Immediate Proposal for handling user-programs in NLS-8

To get everything done as soon as possible, NLS development has already started doing what is listed here. However, your opinions sent to me before next week are solicited and will be considered.

KIRK 15-OCT-74 20:04 24226

Immediate Proposal for handling user-programs in NLS-8

The primilimary proposal <hjournal, 24085,> for converting most NLS-7 userprograms as commands integrated into NLS-8 was accepted by development but had to be postponed until after the front-end, back-end split due to space limitations. It was decided that for now, accurate documentation and keeping the Calculator subsystem as a part of NLS was more important. Instead, the following list of NLS-7 userprograms will be converted into NLS-8 userprograms unless otherwise noted. In the near future, a seperate list of guaranteed procedures will be published that we promise will continue to work as advertised. Beyond that, the userprogrammer is on his own. 1

Suggestions for handling user-programs listed by Applications <hjournal, 23986,>. Codes in parens: Difficulty (a, b, c), Priority (1, 2, 3). 2

LETTER (works now, no change) 2a

FORMAT, DELDIR, and SHOWDIR (work now, no change) 2b

part of FORMAT user-subsystem 2b1

JFORM3: (a1) works now, 2c

also to be integrated as the default journal format 2c1

DELSP (a1) 2d

ADDRESS (b1) works now 2e

MESSAGE (b1) works now 2f

INSEQH (being re-implemented by Kirk as the BASE command) 2g

"COPY" "SEQUENTIAL" 2g1

<"file from"> source _ LSEL(#"OLDFILELINK") 2g1a

<" to follow"> dest _ DSEL(#"STATEMENT") 2g1b

level _ LEVADJ 2g1c

<"using"> 2g1d

("ONE" <"<CR> ends statement"> 2g1d1

/ "TWO" <"<CR>s ends statement"> ["JUSTIFIED"] 2g1d2

/ "ASSEMBLER") 2g1d3

SENDMES (b2) 2h

APPEND (c2) 2i

ADDTEXT (c2) 2j

TOC(b1), INDEX(c1), MAKEREF(c2), WORDCOUNT(a3): 2k

To be implemented as part of a new PUBLISH usersubsystem 2k1

Generate 2k1a

Table (of Contents) 2k1a1

Index 2k1a2

References 2k1a3

Count Words 2k1b

Delete (leading spaces) 2k1c

SORTNOCASE, sortrev, sortalphabetic: (b2) 2l

SUBFTPM, (ftpmsys), Load Remote (file) To be a usersubsystem. 2m

DELNAME (a3) 2n

SUBLIST (b3) 2o

NOTABS (c3) (doesn't work right in nls-7) 2p

Catalog Programs (Journal and Xdoc) being de-bugged by DSM and JCP 3

The IDENTIFICATION user-subsystem (being redone by KJM) 4

NIC programs to be located in directory <NICPROG> 5

HOSTS and Formating tables for Arpanet Directory (done by DSM) 5a

MEMLYST (memlist, memlistnew,) 5b

NICSITES 5c

NICLIST 5d

CHECKQ 5e

NIC command (done as QUERY userprogram by dsm) 5f

Immediate Proposal for handling user-programs in NLS-8

NON L-10 programs 5g

doug.sav TENEX conferencing program (Jim Calvin, Victor?) appears to be implemented as TALK (as of this statement, not available at ARC) Check with Calvin? at BBN 5g1

<kudlick>nic. runs as HELP both here and at office-1. Gets it's information from TXT files in directory NIC (most of which appear to be archived here and at Office-1). 5g2

currently available userprograms NLS-8 ARC 6

address 6a

calculator 6b

identification 6c

jform3 6d

letter 6e

message 6f

mouse 6g

sortalphabetic 6h

sortnocase 6i

sortrev 6j

inseqh is part of running system 6k

letter 6l

no change to subsystem 6l1

format, deldir, and showdir done as part of FORMAT user-program 6m

References:
Applications' list <hjournal, 23986,>
RWW's proposal <hjournal, 23992,> and <hjournal, 23999,>
Kirk's Preliminary proposal's <jjournal, 24085,>. 7

KIRK 15-OCT-74 20:04 24226

Immediate Proposal for handling user-programs in NLS-8

(J24226) 15-OCT-74 20:04;;;; Title: Author(s): Kirk E. Kelley/KIRK; Distribution: /SRI-ARC([INFO-ONLY]) FDBK([INFO-ONLY]) JHB([INFO-ONLY] you may want to forward this to KWAC) ; Sub-Collections: SRI-ARC; Clerk: KIRK;

test

this idea is a test. 1

test

(J24227) 16-OCT-74 08:47;;;; Title: Author(s): ADRIAN C. MCGINNIS/ACM; Distribution: /ACM([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: ACM;

DVN 16-OCT-74 09:10 24228

mes watson,message,txt; 1

(J24228) 16-OCT-74 09:10;;;; Title: Author(s): Dirk H. Van Nouhuys/DVN; Sub-Collections: SRI-ARC; Clerk: DVN;

JEW 16-OCT-74 10:06 24229

Reality Revisited

No matter how you slice it, system response is bad because too many users are on the system; it makes no difference whether they're running in allocated slots or offquota. The new offquota algorithm, which forces the number of offquota users to zero when things get bad, will only reduce the number of logged in users (and hence the load average) when one or more allocated slots are vacant; at any other time, the system will allow in more users than we know by experience it can satisfactorily service. Assuming we can't buy ourselves more core, as we did for OFFICE-1, the only way to EVER give ourselves a responsive system is to reduce the maximum number of slots to a value which our system can support, even if that necessitates moving some of us and/or some of our work off-line. When will we ever learn. 1

JEW 16-OCT-74 10:06 24229

Reality Revisited

(J24229) 16-OCT-74 10:06;;;; Title: Author(s): James E. (Jim) White/JEW; Distribution: /SRI-ARC([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: JEW;

RLL 16-OCT-74 12:13 24230

Subjective help in isolating Office-1 problems

In order to better know the subjective views of OFFICE-1 users, I am asking you to help me by making notes whenever the system is not performing as you would like (and when it is performing,[that would be nice]). The time, date, and load average would help, but any info would be welcomed. 1

The superwatch graphs are averaged over a 15 minute period and seem not to indicate recent high averages. 1a

You might also pass along any such complaints from others in your respective groups (include time, date, load ave.). 1b

This is only a small (unscientific) "look see" in to what is happening. I will keep you posted on any results that may come out of it. Thanks Robert (RLL) or (LIEBERMAN@SRI-ARC) 1c

A copy to feedback would be really neat. 1d

RLL 16-OCT-74 12:13 24230

Subjective help in isolating Office-1 problems

(J24230) 16-OCT-74 12:13;;;; Title: Author(s): Robert N. Lieberman/RLL; Distribution: /KWAC([ACTION]) JCN([INFO-ONLY]) JHB([INFO-ONLY]) DCE([INFO-ONLY]) ; Sub-Collections: SRI-ARC KWAC; Clerk: RLL;

RLL 16-OCT-74 12:18 24231

Reqquest to be on review team for documentation

This is to formally ask you (JCN) if I can be on the team that reviews documentation before final approval. A mechanism for this should be established. Copy goes to you and you redistribute it , or someone else (JHB?), or RWW sends it to the team, or documentators (with RWW) approval send it to us, etc. 1

RLL 16-OCT-74 12:18 24231

Reqquest to be on review team for documentation

(J24231) 16-OCT-74 12:18;;;; Title: Author(s): Robert N. Lieberman/RLL; Distribution: /JCN([ACTION]) JHB([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: RLL;

RLL 16-OCT-74 12:27 24232

sug: Needed for an OK in the send the mail command (in Sendmail).

I find that I like to do a show status command in sendmail before sending the item, however, if I forget the space and continue to type without looking (SH <CA>) I have found out that I have sent the mail. My recommendation is to have an additional <CA> (as a OK) needed to send the mail.(This is one of the few times I am in favor of aditional strokes.)

1

RLL 16-OCT-74 12:27 24232

sug: Needed for an OK in the send the mail command (in Sendmail).

(J24232) 16-OCT-74 12:27;;;; Title: Author(s): Robert N. Lieberman/RLL; Distribution: /FDBK([ACTION]) JHB([ACTION]) JDH([INFO-ONLY]) JCN([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: RLL;

DVN 16-OCT-74 14:54 24233

Primer, DCA Internetting Study Drafts, Font Test Tape to DDSI

On Thursday evening I moved a corrected draft of the DCA networking paper (documentation,dcapreface,) to ISI as file <com,(dvn)dcapreface,com;4> and a revised versin of the NLS-8 primer as file <com,(dvn)primer,com;2>. A combination of the ISI machine being down, their tape drives being attached to he other machine, our crashing, their tape drives being flakey, and operators at ISI who could no find our tapes, prevented the file from moving from ISI online to tape until late saturday afternoon. It went on tape 0002 I called DDSI who picked it up some time over the weekend. They reported they mailed te copy flow proofs to us Monday afternoon. As of Wednesday afternoon they had not arrived. 1

On Monday morning Floy Dosier of DDSI called me. He had recieved a copy of Duane Stone's memo talking abou changing tabs in the JOVIAL Manul and was anxious about whether it ment they were expect to do any reprogramming. I assured him it did not. He said he tought he had the stick font's fixed but wanted a tape from us with stick fonts to test it on...they had returned all the tapes to ISI. 2

Tuesday morning I put a COM version of <journal,12214,> on tape at ISI and told DDSI to pick it up. That file exercises all typefaces. It went down as <fontest,;1> at ISI, and came off the machine on tape 0004. 3

DVN 16-OCT-74 14:54 24233

Primer, DCA Internetting Study Drafts, Font Test Tape to DDSI

(J24233) 16-OCT-74 14:54;;;; Title: Author(s): Dirk H. Van Nouhuys/DVN; Distribution: /SRL([INFO-ONLY]) EKM([INFO-ONLY]) &DPCS([INFO-ONLY]) NDM([INFO-ONLY]) JOAN([INFO-ONLY] please put this in the DPCS notebook) ; Sub-Collections: SRI-ARC DPCS; Clerk: DVN;

DVN 16-OCT-74 16:09 24234

Minutes of Documentation Meeting of October 7; Command Summary, Userguides, Help and Syntax, Proofing

Attendees: Kirk Kelley, Jeanne Beck, Ann Weinberg 1

The Command Summary 2

Since the command language was supposed to be in its frozen form last Friday, I had asked Jeanne Beck to generate a new command summary from the syntax subsystem and do appropriate editing to make a presentable document. Making the Command Summary brought to light a certain number of small bugs, e.g. TNLS Commands designed to operate on a bug mark. Jeanne brought the bugs to the attention of appropriate programmers. When the bugs are fixed she or some one can run the system again. Further work has been delayed by the Monday-Tuesday crash. 2a

This work was completed 10/12 2a1

Userguide Shelves 3

We delivered responsibility for maintenance of the user guide shelves to Ann Weinberg. Jeanne and Ann, and later Ann and I, went over the status of various documents. Ann is drafting an annotated list of all ARC Documentation. The DRAFT is (weinberg,doclist,) 3a

Help Syntax And Functions Statements 4

The node in the help data base that describes each command must give an account of its function and its syntax. The syntax of some commands has appeared on the first line of the statement describing its function and of others it has resided in a branch headed by the name "syntax" and the object of links in substatements of the function statement. 4a

For one thing, it is at present very difficult to set up a proper search for information about the Syntax command because the top level branch pre-empts the name. 4b

More important, we all agree that the Help Data Base should shortly change and cease to store syntax as written information. Instead it should contain link-like constructs that would call the syntax as appropriate from the syntax generator. 4c

At this meeting we agreed that the syntax generator should deliver to the first line of the function statement and that Kirk should prepare for this by moving the information now in the branch syntax into the function statement, incidentally freeing the name syntax for an account of the command. 4d

Minutes of Documentation Meeting of October 7; Command Summary, Userguides, Help and Syntax, Proofing

This work was completed 10/13 4d1

Examples 5

We reviewed the possibilities in using process command branch for interacting examples. We agreed more concrete planning should wait. 5a

Content and Copy Proofing 6

We agreed that any online files intended for people outside ARC should go thru the following steps: 6a

When the author (or author of a revision) is satisfied that it is complete and accurate she should give it to some other documentation person for content and copy proofing. If more than trivial changes result, some third person should read for copy errors before it leaves ARC. In the case of documents to be provided for Applications for their use, e.g. userguides, the content- and copy-proofed document should go to Dick Watson before going to anyone in applications. 6a1

Finally and in addition to all of the above, whenever hard copy is to go to some one outside of ARC, a third person should read that very hard copy for copy errors specifically considering the problem of printer errors. 6b

We must point out however that passing copy-proofed drafts to people who are likely to make suggestions about content or format can be frustrating and time consuming. If Dick Watson or Jim Norton make substantial suggestions, we are back to the starting point in this cycle. 6c

Certain policies will reduce the time consumed: 6d

If documentors and reviewers agree thoroughly on form and content before the proofing stage, 6d1

If reviewers do not feel obliged to make suggestions, 6d2

If reviewer solicit drafts before the proofing. 6d3

DVN 16-OCT-74 16:09 24234

Minutes of Documentation Meeting of October 7; Command Summary, Userguides, Help and Syntax, Proofing

(J24234) 16-OCT-74 16:09;;;; Title: Author(s): Dirk H. Van Nouhuys/DVN; Distribution: /JOAN([ACTION] please add this to the dirt notebook) DIRT([INFO-ONLY]) ; sub-Collections: SRI-ARC DIRT; Clerk: DVN; Origin: < HAMILTON, MINUTESOFDOCUMEMTATIONMEETING.NLS;2, >, 16-OCT-74 16:04 DVN ;;;;####;

JAKE 17-OCT-74 11:41 24235

Anthropowhatchamacallits

In answer to JHB and DVN, I would prefer not to call a spade an anthropomorphism. Brevity and clarity are the essence of a written procedure - espcially if it is online on a slow tty. Therefore, I would opt for:

"After (blap) prints out, type..." instead of
"Upon seeing (blap) print out, you are expected to..." or
"After (blap), NLS expects you to ..." and
"After TYPEIN, enter a filename," instead o
"In place of TYPEIN, type the name of the file" or
"In place of TYPEIN, type the name of the file"

My defense of these is they are clear and shorter - nothing literary. 1

JAKE 17-OCT-74 11:41 24235

Anthropowhatchamacallits

(J24235) 17-OCT-74 11:41;;;; Title: Author(s): Elizabeth J. (Jake) Feinler/JAKE; Distribution: /DIRT([INFO-ONLY]) ; Sub-Collections: SRI-ARC DIRT; Clerk: JAKE;

JAKE 17-OCT-74 11:47 24236

Sendmail now works as advertised.

Thanks feedback (Dave H.?) for fixing my sendmail file. Now seems to be working fine. (How about that - no gripe this time!!) Jake 1

JAKE 17-OCT-74 11:47 24236

Sendmail now works as advertised.

(J24236) 17-OCT-74 11:47;;;; Title: Author(s): Elizabeth J. (Jake) Feinler/JAKE; Distribution: /FDBK([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: JAKE;

DVN 17-OCT-74 12:53 24237

Proposal Posibility: Output Processorr Direct to XGP [To add this item to DPCS subcollection]

See <mjournal,24134,>

1

DVN 17-OCT-74 12:53 24237

Proposal Posibility: Output Processorr Direct to XGP [To add this item to DPCS subcollection]

(J24237) 17-OCT-74 12:53;;;; Title: Author(s): Dirk H. Van Nouhuys/DVN; Distribution: /JOAN([INFO-ONLY] please add <mjournal,24132,> to the dpcs notebook) DCE([INFO-ONLY] Please don't forget the DPCS subcollection) ; Sub-Collections: DPCS SRI-ARC; Clerk: DVN;