Jovial Equations - Corrected Version

**********EQUATIONS**********

Appendix A

SYNTAX EQUATIONS

The following pages contain the complete syntactic description of JOVIAL (J73). The metalinguistic equations are in alphabetical order of the metalinguistic terms being defined. In general, each defining equation is individually boxed. The boxes are numbered sequentially in the upper left hand corner by a number in italics followed by a colon. Following the colon is a list of the box numbers in which the current term is used as a part of the definitions of other terms. The metalinguistic symbology is explained in section 1.4.

Equations 94 and 95 are in one box. These are both valid and necessary definitions for format:list. Equation 144 is the definition for mark. In the same box, opposite each mark is a metalinguistic term (or two). These marks constitute the definitions of these terms. Equation 172 defines pattern:digit. In the same box is information giving the bit pattern corresponding to each pattern:digit, depending on the order of the pattern:constant. In the box with equation 190, the definition of relational:operator, is a list of the meanings of the relational:operators. Box 234 contains a definition for system:dependent:character, but the definition is a prose description; a metalinguisitic equation is not feasible. Equations 247 and 248 are in one box. They are both valid and necessary definitions for variable.

1: 233

abbreviation ::= letter

2: 63

abnormal:directive ::= !ABNORMAL data:name ;

3: 130

absolute:function:call ::= ABS (numeric:formula)

4: 58, 170

actual:define:parameter ::=
definition
" definition "

5: 101, 170, 180, 191

STOP
RETURN
TEST
alternate:entrance:name
procedure:name
control:variable
EXIT statement:name

actual:input:parameter ::=
statement:name
procedure:name
formula
table:name
data:block:name
variable
@ pointer:formula

6: 170, 180, 191

actual:output:parameter ::= variable

7: 166, 217

allocation:increment ::= number

8: 9, 49, 166, 182, 205, 217

allocation:specifier ::= @ pointer:formula

9: 75, 184

alternate:entrance:declaration ::=

ENTER alternate:entrance:name

(formal:input:parameter

: formal:output:parameter)

environmental:specifier item:description
allocation:specifier

packing:specifier [bit:number]

= + constant ;
-

10: 130

alternate:entrance:function:call ::= ALT (
procedure:name)

11: 5, 9, 53, 101, 122, 138, 180, 187, 193

alternate:entrance:name ::= name

12: 159

```
        +
        -
        *
arithmetic:operator ::=
        /
        \
        **
```

13:

```
assignment:operator ::=  =
```

14: 207

```
assignment:statement ::=
                formula
        variable                indexed:variable:range ;
        indexed:variable:range =
                format:function:call

                formula
    format:variable     =               ;
                indexed:variable:range
```

15: 159

```
attribute:association ::= @@ [ description:attibute ]
```

16: 63

begin:directive ::= !BEGIN reference ;

17: 18

bit:form ::= form

18: 18, 29, 97, 159, 196

pattern:constant
entry:variable
comparison
chain:comparison
bit:string:function:call
shift:function:call
bit:formula ::= bit:form
bit:formula logical:operator bit:formula
NOT bit:formula
bit:formula & bit:formula
(bit:formula)
numeric:formula
character:formula

19: 9, 182, 205, 217, 218

bit:number ::= number

20: 18, 130

bit:string:function:call ::=

BIT (formula , numeric:formula ,
numeric:formula)

21: 247

bit:variable ::=

entry:variable
BIT (named:variable , numeric:formula ,
numeric:formula)

22: 217

bits:per:entry ::= number

23: 130

byte:string:function:call ::=

BYTE (character:formula , numeric:formula ,
numeric:formula)

24: 18

chain:comparison ::= comparison relation:operator
formula

25: 26, 32, 46, 62, 100, 112, 120, 137, 213, 234, 240

sign
character ::=
system:dependent:character

26: 29, 39

character:constant ::= count ' character '

27: 29

character:form ::= form

28: 92

character:format ::= count C

29: 18, 23, 29, 93, 94, 97

character:formula ::= character:constant
| character:variable
| character:form
| character:function:call
| character:formula & character:formula
| (character:formula)
| bit:formula

30: 29

character:function:call ::= function:call

31: 29, 96, 247

```
         named:character:variable
character:variable  ::=        BYTE  (
named:character:variable  ,  numeric:formula

                 ,  numeric:formula    )
```

32: 233

```
comment  ::=  "  character       "
```

33: 18, 24

```
comparison  ::=  formula  relational:operator  formula
```

34: 63

```
compool:directive  ::=
                       name
     compool:name
   !COMPOOL            (  name  )      ;

                  (  compool:name  )
```

35: 34

```
compool:name  ::=  name
```

36: 219

```
        statement
compound:statement  ::=  BEGIN                END    ;
        declaration
```

37: 38, 97, 238, 241

```
conditional:formula  ::=  formula
```

38: 207

```
conditional:statement  ::=

   IF  conditional:formula  ;  controlled:statement

            statement:name  :      ELSE
        controlled:statement
```

39: 9, 42, 182, 205, 233

```
        numeric:constant
constant  ::= pattern:constant
        character:constant
```

40: 97

```
constant:formula  ::=  (  formula  )
```

```
41:    166, 167, 217, 218

   constant:list  ::=

          [  index  ]
       [  index  ]    constant:list:element
       constant:list:element
                 ,



42:    41, 42

               ,     +     constant      ,
   constant:list:element  ::=                  -
          count  (  constant:list:element  )
          constant:list:element



43:    136, 148

          increment:phrase          terminator:phrase
          replacement:phrase
   control:clause  ::=        initial:phrase
                  increment:phrase
          terminator:phrase          replacement:phrase



44:    5, 239

          named:variable
   control:variable  ::=
          letter:control:variable



45:    38, 141

   controlled:statement  ::=  statement
```

46: 63

copy:directive ::= !COPY character ;

47: 26, 28, 42, 72, 78, 95, 98, 103, 120, 124, 126, 171, 173, 199

count ::= number

48: 182

data:allocator:specifier ::= @

49: 49, 51, 75

data:block:declaration ::=

environmental:specifier
BLOCK data:block:name ;
allocation:specifier

simple:item:declaration
table:declaration
BEGIN END ;
data:block:declaration
independent:overlay:declaration

50: 5, 49, 52, 90, 109, 138, 209

data:block:name ::= name

```
51:    54

          item:declaration
   data:declaration  ::= table:declaration
          data:block:declaration
          overlay:declaration



52:    2, 129, 174, 195, 244

          item:name
   data:name  ::=         table:name
          data:block:name



53:    130

          procedure:name
   data:size:function:call  ::=  DSIZE  (                )
          alternate:entrance:name



54:    36, 54, 112, 179

          status:list:declaration
          form:declaration
          data:declaration
          null:declaration
   declaration  ::=      define:declaration
          name:declaration
          processing:declaration
          external:declaration
          BEGIN declaration      END    ;



55:    54

   define:declaration  ::=
```

```
DEFINE define:name    (  formal:define:parameter    )
"  definition  ";
```

```
56:    55, 58

   define:name  ::=  name
```

```
57:    4, 55

   definition  ::=  sign
```

```
58:

   definition:invocation  ::=  define:name    (
   actual:define:parameter      )
```

```
59:    185

   dependent:program:declaration  ::=  procedure:declaration
```

```
60:    15, 69

         item:name
   description:attribute  ::=
         item:description
```

(equ61)

61: 166, 217

```
dimension:list ::= [     lower:bound : upper:bound
]
```

62: 207

```
direct:statement ::= DIRECT character JOVIAL ;
```

63:

```
                compool:directive
                skip:directive
                begin:directive
                end:directive
                trace:directive
                copy:directive
                abnormal:directive
                sets:directive
        directive ::=       uses:directive
                pointer:directive
                order:directive
                recursive:directive
                time:directive
                space:directive
                linkage:directive
                interference:directive
                frequency:directive
```

64: 233

```
        !COMPOOL
        !SKIP
        !BEGIN
        !END
        !TRACE
        !COPY
        !ABNORMAL
        !SETS
directive:key ::=   !USES
        !POINTER
        !ORDER
        !RECURSIVE
        !TIME
        !SPACE
        !LINKAGE
        !INTERFERENCE
        !FREQUENCY
```

65: 63

```
end:directive ::= !END ;
```

66: 217

```
entries:per:word ::= number
```

67: 18, 21, 117, 252

```
entry:variable ::= table:name [ index ] @
pointer:formula
```

68: 9, 49, 166, 182, 205, 217

```
                   program:name
         IN        procedure:name
environmental:specifier  ::=            RESERVE

         RESERVE
```

69: 159

```
evaluation:control  ::=  @  [  description:attribute  ]
```

70: 207

```
exchange:statement  ::=  variable  ==  variable  ;
```

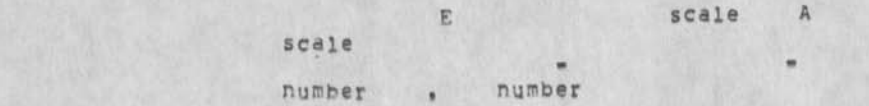
71: 207

```
exit:statement  ::=  EXIT  statement:name  ;
```

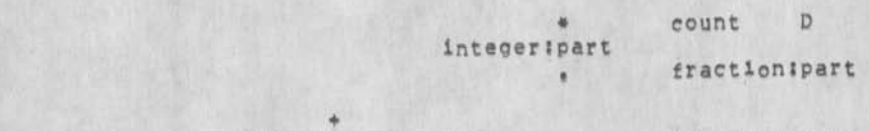
72: 82

```
                   count    D       count    Z

          +
exrad     ::=               count    Z       count    D
          -

                   count    Z       *
```

```
73:    130
   exrad:function:call  ::=  XRAD  (  numeric:formula  )


74:    132
   exrad:specifier  ::=  number


75:    54
   external:declaration  ::=
                simple:item:declaration
                table:declaration
                data:block:declaration
                name:declaration
      DEF                procedure:declaration
                alternate:entrance:declaration
      REF                simple:item:declaration
                table:declaration
         BEGIN  data:block:declaration  END    ;
                name:declaration
                procedure:declaration
                alternate:entrance:declaration


76:    87
   field:width  ::=  number


77:    157
   fixed:constant  ::=
      number    .
                          +              +
```



78: 158



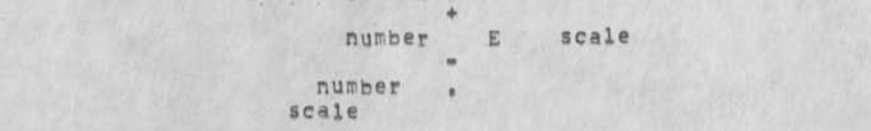
79: 160

```
fixed:function:call  ::=  function:call
```

80: 162

```
fixed:variable  ::=  named:variable
```

81: 157



```
                    +
          E                  scale
                    -
    number   .   number
```

82: 158

floating:format ::= significand E exrad R

83: 160

floating:function:call ::= function:call

84: 162

floating:variable ::= named:variable

85: 141

for:clause ::= FOR loop:control ;

86: 17, 27

form ::= form:name (formula)

87: 54

B

```
form:declaration  ::=  FORM  form:name
field:width       ;
     C
```

88: 86, 87

```
form:name  ::=  name
```

89: 55, 170

```
formal:define:parameter  ::=  letter
```

90: 9, 170, 182

```
        statement:name
        simple:item:name
formal:input:parameter  ::=  procedure:name
        table:name
        data:block:name
```

91: 9, 170, 182

```
formal:output:parameter  ::=  simple:item:name
```

92: 95

```
        null:format
        insert:format
format  ::=  skip:format
        character:format
        pattern:format
        numeric:format
```

```
93:    14, 130

   format:function:call  ::=

      FORMAT  (  character:formula     ,      format:list
      ,  procedure:name       )



94:    93, 95, 96

   format:list  ::=  character:formula



95:    93, 95, 96

         format
   format:list  ::=
         count  (  format:list  )



96:    14, 102, 247

   format:variable  ::=

      FORMAT  (  character:variable     ,     format:list
      ,  procedure:name       )
```

```
97:    5, 14, 20, 24, 33, 37, 40, 86, 119, 159, 192, 203,
209, 242, 245

        pointer:formula
        numeric:formula
        bit:formula
   formula  ::=  conditional:formula
        character:formula
        value:formula
        numeric:value:formula
        constant:formula


98:    78

        count    D          count    Z

   fraction:part  ::=

        count    D


99:    130

   fraction:part:function:call  ::=  FRAC  (  numeric:formula
   )


100:   63

   frequency:directive  ::=  !FREQUENCY  character      ;
```

101: 30, 79, 83, 125

intrinsic:function:call

procedure:name
function:call ::= @ pointer:formula
alternate:entrance:name

(actual:input parameter)

102: 248

format:variable

BYTE (named:character:variable ,
numeric:formula

functional:variable ::= , numeric:formula
)

BIT (named:variable , numeric:formula

, numeric:formula)

103: 158

generalized:numeric:formula ::= count N R

104: 207

go:to:statement ::= GOTO statement:name [
index] ;

105: 115

high:point ::= numeric:formula

106: 233

```
        +
        -
        *
        /
        **
        \
        &
        =
        ==
        <
        >
        <=
        >=
ideogram ::= <>
        .
        ,
        :
        ;
        !
        "
        '
        (
        )
        [
        ]
        @
        @@
```

107: 43

```
      numeric:formula
increment:phrase ::= BY
      numeric:value:formula
```

```
108:   49, 168

   independent:overlay:declaration  ::=

          [ number ]
      OVERLAY              independent:overlay:expression  ;
          [ pattern:constant ]



109:   111

          spacer
          simple:item:name
   independent:overlay:element  ::=      table:name
          data:block:name
          ( independent:overlay:expression )



110:   108, 109

   independent:overlay:expression  ::=

      independent:overlay:string    :
      independent:overlay:string



111:   110

   independent:overlay:string  ::=
   independent:overlay:element



112:   185

   independent:program:declaration  ::=
          statement
      PROGRAM program:name    ( character    )    ;
          declaration
```

113: 41, 67, 104, 237

```
index ::= index:component
```

114: 113, 116

```
index:component ::= numeric:formula
```

115: 116

```
index:component:range ::= low:point : high:point
```

116: 118, 155

```
index:range ::=
      index:component:range
      index:component
```

117: 150

```
indexed:variable ::=
      table:variable
      entry:variable
```

118: 14

```
indexed:variable:range ::=

        item:name
                          [ index:range ]      @
      pointer:formula
        table:name

            item:name
      ALL (                    @ pointer:formula    )
            table:name
```

119: 43

```
initial:phrase ::= formula
```

120: 92

```
      count    S

               numeral
insert:format  ::=   count    /
               letter

      count    "  character      "
```

121: 182

```
instruction:allocation:specifier ::= pointer:formula
```

122: 130

```
      procedure:name
instruction:size:function:call  ::=  ISIZE  (          )
      alternate:entrance:name
```

123: 157, 175

```
integer:constant  ::=  number
```

124: 158

```
                count    Z          count    D
      +
integer:format  ::=                   R
      -
                count    D          count    Z
```

125: 160

```
integer:function:call  ::=  function:call
```

126: 78

```
      count    Z          count    D
integer:part  ::=
      count    D
```

```
127:  130

   integer:part:function:call  ::=  INT  (  numeric:formula )



128:  162

         named:variable
   integer:variable  ::=
         letter:control:variable



129:  63

   interference:directive  ::=  !INTERFERENCE    data:name  ;
   data:name            ;



130:  101

         format:function:call
         byte:string:function:call
         bit:string:function:call
         alternate:entrance:function:call
         number:of:entries:function:call
         location:function:call
         shift:function:call
         absolute:function:call
         words:per:entry:function:call
   intrinsic:function:call  ::=  exrad:function:call
         significand:function:call
         signed:function:call
         signum:function:call
         size:function:call
         type:function:call
         fraction:part:function:call
         integer:part:function:call
         instruction:size:function:call
         data:size:function:call
```

```
131:   51
            simple:item:declaration
    item:declaration  ::= ordinary:table:item:declaration
            specified:table:item:declaration



132:   9, 60, 166, 167, 182, 205, 217, 218

    item:description  ::=

        C     size:specifier

        F  ,  R        significand:specifier      ,
        exrad:specifier

                     status:list
        S            status:list:name
           , R     size:specifier          +
        U            ,                precision:specifier
                                -


133:   52, 60, 118, 167, 187, 205, 218, 229, 237,

    item:name  ::=  name
```

134: 1, 89, 120, 135, 136, 145, 197, 221

letter ::= A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z

135: 44, 128, 233, 248

letter:control:variable ::= letter

136: 140

letter:loop:control ::= letter (control:clause)

```
137:  63
   linkage:directive  ::=  !LINKAGE    character      ;


138:  130
         statement:name
         named:variable
   location:function:call  ::=  LOC  (   table:name      )
         data:block:name
         procedure:name
         alternate:entrance:name


139:  18
         AND
         OR
   logical:operator  ::=
         EQV
         XOR


140:  85
         named:loop:control
   loop:control  ::=
         letter:loop:control


141:  207
   loop:statement  ::=  for:clause  contolled:statement
```

142: 115

low:point ::= numeric:formula

143: 61

number
lower:bound ::=
simple:item:name

144: 197

	+	plus:sign
	-	minus:sign
	*	asterisk
	/	slash
	\	back:slash
	&	ampersand
	>	greater:than:sign
	<	less:than:sign
	=	equals:sign
	@	at:sign
mark	::=	. decimal:point
	:	colon
	,	comma
	;	semicolon
		space
	(	left:parenthesis, parenthesis
	)	right:parenthesis, parenthesis
	[	left:bracket, bracket
	]	right:bracket, bracket
	'	prime
	"	quotation:mark
	$	dollar:sign
	!	exclamation:point

145: 11, 34, 35, 50, 56, 88, 133, 183, 186, 206, 220, 221, 225, 233, 236, 241

```
                letter
        letter  numeral
name    ::=
          $       $
                  #
```

146: 54, 75

```
        statement:name
name:declaration  ::=  NAME          ;
        procedure:name
```

147: 31, 102

```
named:character:variable  ::=  named:variable
```

148: 140

```
named:loop:control  ::=  named:variable  (  control:clause
)
```

149: 219

```
named:statement  ::=  statement:name  !  statement
```

150: 21, 44, 80, 84, 102, 128, 138, 147, 148

simple:variable
named:variable ::=
indexed:variable

151: 54, 164, 215

NULL ;
null:declaration ::=
BEGIN END ;

152: 92

null:format ::=

153: 219

NULL ;
null:declaration ::=
BEGIN END ;

154: 7, 19, 22, 47, 66, 74, 76, 77, 81, 108, 123, 143, 169, 177, 189, 194, 201, 210, 214, 223, 232, 233, 243, 249, 250

number ::= numeral

155: 130

number:of:entries:function:call ::= NENT (table:name
[index:range])

156: 120, 145, 154, 197

0
1
2
3
numeral ::= 4
5
6
7
8
9

157: 39, 159

integer:constant
fixed:constant
numeric:constant ::= floating:constant
status:constant
qualified:status:constant

158: 92

generalized:numeric:format
integer:format
numeric:format ::=
fixed:format
floating:format

159: 3, 18, 20, 21, 23, 31, 73, 97, 99, 102, 105, 107, 114, 127, 142, 159, 161, 175, 196, 198, 200, 202, 232

numeric:constant
numeric:variable
numeric:function:call
+
numeric:formula
-

numeric:formula ::= numeric:formula arithmetic:operator

evaluation:control numeric:formula
evaluation:control
formula
attribute:association
(numeric:formula)
bit:formula

160: 159

integer:function:call
numeric:function:call ::= fixed:function:call
floating:function:call

161: 97, 107

numeric:value:formula ::= [numeric:formula]

162: 159, 176, 247
integer:variable
numeric:variable ::= fixed:variable
floating:variable

```
163:  63

   order:directive  ::=  !ORDER  ;



164:  165

         null:declaration
         ordinary:table:item:declaration

   ordinary:table:body  ::=
                  ordinary:table:item:declaration
         BEGIN                END    ;
                  subordinate:overlay:declaration



165:  235

   ordinary:table:declaration  ::=  ordinary:table:heading
   ordinary:table:body



166:  165

   ordinary:table:heading  ::=
         environmental:specifier
      TABLE  table:name
         allocation:specifier

         :  allocation:increment     dimension:list

         structure:specifier         packing:specifier

         item:description         =  constant:list    ;
```

167: 131, 164

ordinary:table:item:declaration ::=

ITEM item:name item:description
Packing:specifier = constant:list ;

168: 51

independent:overlay:declaration
overlay:declaration ::=
subordinate:overlay:declaration

169: 9, 166, 167, 182, 205, 217, 218

N

packing:specifier ::= M number

D

170:

actual:define:parameter
formal:define:parameter
parameter ::= actual:input:parameter
actual:output:parameter
formal:input:parameter
formal:output:parameter

```
171:   18, 39, 108

          1
          2
   pattern:constant  ::= 3    B    count    ' pattern:digit
          4
          5


172:   171

          pattern pattern:digit   order

     0 0 0 0 0          0
     0 0 0 0 1          1              1
     0 0 0 1 0          2
     0 0 0 1 1          3              2
     0 0 1 0 0          4
     0 0 1 0 1          5
     0 0 1 1 0          6
     0 0 1 1 1          7              3
     0 1 0 0 0          8
     0 1 0 0 1          9
     0 1 0 1 0          A
     0 1 0 1 1          B
     0 1 1 0 0          C
     0 1 1 0 1          D
     0 1 1 1 0    pattern:digit ::=    E
     0 1 1 1 1          F              4
     1 0 0 0 0          G
     1 0 0 0 1          H
     1 0 0 1 0          I
     1 0 0 1 1          J
     1 0 1 0 0          K
     1 0 1 0 1          L
     1 0 1 1 0          M
     1 0 1 1 1          N
     1 1 0 0 0          O
     1 1 0 0 1          P
     1 1 0 1 0          Q
     1 1 0 1 1          R
     1 1 1 0 0          S
     1 1 1 0 1          T
     1 1 1 1 0          U
     1 1 1 1 1          V              5
```

```
173:  92

          1
          2
   pattern:format  ::=  3      B      count    P
          4
          5


174:  63

   pointer:directive  ::=  !POINTER   pointer:formula  ;
   data:name          ;


175:  5, 8, 67, 97, 101, 118, 121, 174, 180, 208, 237

         integer:constant
   pointer:formula  ::=  simple:integer:variable
         (  numeric:formula  )


176:  247

   pointer:variable  ::=  numeric:variable


177:  132

   precision:specifier  ::=  number
```

178: 221, 233

```
        ABS
        ALL
        ALT         NENT
        AND         NOT
        BEGIN       NULL
        BIT         NWDSEN
        BLOCK       OR
        BY          OVERLAY
        BYTE        PROC
        DEF         PROGRAM
        DEFINE      REF
        DIRECT      REMQUO
        DSIZE       RESERVE
        ELSE        RETURN
        END         SHIFT
primitive ::=       ENTER   SIG
        EQV         SIGNED
        EXIT        SIGNUM
        FOR         SIZE
        FORM        STATUS
        FORMAT      STOP
        FRAC        SWITCH
        GOTO        TABLE
        IF          TEST
        IN          THEN
        INT         TYPE
        ISIZE       UNTIL
        ITEM        WHILE
        JOVIAL      XOR
        LOC         XRAD
        NAME        ZAP
```

179: 181

```
        declaration
procedure:body ::=
        statement
```

```
180:  207

   procedure:call:statement  ::=

      remquo:procedure:call:statement

         procedure:name
                @ pointer:formula
         alternate:entrance:name

            (  actual:input:parameter      )

            (    actual:input:parameter      :        ;

               actual:output:parameter    )



181:  59, 75, 184

   procedure:declaration  ::=  procedure:heading
   procedure:body



182:  181

   procedure:heading  ::=

                           environmental:specifier
   PROC procedure:name
                           data:allocation:specifier

      :  instruction:allocation:specifier

      (    formal:input:parameter        :
      formal:output:parameter      )

         environmental:specifier
                                  item:description
         allocation:specifier

      packing:specifier        [  bit:number  ]

         +
```

= constant ;

183: 5, 10, 53, 68, 90, 93, 96, 101, 122, 138, 146, 180, 182, 187, 193

procedure:name ::= name

184: 54

program:declaration
processing:declaration ::= procedure:declaration
alternate:entrance:declaration

185: 184

independent:program:declaration
program:declaraton ::=
dependent:program:declaration

186: 68, 112

program:name ::= name

187: 157

status:list:name
item:name
qualified:status:constant ::= V(table:name :
status)
procedure:name
alternate:entrance:name

188: 63

recursive:directive ::= !RECURSIVE ;

189: 16, 211

reference ::= number

190: 24, 33, 246

```
        <       less than
        =       equal
        >       greater than
relational:operator ::=
        >=      greater than or equal, not less than
        <>      less than or greater than, not equal
        <=      less than or equal, not greater than
```

191: 180

remquo:procedure:call:statement ::=

REMQUO (actual:input:parameter ,
actual:input:parameter
: actual:output:parameter ,
actual:output:parameter) ;

192: 43

```
        formula
replacement:phrase ::= THEN
        value:formula
```

193: 207

```
        procedure:name
return:statement ::= RETURN ;
        alternate:entrance:name
```

194: 77, 81

```
scale ::= number
```

195: 63

```
sets:directive ::= !SETS data:name ;
```

196: 18, 130

```
shift:function:call ::= SHIFT ( bit:formula ,
numeric:formula )
```

197: 25, 57, 234

```
        letter
sign ::=    numeral
        mark
```

198: 18, 130

signed:function:call ::= SIGNED (numeric:formula)

199: 82

count D . count D
count Z
count D . count D
count D .
+
significand ::=
-
count D * count D
count * count D
count D count *

200: 130

significand:function:call ::= SIG (numeric:formula)

201: 132

significand:specifier ::= number

202: 130

signum:function:call ::= SIGNUM (numeric:formula)

203: 207

simple:assignment:statement ::= variable = formula ;

204: 175

simple:integer:variable ::= simple:variable

205: 49, 75, 131

simple:item:declaration ::=

environmental:specifier
ITEM item:name
allocation:specifier
item:description packing:specifier

[bit:number] = constant
;

206: 90, 91, 109, 143, 208, 243

simple:item:name ::= name

207: 219

```
        simple:assignment:statement
        assignment:statement
        exchange:statement
        go:to:statement
        exit:statement
        test:statement
   simple:statement  ::= return:statement
        zap:statement
        stop:statement
        loop:statement
        conditional:statement
        switch:statement
        procedure:call:statement
        direct:statement
```

208: 150, 204

```
   simple:variable  ::=  simple:item:name     @
   pointer:formula
```

209: 130

```
        formula
   size:function:call  ::=  SIZE  (              )
        data:block:name
```

210: 132

```
   size:specifier  ::=  number
```

```
211:  63
   skip:directive  ::=  !SKIP    reference    ;


212:  92
   skip:format  ::=  X


213:  63
   space:directive  ::=  !SPACE    character        ;


214:  109
   spacer  ::=  number


215:  216
        null:declaration
   specified:table:body  ::=
   specified:table:item:declaration
        BEGIN specified:table:item:declaration      END    ;


216:  235
   specified:table:declaration  ::=  specified:table:heading
   specified:table:body
```

217: 216

```
specified:table:heading ::=

      environmental:specifier
TABLE table:name
      allocation:specifier

      : allocation:increment    dimension:list

      structure:specifier

      words:per:entry
      bits:per:entry        bit:number    entries:per:word

      packingspecifier        item:description
      packing:specifier

      [ bit:number    , word:number    ]      =
      constant:list    ;
```

218: 131, 215

```
specified:table:item:declaration ::=

   ITEM item:name      item:description

      packing:specifier    [ bit:number

      , word:number     ]    = constant:list    ;
```

219: 36, 45, 112, 149, 179, 232

```
      null:statement
statement ::=        simple:statement
      compound:statement
      named:statement
```

220: 5, 38, 71, 90, 104, 138, 146, 149, 232

statement:name ::= name

221: 187, 222, 233

status ::= primitive name letter

222: 157, 223

status:constant ::= V(status)

223: 132, 224

status:list ::=
[number]+ status:constant [number]+ status:constant

224: 54

status:list:declaration ::= STATUS status:list:name status:list ;

225: 132, 187, 224

status:list:name ::= name

226: 207

stop:statement ::= STOP ;

227: 166, 217

structure:specifier ::= P | T

228: 164, 168

subordinate:overlay:declaration ::= OVERLAY
subordinate:overlay:expression ;

229: 231

subordinate:overlay:element ::= item:name | (subordinate:overlay:expression)

230: 228, 229

subordinate:overlay:expression ::=

subordinate:overlay:string :
subordinate:overlay:string

231: 230

subordinate:overlay:string ::=
subordinate:overlay:element

232: 207

switch:statement ::=

SWITCH numeric:formula ; statement:name :
+
BEGIN [number] statement
, END ;
-

233:

primitive
ideogram
name
letter:control:variable
symbol ::= abbreviation
number
constant
comment
directive:key
status

234: 25

system:dependent:character

Most computer systems can read and write more characters than are encompassed in the set of JOVIAL signs. The entire set that can be handled is known as the set of characters. The characters that are not signs are known as system:dependent:characters.

```
235:   49, 51, 75

        ordinary:table:declaration
   table:declaration  ::=
        specified:table:declaration


236:   5, 52, 67, 90, 109, 118, 138, 155, 166, 187, 217, 251,
252

   table:name  ::=  name


237:   117
   table:variable  ::=  item:name    [    index    ]        @
   pointer:formula


238:   43

         WHILE
                  conditional:formula
   terminator:phrase  ::=         UNTIL

        value:terminator


239:   207

   test:statement  ::=  TEST    control:variable    ;


240:   63

   time:directive  ::=  !TIME    character        ;
```

```
241:  63
   trace:directive  ::=  !TRACE    (  conditional:formula  )
   name    ;


242:  130
   type:function:call  ::=  TYPE  (  formula  )


243:  61
         number
   upper:bound  ::=
         simple:item:name


244:  63
   uses:directive  ::=  !USES  data:name      ;


245:  97, 192, 246
   value:formula  ::=  [  formula  ]
```

```
246:   238
   value:terminator  ::=

      WHILE       value:formula  relational:operator  variable
      UNTIL       variable  relational:operator  value:formula



247:   5, 6, 14, 70, 203, 246

         pointer:variable
         numeric:variable
   variable  ::= bit:variable
         character:variable
         format:variable



248:   5, 6, 14, 70, 203, 246

         named:variable
   variable  ::= letter:control:variable
         functional:variable



249:   217, 218

   word:number  ::=  number



250:   217

   words:per:entry  ::=  number
```

```
251:   130

   words:per:entry:function:call  ::=  NWDSEN  (  table:name
   )


252:   207

         table:name
   zap:statement  ::=  ZAP        ;
         entry:variable
```

**********TABLES**********

Column	0	1	2	3	4	5	6	7
	8	9	10	11	12	13	14	15
Code	0	1	2	3	4	5	6	7
	8	9	A	B	C	D	E	F
Row								
0		space	0	@	P		p	
1		!	1	A	Q	a	q	
2		"	2	B	R	b	r	
3		#	3	C	S	c	s	
4		$	4	D	T	d	t	
5		%	5	E	U	e	u	
6		&	6	F	V	f	v	
7		'	7	G	W	g	w	
8		(	8	H	X	h	x	
9		)	9	I	Y	i	y	
10		*	:	J	Z	j	z	
11		+	;	K	[	k		
12		,	<	L	\	l		
13		-	=	M	]	m		
14		.	>	N		n		
15		/	?	O		o		

Notes: row 0, column 3: zero
row 1, column 3: one
row 7, column 2: prime, often rendered as a vertical mark in JOVIAL
row 12, column 6: a lowercase letter
row 15, column 4: an uppercase letter

Figure 2-1. Characters

	fixed:constant	value	size	precision
19A0	19	5	0	
19A3	19	8	3	
19A-2	16	3	-2	
2,3A0	2	2	0	
2,3A-1	2	1	-1	
2,3A2	2,25	4	2	
2,3A5	2,28125	7		5
2,3A6	2,296875	8		6

Left symbol ends in	Right symbol starts with: numeral	letter	$	'	.	"
numeral		SR		SR	SR	SR
letter		SR		SR	SR	SR
$	SR		SR	SR	SR	
'	SR		SR	SR	SR	
.	SR		SR			
"						

SR

Value of the conditional:formula

	0	1
IF	Skip the controlled:statement following this conditional:formula,	Execute the following controlled:statement then skip the controlled:statement
	Execute the controlled:statement following the matching ELSE if there is one,	immediately following the matching ELSE if there is one,
UNTIL	Execute the controlled:statement,	Go on to the next control:clause
	or exit the loop if there is	

no further control:clause,

WHILE Go on to the next contol:clause Execute the controlled:statement,
or exit the loop if there
is no further control:clause,

bit:formula 10000101	110 01111100	01011100	01010111
padded 10000101	00000110 01111100	01011100	01010111
selected 10000101		01011100	01010111

x	y	x\y	x	y	x\y
7	0	undefined	-3,7	2	0,3
1	2	1	4,6	1,5	0,1
2	2	0	-0,1	1,5	1,4
3	2	1	1	-2	-1
-3	2	1	3,7	-2	-0,3
3,7	2	1,7	-3,7	-2	-1,7

First or Only Character	Meaning
I	Number of integer bits
A	Number of fraction bits (bits after the point).
Z	Maximum size (I+A) the system ordinarily allows for fixed and integer arithemetic.
Y	An even larger maximum size (I+A) allowed under evaluation:control (often about 2*z).
V	Value.

Second Character	Meaning: "Of the..."
1	First operand
2	Second operand
M	Modulus (for x\y).
N	Numerator (for x/y or x\y).
D	Denominator (for x/y).
I	Integer operand (if the other is fixed).
A	Fixed operand (if the other is integer).
R	Result (preliminary result if S exists).
S	Result required by evaluation:control.
B	Base in exponentation.

E Exponent.

Value of original bit:formula	Value of numeric:formula	Value of bit:formula resulting from SHIFT
11111	3	11000
11111	-1.7	01111
000001000000	5	100000000000
000001000000	-3	000000000100
101	3	000
101	-3	000
101	-2	001

bit:formula	&	bit:formula	result
10	1	101	
111001	00011110101		11100100011110101
00010000	0000010		000100000000010
0	0	00	

p	q	NOT p	p OR q	p EQV q	p AND q	P XOR q
0	0	1	0	1	0	0
0	1	1	1	0	0	1
1	0	0	1	0	0	1
1	1	0	1	1	1	0

```
0     = (assignment)  == (exchange)
1     EQV    XOR
2     OR
3     AND     (logical)
4     NOT
5     = < > <= >= <>     (relational)
6     &
7     + -
8     * / \   ( with or without evaluation:control )
9     **
10    indexing @      ( pointing, evaluation:control )
      @@      ( attribute:association )
```

In the algorithm it is necessary to consider several operands and operations simultaneously. The following diagram shows the relationships. All are pegged in relation to the present operand. Any operation may replace #.

A # B # C # D # E # F # G # H # I # J # K

the next operation

the prior operand the next operand

the prior operation the current operation

the present operand

The leftmost operand of the formula is initally the present operand.

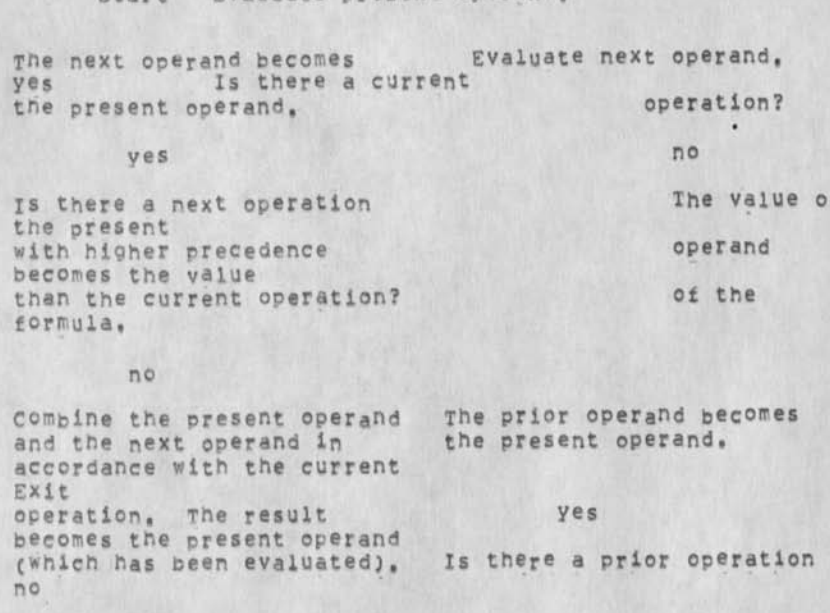


Figure 4-2. Combination Algorithm

ABC Operation type	XYZ type	Char	Bit	Int	Fix	Float
		Converted to				
ABC assignment (also parameter matching and exchange, both ways)	XYZ Char	Char	Bit	Bit	Bit	Bit
	Bit	Bit	Bit	Bit	Bit	Bit
	Int	Bit	Int	Int	Int	Bit
	Fix	Bit	Bit	Fix	Fix	Fix
	Float	Bit	Float	Float	Float	
ABC arithmetic	XYZ Float	Note 1	Note 2	Float	Float	Float
XYZ arithmetic	ABC Other	Note 3	Note 4	Scale	Scale	Float
ABC relational or XYZ relational	XYZ Char	Note 5	Int	Note 6	Note 6	Note 6
	Bit Note 6	Int	Note 6	Note 6	Note 6	
	Int Note 6	Int	Int	Scale	Float	
	ABC Float	Note 6	Int	Float	Float	
ABC & XYZ	Char	Char	Bit	Bit	Bit	Bit
XYZ & ABC	Other	Bit	Bit	Bit	Bit	Bit
ABC logical XYZ		Any	Bit	Bit	Bit	Bit
XYZ logical ABC						
Indexing, pointing		Note 7	Int	Int	Int	Int

Figure 4-3. Type Conversion

Entrance used	ALT (procedure:name)
integer	status:constant
normal 0	V (procedure:name)

```
first alternate          1          V ( first
alternate:entrance:name )
second alternate         2          V ( second
alternate:entrance:name )
etc.     etc.                       etc.
```

```
  parameter   type:function:call
qualified:status:constant
     type              value   status:constant

bit:formula              0              V(BIT)
V(TYPE:BIT)

integer:formula                  1              V(INT)
V(TYPE:INT)
(signed or unsigned)
fixed:formula            2              V(FIX)
V(TYPE:FIX)
(signed or unsigned)
floating:formula                 3              V(FLOAT)
V(TYPE:FLOAT)

character:formula                4              V(BYTE)
V(TYPE:BYTE)
```

```
START


          0-- skip A1, do A2
  A?       E=3

     1-- do A1

          0-- skip B1              0-- skip D1, do D2
(NULL
  B?                         D?
               do B2                          or none),
skip A2

     1-- do B1                   1-- do D1

                                      Skip D2 (if any
and A2
                                 E=2

          0-- skip C1, do C2 (NULL or none), skip B2
and A2
  C?

     1-- do C1

           Skip C2 (if any), B2 and A2
E=1        EXIT
```

	No terminator:phrase	Terminator:phrase
No initial:phrase, replacement:phrase, or increment:phrase	1A. Leave control:variable alone, execute controlled:statement just once.	1B. Same as 1A except execute controlled:statement zero or one time depending on terminator:phrase.
Initial:phrase only	2A. Initialize control:variable, execute controlled:statement just once.	2B. Same as 2A except execute controlled:statement zero or one time depending on terminator:phrase.
Replacement:phrase only	3A. Leave control:variable alone for the first execution. Before each subsequent execution of the controlled:statement, replace the value of thecontrol:variable.	3B. Same as 3A except test in accordance with terminator:phrase before every execution of the controlled:statement -- even the first one. Repeat executions "forever".
Increment:phrase only	4A. Same as 3A except add to value of control:variable instead of replacing value.	4B. Same as 4A except test as in 3B.
Initial:phrase and replacement:phrase	5A. Initialize control:variable. Execute controlled:statement once. Replace value of control:variable before each execution, before each subsequent execution. Repeat executions "forever".	5B. Same as 5A except check for termination before each execution.
Initial:phrase and increment:phrase	6A. Initialize control:variable. Execute controlled:statement once. Add to value of control:variable before each execution, before each subsequent execution.	6B. Same as 6A, except check for termination before each execution.

Repeat executions "forever".

```
                1
start           ALPHA           Set BETA to 3

                =

                ?   2 or 3      Set GAMMA to the
                                value of BETA

                    4           If GAMMA equals 2
                                set BETA to 2

                    6           Set BETA to the
                                value of GAMMA



                                Set ALPHA to 7  next

                    <1, 5 or >6
                                Undefined
```

```
                                                        Format:Lists

Input Buffer    Field                   ,,10C           10C,,

28,3b'bABCD'bABb                        1           '28,3bb'
'28,3bs27bABC'
                2               'bABCDb'            'Ds27bbbb'
                3               'ABbbbbbbbb'        'ABbbbb'

ALPHAbBET'AbTHERMOPILEb                             1       'ALPHAb'
'ALPHAbBETs27'
                2               'BETs27Ab'          'Abbbbb'
                3               'THERMOPILE'        'THERMO'
```

```
      4B3PS3PS3PS3P   5B5PS5P

Bah!  000 042 616 821 00011 62Q11
Humbug        487 56D 627 567 28ELM M4TB7



Input Buffer  Field   5N, 6N, SNSN

1,2E3=b485bb3b7               1        1,2E3
              2       =485
              3        37

+46b7,00015A1B,               1        illegal field
(contains blank)
              2       ,00015
              3       1,



      +SD4=3ZDDR      =SDSZSZSZ       +SZ","ZZ","DDR

  1573.64      + 1573   1574   1 5 7 3          + 15,74
  - 27         - 0027    -27   -2 7              - 27
    0.0          0000     00    0                 00
-10740         undefined       -10740  undefined       -
1,07,40



                                  Format;Lists
Input Buffer  Field   "SPEEDb",DDD,"MPH"      ,DDDSSSS

SPEEDb100bMPH         1                100     'SPEED'
              2                                100
```

```
      +ZZDD,DZZ        -4Z*3DR -4Z*     -4Z2*R

 1573,6405     +1573,64         1573641 1573      157
  -27   -27,0  -27000  -27       -3
    0,0          00,0     000
-10740         undefined        undefined        undefined
-1074



      1573,6400        1573,6410        1573,0000
1570,0000
       -27      -27      -27      -30
   0,0            0,0      0,0      0,0
      undefined        undefined        undefined
-10740



      +,6DE+3ZR        -SD*6ZSES-3Z*    -S3D,SES-3DR
+S3*5DSES+S3Z

+ 39,7528      +,397528E+2        397528  E   1   398, E
-001  + 39752 E + 4
-,008711246    -,871125E-2      - 8711246 E  -3 - 871, E
-005  - 87112
```

```
PROGRAM AA
   XX (table;name)

      PROC BB
         XX (item;name)

            PROC CC
               no occurrence of XX

                  PROC DD
                  XX used but not declared


                  PROC EE
```

Figure 7-1, Scope of Names

Serial Structure	Parallel Structure
1st half AB[0]	1st half AB[0]
2d half AB[0]	1st half AB[1]
XY[0]	1st half AB[2]
1st half AB[1]	1st half AB[3]
2d half AB[1]	2d half AB[0]
XY[1]	2d half AB[1]
1st half AB[2]	2d half AB[2]
2d half AB[2]	2d half AB[3]
XY[2]	XY[0]
1st half AB[3]	XY[1]
2d half AB[3]	XY[2]
XY[3]	XY[3]

Example: Table MN has 2 items, AB and XY, and 4 entries, 0, 1, 2, and 3,
Item AB occupies 2 words,
Item XY occupies 1 word,
Note: 12 consecutive computer words are shown in each illustration above.

Figure 7-2. Serial and Parallel Table Structure

Tight Structure

entry [0]	entry [1]	entry [2]
entry [3]	entry [4]	entry [5]

A table of six entries is medium packed, three entries to the word.

```
OVERLAY AA, AB, AC : BA, (BX : BY, BZ), BC ;
  AC            AA              AB
   BA             BX              BC
                  BY   BZ
```

```
RELATED STRUCTURE        AA,100,(BB : EE)        CC,EE,DD
200,FF,GG : DD
```

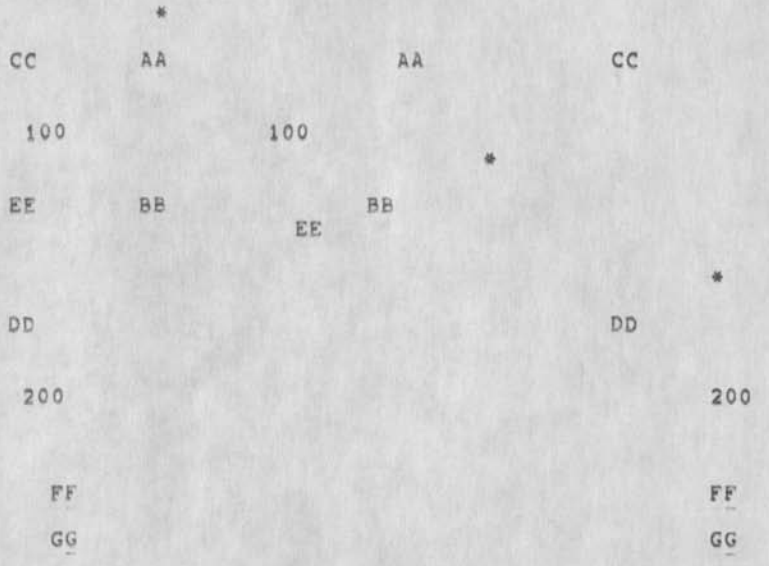


Figure 7-4. Allocation of a Related Structure

```
entrance        number          status:constant

normal                  0        V( procedure:name )
first alternate                  1       V(
alternate:entrance:name )
second alternate                 2       V(
alternate:entrance:name )
```

RELATIVE WORD	SERIAL	PARALLEL
0	1 AB[0,0,0]	1 AB[0,0,0]
1	2 AB[0,0,0]	1 AB[0,0,1]
2	XY[0,0,0]	1 AB[0,1,0]
3	1 AB[0,0,1]	1 AB[0,1,1]
4	2 AB[0,0,1]	1 AB[0,2,0]
5	XY[0,0,1]	1 AB[0,2,1]
6	1 AB[0,1,0]	1 AB[0,3,0]
7	2 AB[0,1,0]	1 AB[0,3,1]
8	XY[0,1,0]	1 AB[1,0,0]
9	1 AB[0,1,1]	1 AB[1,0,1]
10	2 AB[0,1,1]	1 AB[1,1,0]
11	XY[0,1,1]	1 AB[1,1,1]
12	1 AB[0,2,0]	1 AB[1,2,0]
13	2 AB[0,2,0]	1 AB[1,2,1]
14	XY[0,2,0]	1 AB[1,3,0]
15	1 AB[0,2,1]	1 AB[1,3,1]
16	2 AB[0,2,1]	1 AB[2,0,0]
17	XY[0,2,1]	1 AB[2,0,1]
18	1 AB[0,3,0]	1 AB[2,1,0]
19	2 AB[0,3,0]	1 AB[2,1,1]
20	XY[0,3,0]	1 AB[2,2,0]
21	1 AB[0,3,1]	1 AB[2,2,1]
22	2 AB[0,3,1]	1 AB[2,3,0]
23	XY[0,3,1]	1 AB[2,3,1]
24	1 AB[1,0,0]	2 AB[0,0,0]
25	2 AB[1,0,0]	2 AB[0,0,1]
.	.	.
.	.	.
.	.	.
62	XY[2,2,0]	XY[1,3,0]
63	1 AB[2,2,1]	XY[1,3,1]
64	2 AB[2,2,1]	XY[2,0,0]
65	XY[2,2,1]	XY[2,0,1]
66	1 AB[2,3,0]	XY[2,1,0]
67	2 AB[2,3,0]	XY[2,1,1]
68	XY[2,3,0]	XY[2,2,0]
69	1 AB[2,3,1]	XY[2,2,1]
70	2 AB[2,3,1]	XY[2,3,0]
71	XY[2,3,1]	XY[2,3,1]

Figure 10-1 Indexing and Storage Allocation

Bits				Bits		
0-1	2-11	12-21	22-31	0-1	2-11	12-21	22-31
0		BB[0,0]	BB[0,1]	BB[0,2]		BB[0,0]	BB[0,1]
1		BB[0,3]	BB[0,4]	BB[0,5]		BB[0,3]	BB[0,4]
2		BB[0,6]	BB[0,7]			BB[0,6]	BB[0,7]
3		BB[1,0]	BB[1,1]	BB[1,2]		BB[1,1]	BB[1,2]
4		BB[1,3]	BB[1,4]	BB[1,5]		BB[1,4]	BB[1,5]
5		BB[1,6]	BB[1,7]			BB[1,7]	BB[2,0]
6		BB[2,0]	BB[2,1]	BB[2,2]		BB[2,2]	BB[2,3]
7		BB[2,3]	BB[2,4]	BB[2,5]		BB[2,5]	BB[2,6]
8		BB[2,6]	BB[2,7]			BB[3,0]	BB[3,1]
9		BB[3,0]	BB[3,1]	BB[3,2]		BB[3,3]	BB[3,4]
10		BB[3,3]	BB[3,4]	BB[3,5]		BB[3,6]	BB[3,7]
11		BB[3,6]	BB[3,7]			BB[4,1]	BB[4,2]
12		BB[4,0]	BB[4,1]	BB[4,2]		BB[4,4]	BB[4,5]
13		BB[4,3]	BB[4,4]	BB[4,5]		BB[4,7]	BB[5,0]
14		BB[4,6]	BB[4,7]			BB[5,2]	BB[5,3]
15		BB[5,0]	BB[5,1]	BB[5,2]		BB[5,5]	BB[5,6]
16		BB[5,3]	BB[5,4]	BB[5,5]			
17		BB[5,6]	BB[5,7]				

Figure 10-2 Indexing and Allocating Tight Structure Tables

RJC 25-OCT-74 12:39 31226

Jovial Equations - Corrected version

(J31226) 25-OCT-74 12:39; Title: Author(s): Roberta J. Carrier/RJC;
Distribution: /RJC DLS; Sub-Collections: NIC; Clerk: RJC;
Origin: <CARRIER>JOVIALEQUA.NLS;2, 8-OCT-74 06:03 RJC ;

MIKE 25-OCT-74 13:00 31227

letter of transmittal for package of our stuff on educational tech, and on Delphi for a guy at AT&T

to Mr. Marty Guice, AT&T, New Brunswick, N.J.

MIKE 25-OCT-74 13:00 31227

letter of transmittal for package of our stuff on educational tech. and on Delphi for a guy at AT&T

Mr. Marty Guice
A T & T
P.O. Box 2017 (H.D.)
New Brunswick, New Jersey
United States 08903 1

Dear Mr. Guice: 2

Enclosed is the package of Business Planning reports that we discussed on the phone last week. 3

These reports document the work to date of the group in the fields of educational technology and technology assessment in education. There is also considerable emphasis in the material on the particular research methodologies used to generate the information. I hope you will find these sections equally useful. 4

We are continuing to explore areas related to educational futures (remote work centers, travel-communications substitution, computer-augmented management systems, etc.) and we would be most interested in learning more of your interest in these fields. In any case, let's keep in touch. 5

Thanks for your interest in our work. If I can be of any further assistance, please let me know.

6

Yours sincerely,

Michael T. Bedford
Supervisor- Business Planning 7

MIKE 25-OCT-74 13:00 31227

letter of transmittal for package of our stuff on educational tech.
and on Delphi for a guy at AT&T

(J31227) 25-OCT-74 13:00; Title: Author(s): Michael T. Bedford/MIKE;
Distribution: /PF; Sub-Collections: NIC; Clerk: MIKE;

FEED 26-OCT-74 20:33 31228

BRL ident: coordinator changed

The BRL group coordinator will be changed to JTN. Let us know any deletions or additions to the group. Thank you FEED *R 1

FEED 26-OCT-74 20:33 31228

BRL ident: coordinator changed

(J31228) 26-OCT-74 20:33;;;; Title: Author(s): Special Jhb Feedback/FEED; Distribution: /JTH([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: FEED;

FEED 26-OCT-74 20:36 31229

IDent request for BRL

Please make the BRL Group coordinator DFT. Thank you. FEED *R 1

FEED 26-OCT-74 20:36 31229

IDent request for BRL

(J31229) 26-OCT-74 20:36;;;; Title: Author(s): Special Jhb Feedback/FEED; Distribution: /JAKE([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: FEED;

FEED 26-OCT-74 20:38 31230

Correction to previous message, BRL coordinator

Sorry, the coordinator will be changed to DFT as requested, Thank you John, FEED,

FEED 26-OCT-74 20:38 31230

Correction to previous message, BRL coordinator

(J31230) 26-OCT-74 20:38;;;; Title: Author(s): Special Jhb Feedback/FEED; Distribution: /JTH([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: FEED;

MIKE 28-OCT-74 17:41 31232

short description of how H.Q. SYstems Planning uses CAMS type systems.....ie. STORET.

I'd like to include this paragraph in a review of the CAMS uses in Bell....how does it look ?

MIKE 28-OCT-74 17:41 31232

short description of how H.Q. Systems Planning uses CAMS type systems.....ie. STORET.

In response to a need for accurate and timely distribution of SERTT committee working papers (agendas, minutes of meetings, up-to-date status of Bell-funded BR research program proposals, memos from H.Q. Planning, etc.), H.Q. Planning developed a computer-based information storage and retreival package known as STORET. With the aid of STORET, members of the SERTT working committee are provided with timely information regarding their upcoming meetings. The STORET program was developed in H.Q. Systems Planning, with the encouragement of the Business Planning Group.

MIKE 28-OCT-74 17:41 31232

short description of how H.Q. Systems Planning uses CAMS type systems.....ie. STOREI.

(J31232) 28-OCT-74 17:41; Title: Author(s): Michael T. Bedford/MIKE; Distribution: /PAN; Sub-Collections: NIC; Clerk: MIKE;

JHB 29-OCT-74 19:41 31233

test

test of feedback 1

JHB 29-OCT-74 19:41 31233

test

(J31233) 29-OCT-74 19:41;;;; Title: (Unrecorded) Title:
Author(s): James H. Bair/JHB; Distribution: /FEEDBACK([ACTION]) JHB(
[INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: JHB;

FEED 29-OCT-74 21:22 31234

Set NNLS file private command (MJOURNAL, 24331,

Mike, The Set NLS file private command now works.
This (MJOURNAL, 24331, 1:w) is the msg refered to in our earlier discussions that was sent to many people, so it's difficult to tell who should or did take action. Please note that the initial inquiry was sent to Feedback at ARC which is an internal mechanism not serving Utility clients. Thank you, Feed

FEED 29-OCT-74 21:22 31234

Set NNLS file private command (MJOURNAL, 24331,

(J31234) 29-OCT-74 21:22;;;; Title: Author(s): Special Jhb Feedback/FEED; Distribution: /MAP2([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: FEED;

File Locking Conflicts

I have been confused and concerned for some time over the various ways that one can get his or someone elses' files locked. This envariably leads to excessive time and effort spent in recovering, sometimes lost input. The problem is particularily severe when combined with a shortage of disk space. This happens enough times here, especially with the PSO personnel, who do not have directories, and often work in a multiplictiy of others, logged in as a number of different people, some times conneected to another directory. You can see how one could become confused and write on files that in theory she/he should not be able to. 1

Let me relate to you a series of events that recently transpired, while trying to find out the ins and outs of file locking conflicts. 2

I loaded Kennedy's initials file, ejk. A Show File Status revealed that it was unlocked and not being modified. 2a

I then proceeded to insert a statement, which it appeared to accept gracefully. 2b

my directory listing then revealed a file < STONE, (KENNEDY)EJK.PC;33, > 2c

I then proceeded to Update File compact, after which the system replied:
<KENNEDY>EJK.NLS;34 cannot be opened 2d

A listing of Kennedy's directory revealed:
<KENNEDY, EJK.NLS;33, > [Being Modified By STONE (DLS] 2e

When Kennedy tried to make a change in his initials file (version 33), the system would not allow it. 2f

I then loaded another file and was told that <KENNEDY>EJK.NLS;34 cannot be closed 2g

Kennedy then did an Update File Compact, with some resulting error message which we did not capture. 2h

A listing of Kennedy's directory revealed a file
< KENNEDY>EJK.NLS;34, >, in addition to the version 33 being modified by stone. 2i

When I tried Load File Kennedy,ejk.NLS;34, (with use of the escape key), I was summarily dismissed from NLS with the error message: 2j

ILLEGAL INSTRUCTION JSYS 30 = 104000,,30 2j1

at JFNTOS+14 = 173773 2j2

When I reset and did a dir, a partial copy -- (KENNEDY)EJK.PC;33 was listed, When I tried to delete it from NLS, it responded with,,that file not on-line try interogate,, I was able to delete it from TENEX. 2k

When I see the protection on a file which says READ and NOT WRITE, I would expect to not be allowed to even attempt a write. I would expect NLS to tell me as soon as I put in an editing command of any kind "YOUCANNOTWRITEONTHISFILE!!!", or some other equally specific message. 3

To continue the tale of woe, and possibly indicate a bug: 4

I decided to check the protection of the files in Kennedy's directory, using the NLS SHOW Directory command, with the option of Protect. The printout went fine, until it reached his initials file, EJK. The printout halted, but a ^T said RUNNING. Probes over the next 15 minutes still indicated RUNNING (at different addresses), and the process was using about a minute of CPU for every 2 minutes of real time. This was confirmed by a link from Norton, who had been looking at system load and discovered I was using 55% of the CPU. Kennedy had a similar experience when he used the same command. 4a

Is there anyone at ARC who can: 5

Explain the file mechanisms to me in layman's terms? 5a

Tell me why one should even be given the appeareance of writing in anothers' files? 5b

Give a list of dos and don'ts to avoid file locking conflicts? 5c

Give a list of ways to undo damage under various common conditions? 5d

File Locking Conflicts

(J31235) 30-OCT-74 07:36;;;; Title: Author(s): Duane L. Stone/DLS; Distribution: /FEED([ACTION]) EJK([INFO-ONLY]) JPC([INFO-ONLY]) ELF([INFO-ONLY]) ; Sub-Collections: RADC; Clerk: DLS;

RSR 30-OCT-74 12:41 31236

Stan Price DOT Contact possibility

Mr. Stanley Price is director , System Analysis and Evaluation Division, Office of R&D, Urban Mass Transportation Administration, DOT, 202-426-4022. A Personal friend of mine. Would like to know more about AKW/nls. Let me know what happens. 1

RSR 30-OCT-74 12:41 31236

Stan Price DOT Contact possibility

(J31236) 30-OCT-74 12:41;;;; Title: Author(s): Robert S. Ratner/RSR; Distribution: /JCN([ACTION]) RSR([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: RSR;

RSR 30-OCT-74 12:44 31237

Catalog file

Several times I have tried to jump to a journal item (generally an older one) and been greeted with the above. Unsusccessful in retrieving from archive (told "not archived"). What gives? 1

Catalog file

(J31237) 30-OCT-74 12:44;;;; Title: Author(s): Robert S. Ratner/RSR; Distribution: /FDBK([ACTION]) RSR([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: RSR;

MIKE 30-OCT-74 13:16 31238

copy of message from Hopper to MIKE re printer program and NLS-8

the name of the program that runs the 1200 lineprocessor printing is <user-progs>lpprint.cent-exp;3, Since we'll be loosing this printer soon (this week ?), and won't be getting our own Centronix untill mid NOV,, could you give the program one last try before it goes, If you run into any probs, give me a shout.

copy of message from Hopper to MIKE re printer program and NLS-8

(HOPPER) 25-OCT-74 1412-PDT at SRI-ARC: Printer
Distribution: BEDFORD AT OFFICE-1
Received at: 25-OCT-74 14:12:37 1

Lppint,cent-exp probably got archived. I'll look in to it. NLS-8 has
been taking most of my time so I haven't done any thinking about what to
try next. There shouldn't be any problem using PREVIEW (NLS-8) instead
of old NLS. It should work just the same, if not better. 1a

MIKE 30-OCT-74 13:16 31238

copy of message from Hopper to MIKE re printer program and NLS-8

(J31238) 30-OCT-74 13:16; Title: Author(s): Michael T. Bedford/MIKE;
Distribution: /IMM; Sub-Collections: NIC; Clerk: MIKE;

loose ends

the shortage of telephone lines in bldg 390 has forced dr. banks to use the hazeltineAnts line for his unitech rje, this terminal had shown no customer usage and is physically in the way at the present location,do you have an alternate user in mind or should we return it to the ants site? 1

if possible i would like to have a file at office. 2

you might be interested in the progress cal and hel have made in acquiring their rje's. 3

hel (dean blazie) has not yet upgraded the pdp 1110 to the rje class, they are implementing a hp calculator with printer and tape cassette as an rje at present. 3a

cal (earl weaver) is still in the paperwork stage at local procurement...not yet advertised for bids. 3b

this is mel using don's file. 4

loose ends

(J31240) 31-OCT-74 11:11;;;; Title: Author(s): Donald F. Taylor/DFT;
Distribution: /SMT([ACTION]) ; Sub-Collections: NIC; Clerk: DFT;
Origin: < DTAYLOR, MEMO31OCT,NLS;2, >, 31-OCT-74 11:08 DFT ;;;;####;

MLK 31-OCT-74 17:41 31241

TEST FOR FEEDBACK

HERE IS A TEST MESSAGE FOR FEEDBACK, IT SHOULD GET DELIVERED AT THE 6:30 PM DELIVERY TONIGHT (THURS), THIS IS AFTER TRANSFERRING THE IDENTFILE, HOPE IT GETS THROUGH, MARCIA 1

MLK 31-OCT-74 17:41 31241

TEST FOR FEEDBACK

(J31241) 31-OCT-74 17:41; Title: Author(s): Marcia Lynn Keeney/MLK; Distribution: /FEEDBACK; Sub-Collections: SRI-ARC FEEDBACK; Clerk: JCP;

FGB 1-NOV-74 05:48 31242

test message

This is a message that I am typing in, Each of the above lines was terminated with a (Cr). Messages are typically short because they are printed Out whenever one asks tO see what mail he has. They usually suplement the actioxxx actuall mail. 1

FGB 1-NOV-74 05:48 31242

test message

(J31242) 1-NOV-74 05:48;;;; Title: Author(s): Frank G. Brignoli/FGB; Distribution: /FGB([ACTION]) CMC([INFO-ONLY]) ; Sub-Collections: NIC; Clerk: FGB;

horror

today is horrible

FGB 1-NOV-74 09:21 31243

horror

(J31243) 1-NOV-74 09:21;;;; Title: Author(s): Frank G. Brignoli/FGB; Distribution: /CMC([ACTION]) ILA([INFO-ONLY]) ; Sub-Collections: NIC; Clerk: FGB;

test message

FGB 1-NOV-74 09:43 31244

FGB 1-NOV-74 09:43 31244

test message

(J31244) 1-NOV-74 09:43;;;; Title: Author(s): Frank G. Brignoli/FGB; Distribution: /CMC([ACTION]) ; Sub-Collections: NIC; Clerk: FGB;

FGB 1-NOV-74 09:45 31245

test message 2

This is a test message. 1

FGB 1-NOV-74 09:45 31245

test message 2

(J31245) 1-NOV-74 09:45;;;; Title: Author(s): Frank G. Brignoli/FGB; Distribution: /CMC([ACTION]) ; Sub-Collections: NIC; Clerk: FGB;

NDM 1-NOV-74 13:20 31246

Program which marks phrases which appear in a glossary

This program marks words or phrases in a file which also appear in a glossary file. It will not mark words or phrases which are in the glossary but have already been marked as part of a larger phrase. It ignores the case that the phrase appears in.

Program which marks phrases which appear in a glossary

This program marks words or phrases in a file which also appear in a glossary file. It will not mark words or phrases which are in the glossary but have already been marked as part of a larger phrase. It ignores the case that the phrase appears in. 1

It puts some delimeter which you may set in front of the phrase, and another delimiter at the end of the phrase. Each of these delimeters should be unique, and should not appear elsewhere in the file. They may be any (reasonable) number of characters long. For some reason, it doesn't work when the delimiters include a semicolon. 1a

In the L10 part of this program there is a declaration statement where you may set the delimiters. They are currently set to & for the left delimiter and * for the right. Of course you can use these and then do a substitution on the file for something else (like a directive). 1a1

Then you will have to compile both the L10 and the CML part (if the REL files aren't already around). Use the PROGRAMS subsystem command Compile File. The branch "cglossary" should be compiled using CML to a file called GLOSS,CML, and "lglossary" using L10 to a file called GLOSS,SUBSYS. 1a2

The assumption is that the glossary file includes a number of statements beginning with a number of words in upper-case only, serarated by spaces AND FOLLOWED BY A PERIOD (.), all at the same level (though they may be in different branches). I.e. one phrase per statement, in upper-case, followed by a period. 1b

You must create a special reference file for this program to use based on the glossary file. To do this: 1b1

1) Create a new file. 1b1a

2) Set your content analyzer filter to the following pattern:

```
$(UL/SP) '. ;
```

1b1b

3) Do a Copy Filtered from the original glossary file to your new file. 1b1c

4) In the PROGRAMS subsystem, load this GLOSSARY program (Load Program GLOSS OK). 1b1d

5) Sort the plex in your new file using the Sort command in the GLOSSARY subsystem. 1b1e

6) Update your new reference file. 1b1f

```
   Then you may load the file(s) to be marked, and use the Mark File
   command in the GLOSSARY subsystem.  It will ask you to point to
   the file to be marked, then give the name of the reference file
   you created.                                                          1c

   This program is very slow; run it during low load-average periods.    1d

FILE cglossary % using (CML.SAV,) to (GLOSS.CML,) %                      2

   % COMMON RULES %                                                      2a

      % ENTITY DEFINITIONS %                                             2a1

         editentity = textent / structure;                               2a1a

      % TEXT ENTITY DEFINITIONS %                                        2a2

         textent = texti / "TEXT"!L1! / "LINK"!L1!;                      2a2a

         texti = "CHARACTER"!L1! / "WORD"!L1! / "VISIBLE"!L1! /
         "INVISIBLE"!L1! / "NUMBER"!L1!;                                 2a2b

      % STRUCTURE ENTITY DEFINITIONS %                                   2a3

         structure = "STATEMENT"!L1! / notstatement;                     2a3a

         notstatement = "GROUP"!L1! / "BRANCH"!L1! / "PLEX"!L1! ;        2a3b

   % DECLARATIONS %                                                      2b

      DECLARE PARSEFUNCTION                                              2b1

         answ,          % reads answer construct %                       2b1a

         answer,                % for questions - returns  0/1 %         2b1b

         sp,            % reads next char, TRUE if space %               2b1c

         readconfirm,   % reads next char if ca %                       2b1d

         readbug,       % reads next char if BUG %                       2b1e

         readoption,    % TRUE if next char is optchar %                 2b1f

         readrepeat,    % TRUE if next char is repeat %                  2b1g

         lookansw,      % TRUE if next char is Y/CA %                    2b1h

         lookconfirm,   % TRUE if next char is CA/REPEAT/INSERT %        2b1i
```

```
    lookdefault,  % TRUE if next char is CA/REPEAT/INSERT %          2b1j
    lookbug,      % TRUE if next char is BUG %                      2b1K
    looknum,      % TRUE if next char is a number %                 2b1l
    clearname,    % clears the name area %                          2b1m
    notca;                % reads next char, TRUE iff not CA
    char %                                                          2b1n
DECLARE EXTERNAL zinsstatement;                                     2b2
DECLARE EXT-KEYWORD % so only one copy exists in system These
keywords are defined as external strings in CONST, %                2b3
  % STRUCTURAL ENTITIES %                                          2b3a
    "BRANCH",                                                       2b3a1
    "GROUP",                                                        2b3a2
    "PLEX",                                                         2b3a3
    "STATEMENT",                                                    2b3a4
  % TEXTUAL ENTITIES %                                            2b3b
    "CHARACTER",                                                    2b3b1
    "INVISIBLE",                                                    2b3b2
    "LINK",                                                         2b3b3
    "NUMBER",                                                       2b3b4
    "PASSWORD",                                                     2b3b5
    "TEXT",                                                         2b3b6
    "VISIBLE",                                                      2b3b7
    "WORD",                                                         2b3b8
  % MISC. ENTITIES %                                              2b3c
    "FILE",                                                         2b3c1
    "OLDFILELINK",                                                  2b3c2
```

```
            "NEWFILELINK",                                          2b3c3
            "NAME",                                                 2b3c4
            "RETURN",                                               2b3c5
            "FILERETURN",                                           2b3c6
            "WINDOW",                                               2b3c7
            "MARKER";                                               2b3c8
        DECLARE EXTERNAL % not defined here %                       2b4
          nlssubs;                                                  2b4a
      SUBSYSTEM gloss KEYWORD "GLOSSARY"                            2c
        COMMAND                                                     2c1
          gmark =                                                   2c1a
            "MARK"!L1! "FILE"!L1!                                   2c1a1
            <"at"> dest _ SSEL(#"STATEMENT")                        2c1a2
            <"glossary file"> ent _ LSEL(#"OLDFILELINK")            2c1a3
            CONFIRM                                                 2c1a4
            xgloss(dest,ent) ;                                      2c1a5
        COMMAND                                                     2c2
          gsort =                                                   2c2a
            "SORT"!L1! "GLOSSARY"!L1! <"plex at">                   2c2a1
            dest _ DSEL(#"PLEX")                                    2c2a2
            CONFIRM                                                 2c2a3
            xsortg(dest) ;                                          2c2a4
        END,                                                        2c3
      FINISH                                                        2d
FILE lglossary % using (L10,SAV,) to (GLOSS,SUBSYS,) %              3
```

Program which marks phrases which appear in a glossary

```
DECLARE STRING ldelim="&", rdelim="*" ;                                    3a
(xgloss) PROCEDURE (resultptr, parsemode, file, glos) ;                    3b
   REF file, glos, resultptr ;                                             3b1
   LOCAL gfilno, start, backup ;                                           3b2
   LOCAL TEXT POINTER fptr, gptr, stptr1, stptr2, stptr3 ;                 3b3
   LOCAL STRING gword[200], str[2000] ;                                    3b4
   CASE parsemode OF                                                       3b5
      =parsing:                                                            3b5a
         BEGIN                                                             3b5a1
         %go to beginning of file%                                         3b5a2
            fptr _ file;                                                   3b5a2a
            fptr.stpsid _ origin;                                          3b5a2b
            start _ fptr ;                                                 3b5a2c
            fptr[1] _ 1;                                                   3b5a2d
         %open glossary file%                                              3b5a3
            lnbgls (&glos,0, $gword) ;                                     3b5a3a
            IF NOT FIND SF(*gword*) [',] THEN *gword* _ *gword*,
            ",NLS" ;                                                       3b5a3b
            gptr _ orgstid ;                                               3b5a3c
            gptr[1] _ 1 ;                                                  3b5a3d
            gptr.stfile _ gfilno _ open (0, $gword) ;                      3b5a3e
            ON SIGNAL ELSE                                                 3b5a3f
               BEGIN                                                       3b5a3f1
               ON SIGNAL ELSE;                                             3b5a3f2
               close (gfilno) ;                                            3b5a3f3
```

```
        END;                                                    3b5a3f4
    %for each phrase in glossary%                               3b5a4
      LOOP                                                      3b5a4a
        BEGIN                                                   3b5a4a1
        %find next glossary word or exit loop%                  3b5a4a2
          IF (gptr _ getnxt(gptr)) = endfil THEN EXIT LOOP
          ;                                                     3b5a4a2a
          IF NOT (FIND SF(gptr) 1s(UL/SP) ', "gptr _gptr)
                                                                3b5a4a2b
            THEN REPEAT LOOP ;                                  3b5a4a2b1
          *gword* _ SF(gptr) gptr ;                             3b5a4a2c
        % mark each statement in object file %                  3b5a4a3
          fptr _ start ;                                        3b5a4a3a
          LOOP                                                  3b5a4a3b
            BEGIN                                               3b5a4a3b1
            IF (fptr _ getnxt(fptr)) = endfil THEN EXIT
            LOOP ;                                              3b5a4a3b2
            *str* _ + SF(fptr) SE(fptr) ; %upper case%          3b5a4a3b3
            FIND SF(*str*) "stptr2 ;                            3b5a4a3b4
            LOOP                                                3b5a4a3b5
              BEGIN                                             3b5a4a3b5a
              lookUp (sstptr2, sgword, word1s) ;                3b5a4a3b5b
              IF stptr2=endfil THEN EXIT LOOP;                  3b5a4a3b5c
              IF (FIND stptr2 > [*rdelim*] "stptr3) THEN
                                                                3b5a4a3b5d
                BEGIN                                           3b5a4a3b5d1
                IF (FIND stptr2 [*ldelim*] "stptr1)
                                                                3b5a4a3b5d2
                  AND (POS stptr1 < stptr3)                     3b5a4a3b5d2a
```

```
                    THEN NULL                                   3b5a4a3b5d3
                    ELSE                                        3b5a4a3b5d4
                       BEGIN                                    3b5a4a3b5d4a
                       FIND stptr3 > ^stptr2 ;                  3b5a4a3b5d4b
                       REPEAT LOOP ;                            3b5a4a3b5d4c
                       END;                                     3b5a4a3b5d4d
                    END;                                        3b5a4a3b5d5
                 FIND stptr2 > $LD ^stptr2 <; %lookup
                 doesn't leave at end%                          3b5a4a3b5e
                 FOR backup _ gword.L DOWN UNTIL <= 0 DO
                 READC ;                                        3b5a4a3b5f
                 FIND ^stptr1 > ;                               3b5a4a3b5g
                 fptr[1]_stptr2[1]; %corresponding place in
                 stmnt%                                         3b5a4a3b5h
                 ST fptr fptr _ *rdelim* ;                      3b5a4a3b5i
                 ST stptr2 stptr2 _ *rdelim* ;                  3b5a4a3b5j
                 fptr[1]_stptr1[1]; %corresponding place in
                 stmnt%                                         3b5a4a3b5k
                 ST fptr fptr _ *ldelim* ;                      3b5a4a3b5l
                 ST stptr1 stptr1 _ *ldelim* ;                  3b5a4a3b5m
                 stptr2[1] _ stptr2[1] + rdelim.L +
                 ldelim.L ;                                     3b5a4a3b5n
                 END;                                           3b5a4a3b5o
              END;                                              3b5a4a3b6
           END; %LOOP%                                          3b5a4a4
        %done%                                                  3b5a5
          ON SIGNAL ELSE;                                       3b5a5a
```

```
            close (gfilno) ;                                        3b5a5b
            IF nlmode=fulldisplay THEN alldsp () ;                  3b5a5c
          END;                                                      3b5a6
        ENDCASE;                                                    3b5b
    RETURN (&resultptr) ;                                           3b6
    END,                                                            3b7
DECLARE FIELD chif=[0,7:35] ;                                       3c
(xsortg) PROCEDURE (result, parsemode, plexptr) ;                   3d
    REF result, plexptr ;                                           3d1
    LOCAL stid1, stid2, stidx, rlevcnt;                             3d2
    CASE parsemode OF                                               3d3
       = parsing:                                                   3d3a
          BEGIN                                                     3d3a1
          stid1 _ plxset (plexptr : stid2 ) ;                       3d3a2
          stidx _ colgdest(stid1 : rlevcnt) ;                       3d3a3
          sort (stidx, stid1, stid2, rlevcnt, ssortg, FALSE) ;      3d3a4
          IF nlmode=fulldisplay THEN alldsp() ;                     3d3a5
          END;                                                      3d3a6
       ENDCASE;                                                     3d3b
    RETURN (&result) ;                                              3d4
    END,                                                            3d5
(sortg)PROCEDURE(stid,outb,num);                                    3e
    LOCAL TEXT POINTER ptr1, ptr2 ;                                 3e1
    LOCAL STRING phrase[500] ;                                      3e2
    REF outb;                                                       3e3
```

Program which marks phrases which appear in a glossary

```
    IF FIND SF(stid) "ptr1 S(UL/SP) "ptr2 ', THEN                    3e4
      BEGIN                                                          3e4a
      *phrase* _ ptr1 ptr2 ;                                         3e4b
      outb _ - (phrase,L) ;                                          3e4c
      END                                                            3e4d
    ELSE outb _ 0;                                                   3e5
    RETURN (TRUE, 1) ;                                               3e6
    END.                                                             3e7
FINISH                                                               3f
```

NDM 1-NOV-74 13:20 31246

Program which marks phrases which appear in a glossary

(J31246) 1-NOV-74 13:20;;;; Title: Author(s): N. Dean Meyer/NDM;
Distribution: /SRL([ACTION]) RJ([INFO-ONLY]) JCN([INFO-ONLY]
) DCE([INFO-ONLY]) DVN([INFO-ONLY]) JHB([INFO-ONLY]) RLL([
INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: NDM; Origin: <
MEYER, GLOSS,NLS;20, >, 1-NOV-74 13:14 NDM ;;;;####;

DLS 1-NOV-74 13:32 31247

New Network Printer

We have acquired a second printer for Network use. We would like people to start using it for a couple of reasons...to get your reactions as users to the print style and the printer itself; and to help us in discovering and fixing any problems that might arise with it. We have not formally signed off on its delivery yet and would like a month or so experience with it before we do. 1

It is interfaced directly to the TIP, without a mini and magtape inbetween. This setup has advantages and disadvantages. 2

Advantages 2a

Two printers allow two people to be printing simultaneously, backup capability, etc. 2a1

Since it prints as it receives data, The problem of a second person sending while the first is printing (and thereby dumping the file on the floor) is nonexistent. One can be 99+% confident that if he is allowed access to the port for the new printer, that his file will be printed. 2a2

Disadvantages 2b

The interface buffer is limited to 1000 characters. It is possible (under conditions where the network is transmitting at full speed and the file contains short lines with lots of carriage returns/line feeds/form feeds) to lose data. There seems to be no problems with "regular" text files. 2b1

If one wants more than one copy, he must retransmit the file. Under normal network conditions, this should not cause problems, since network transmission rate and printing rate on both printers are closly matched. 2b2

Procedures for using the new printer 3

It is located next to the TIP right now. It is most easily accessed via the side door to the main facility room...around the corner from the main entrance, toward B Bay. It will be moved eventually...just inside the door to the Buffer room (old CDC-1604 room). 3a

When using SENDPRINT specify port 5 for the new printer. 3b

When the file has completed printing, press the STOP button, hit the HOME button a couple of times to advance the paper, and press the RUN button. DO not lift the cover of the printer and tear off paper inside, since the next time a file is sent the paper may not feed out the back properly, and will chew itself up. 3c

If you have any problems, comments, etc, send them to me or drop by the office.

Thanks

Stoney

DLS 1-NOV-74 13:32 31247

New Network Printer

(J31247) 1-NOV-74 13:32;;;; Title: Author(s): Duane L. Stone/DLS;
Distribution: /RADC([ACTION]) JHB([INFO-ONLY]) JCN([INFO-ONLY
]) ; Sub-Collections: RADC; Clerk: DLS;

My guess is that it would be a trivial CML fix.

I like RLL's suggestion in (24346,). I think the OK/C: is a much nicer way to do things in general than OK/[**]:. I feel the <control-u> should only be used where syntax requires it or for options very seldom used. 1

(J31248) 1-NOV-74 13:40;;;; Title: Author(s): N. Dean Meyer/NDM; Distribution: /FDBK([INFO-ONLY]) JHB([INFO-ONLY]) KIRK([INFO-ONLY]) RLL([INFO-ONLY]) ; Sub-Collections: SRI-ARC; Clerk: NDM;

Formatted Sequential Files

This change is now in the running version at ARC and will be transfered to office-1 whenever the next transfer is made.

NDM 1-NOV-74 13:56 31249

Formatted Sequential Files

I am pleased to announce the Elizabeth has just added an option to the Output Terminal command which allows you to put the result in a file instead of on your terminal. In keeping with all the other Output Processor commands, the syntax is: Output Terminal File OK. It then prompts for additional information as previously. This allows you to create formatted sequential files for SNDMSG or other purposes.

Formatted Sequential Files

(J31249) 1-NOV-74 13:56;;;; Title: Author(s): N. Dean Meyer/NDM;
Distribution: /FDBK([INFO-ONLY]) SRI-ARC([INFO-ONLY]) ;
Keywords: Output Processor, Sequential Files; Sub-Collections: SRI-ARC;
Clerk: NDM;

Functional Documents: Access and Numbering

See also (24325,) and (24341,) for opposing opinions.

Functional Documents and Publications Numbering 1

The following system now seems to exist: 2

For each functional document, there is a journalized file which tells about the document and points to the online version in the <USERGUIDES> directory (the most up-to-date version). 2a

Whenever we publish a document we also journalize it, assigning that publication a unique number. 2b

In fact: 3

Only a portion (at this point a relatively small portion) of our userguides have a journal file describing and pointing to them. 3a

We have published a number of userguides (before Dirk set up the functional document numbering scheme) with their journal numbers on them. 3b

The question: 4

When we publish userguides, should we print their actual journal number on them, or the number of the pointer file which always points to the most up to date version? 4a

Situations: 5

1) A user asks ARC for a publication. 5a

2) A user asks an architect for a publication. 5b

3) A user uses LOCATOR to find a publication. 5c

4) A user uses the Journal Catalogs to find a publication. 5d

5) A user asks a colleague (instead of ARC or their architect) for a publication. 5e

6) A user wishes to know if his hard copy is up to date, or more recent than another, in some substantive way. 5f

7) A user wishes to see how things were (to see what's different). 5g

Method One: 6

We could print the actual journal number on each publication, and journalize it under the title of "=== userguide publication file",

while the pointer files are journalized as "how to use the ---
userguide". 6a

Different journal numbers would represent the same functional
document but with substantive revisions. 6a1

Method Two: 7

We could print the number of the pointer file on the publication
while journalizing it under its own number. 7a

A journal number represents the functional document, not the
published version. 7a1

Situations vs Methods: 8

In situations 1 through 3, it won't matter which method we use.
ARC, Locator, and the KWAC will have the most recent printed
version and/or will know how to find the most recent version
online. 8a

Situation 4 tends to favor method two, but method one can be
handled by carefully naming the journalized files (as described
above). 8b

Situation 5 only method two will lead him to the most up-to-date
file. 8c

In situations 7 and 8 only method one will allow him easy
comparison. The date on a file is less dependable, since a user
may print out the current online version (with the day's date) and
it may not be substantively different from the last publication
(with a past date). 8d

Choice: 9

The choice should therefore depend on our perceptions of the
relative frequencies of situation 5 versus situations 6 and 7,
with some awareness of precedence. 9a

Two precedents apply: 9b

1) Every ARC publication that is in some way unique is
released with a unique number on it. 9b1

2) A few of the (less important, I think) userguides have been
printed with the number of the pointer file rather than the
actual journalized file. 9b2

Conclusion: 10

I feel Precedent 1 is far better established and more important, for all the reasons behind a sequentially numbered journal system. 10a

I feel that situation 5, asking a colleague for a document of import rather than one of the obvious sources of such information, is less common and less worthy of support than situations 6 and 7, the comparison of successive versions of a publication. 10b

I am rewriting the L10 Users' Guide and the Output Processor Users' Guide. I know that, except for intent, these new version will be very different (in content and approach) than the previous version. I would hate to see them confused. 10b1

Furthermore, I feel that publishing a document, to be used by beginners trying to understand the concept of a journal, with a number on it that looks like its journal number but is in fact the number of another journal item would be unduly confusing. 10c

I vote for Method One, Printing publications with their real journal numbers. 10d

Functional documents would exist to the degree that people make use of Locator (evidence is that they currently do) and that we are careful about naming journalized publications and pointer files. 10d1

Functional Documents: Access and Numbering

(J31250) 1-NOV-74 15:22;;;; Title: Author(s): N. Dean Meyer/NDM;
Distribution: /DIRT([INFO-ONLY]) DCE([INFO-ONLY]) JCN([
INFO-ONLY]) RWW([INFO-ONLY]) DVN([INFO-ONLY]) JMB([INFO-ONLY
]) POOH([INFO-ONLY]) JOAN([INFO-ONLY]) JHB([INFO-ONLY])
RLL([INFO-ONLY]) ; Sub-Collections: SRI-ARC DIRT; Clerk: NDM;
Origin: < MEYER, ANS,NLS;2, >, 1-NOV-74 15:18 NDM ;;;;####;

NDM 4-NOV-74 14:13 31251

REPEAT after Execute

When I accomplish a command using the Execute command and return to the herald, the <REPEAT> feature doesn't work. It neither repeats the command I just executed nor the prior command in the current subsystem nor just the Execute command. Is this the way it is supposed to be? (I am in Preview at Office-1.) 1

REPEAT after Execute

(J31251) 4-NOV-74 14:13;;;; Title: Author(s): N. Dean Meyer/NDM;
Distribution: /FDBK([ACTION]) RLL([INFO-ONLY]) ;
Sub-Collections: SRI-ARC; Clerk: NDM;

TEST

test...

TEST

(J31253) 4-NOV-74 16:06;;;; Title: (Unrecorded) Title: Author(s): Michael A. Placko/MAP2; Distribution: /MAP2([ACTION]) PWO([INFO-ONLY]) ; Sub-Collections: NIC; Clerk: MAP2;

JI 4-NOV-74 16:09 31254

Content Analyzer Pattern [before (d-t)]

After many unsuccessful attempts at employing BEFORE (date time) both singly and in combination as a Content Analyzer pattern, I would venture that NLS has a bug there. Respectfully, Jean 1

Content Analyzer Pattern [before (d-t)]

(J31254) 4-NOV-74 16:09;;;; Title: Author(s): Jean Iseli/JI;
Distribution: /FEEDBACK([ACTION]) JI([INFO-ONLY]) ;
Sub-Collections: NIC FEEDBACK; Clerk: JI;

FGB 5-NOV-74 11:01 31256

Meeting on Thursday for you to attend

Gretings mike how are you doing? We just found out that there is an ELF users meeting in San Diego on Thursday evening from 8-11 PM at the Town and Country hotel and from 9-12 Am on Friday. Frank cannot attend because of the short notice. Could you aeld you attend Thursday night session. Elf is a PDP 11 system which will be used as the basis for NALCON. Please acknowledge this message if you get it, Regards--Larry and Frank

1

FGB 5-NOV-74 11:01 31256

Meeting on Thursday for you to attend

(J31256) 5-NOV-74 11:01;;;; Title: Author(s): Frank G. Brignoli/FGB; Distribution: /CMC([ACTION]) ILA([INFO-ONLY]) ; Sub-Collections: NIC; Clerk: FGB;

RJM2 5-NOV-74 12:13 31258

GREETINGS FROM NSRDC

Hi Mike how's it going. If you are readinng this I guess it's safe to assume that you got out of LV without losing your shirt. How did the bets turn out? I assume that you've seen my other message about Rhodes. If not I suggest you read it as soon as possible. Enjoy Calf., it started raining here last night and is supposed to continue for the next few days. ENJOY!!!.....Roger

1

RJM2 5-NOV-74 12:13 31258

GREETINGS FROM NSRDC

(J31258) 5-NOV-74 12:13;;;; Title: Author(s): Roger J. Martin/RJM2;
Distribution: /CMC([ACTION]) RJM2([INFO-ONLY]) ;
Sub-Collections: NIC; Clerk: RJM2;

loes your documentation maintenance responsibility include the duty
lf checking with the author to see if there have been or are to be
lny material revisions in the documentation before reprinting the
lnline or camera-ready file? 1

ıe 24413

(J31260) 5-NOV-74 15:47;;;; Title: Author(s): N. Dean Meyer/NDM;
)istribution: /DVN([ACTION]) POOH([ACTION]) JML([ACTION])
;LJ([ACTION]) ; Sub-Collections: SRI-ARC; Clerk: NDM;

NDM 5-NOV-74 15:53 31261

Interface to Sequential World

I would like my SNDMSGs delivered to an ACTION branch if I am a member of the primary distribution list, and to an Info-only branch if I am on the CC list.
I like Dick's list of possibilities, and would find all of them useful at some time. (24410,) 1

NDM 5-NOV-74 15:53 31261

Interface to sequential World

(J31261) 5-NOV-74 15:53;;;; Title: Author(s): N. Dean Meyer/NDM;
Distribution: /FDBK([INFO-ONLY]) RWW([INFO-ONLY]) ;
Sub-Collections: SRI-ARC; Clerk: NDM;

FEED 5-NOV-74 19:29 31262

Re:31254 -- Bug in Content Analyzer

Jean, The bug seems to be random, We are working on it. In the interim, it has a higher probility of working if you get it to compile the first time after reset or entry into NLS. Thanks, Feed/jim

1

FEED 5-NOV-74 19:29 31262

Re:31254 -- Bug in Content Analyzer

(J31262) 5-NOV-74 19:29;;;; Title: Author(s): Special Jhb Feedback/FEED; Distribution: /JI([ACTION]) FEED([INFO-ONLY]) JDH([INFO-ONLY] see bugs branch) ; Sub-Collections: SRI-ARC; Clerk: FEED;