About MAGIC

- Proprietary offshoot of MUMPS, aka “M”
- Developed by Dr. Octo Barnet with help from A. Neil Papalardo at Mass General
- Designed to be a lightweight language, particularly suited for string (text) data
- Popular for Medical applications such as Sunquest, Cerner and IDX
MAGIC - Proprietary OS/MAGIC language

C/S - Windows OS/MAGIC language aka VMAGIC

FS - Functional System. C/S platform, EMR & PCS

FOCUS (M/AT 6.0) - EMR, PCS, POM, OE EDM move to FS with toolset. PCS data NOT returned to C/S
New Focus Report Writer

- Goals - No code required
- Reports from FOCUS and C/S databases
- C/S VMAGIC can get data from FOCUS with some complicated programming (iDAD)
• LEFT to RIGHT evaluation
• Everything is a string
• Value = True
• No Value (nil) = False
• Powerful string operators (like MUMPS)
• Weak math skills
  10+2*5= 60 (!)  5/10 = 0 (!)
Data automatically stored in tree

\[ *\text{AA[aa]} = \text{data} \]

Data automatically sorted by subscripts

No need for searching or sorting algorithms
• Variable name + string < 256 in length
• Lines of code < 256 in length
• MAGIC only - 1K for variables and their values across all programs in session unless you stack symbol table
In MAGIC

- 1024 bytes is all you get unless
  \texttt{%ADM.PAT.zcus.is.your.macro.M.do(urn)X}
- You get space back when you nil a variable
• Symbol table limit applies to local variables only: STUFF, x, y, aa
• Not to slash variables
  /STUFF b.dis.date e.dis.date @.user
• b.dis.date translates to /b.discharge.date @.user translates to /.USR
Data is packed or queued
Q(“FIRST”, “SECOND”, “THIRD”)^STUFF

• STUFF|0 = FIRST
• STUFF|1 = SECOND
• STUFF|2 = THIRD
Alternative Syntax

```haskell
{"FIRST","SECOND","THIRD"} ^ STUFF
"THIRD" ^ STUFF | 2
Add quotes automatically
`FIRST,SECOND,THIRD` ^ STUFF
```
STUFF = FIRST^SECOND^THIRD

- STUFF|0=FIRST
- STUFF|1=SECOND
- STUFF|2=THIRD
- STUFF^\{A,B,C\}
- A=FIRST, B=SECOND, C=THIRD
1) String operators
2) IF syntax
3) @Next, @Prev, + and –
4) DO syntax

-----------------
MAGIC – Prefix management/Looping
C/S – Opening Database/Looping
String Operators

# string at the position ABC#1 = B
$ to the left YYYYMMDDD$4= YYYY
% to the right YYYYMMDDD%5 = DD
` = not

ABC’#2 = AC
YOURSTRING

- 0123456789
- YOURSTRING#3 = R
- YOURSTRING%3$3 = STR
- YOURSTRING’#3 = YOUSTRING
ER admissions by hour of the day:

- xx.hour
- DAT=INT
- LEN=2
- VAL=@service.time$2+0
“BERMAN,JOEL F”"#"0,” = BERMAN

“BERMAN,JOEL F”"#"1,” = JOEL F

“BERMAN,JOEL F”"#"1,""#"0<space>” = JOEL
Mnemonics

- \text{NUR.COCJ}^{	ext{"0,"}} = \text{NUR}
- \text{IS.SMIF}^{	ext{"0."}} = \text{IS}
- \text{PURC.JOE}^{	ext{"0."}} = \text{PURCH}
Strip SSN of Dashes

NNN-NN-NNNN’#3’#5
NNN-NN-NNNN’#3 = NNNNNN-NNNN
01234567890
NNNNN-NNNN’#5
0123456789
(~ for general stripping)
Left to Right – no precedence of operation. Decimal precision:

- $+ - *$ places = operand with most places
- $/$ decimal places in numerator – places in denominator:
  
  $5/10 = 0$ (zero places – zero places = zero)
  
  $5.0/10 = 0.5$ (1 place – zero places = 1 place)
X * 1.000000000000/Y :2D

Give X lots of places, then round with :nD
n = desired number of places with 5/4 rounding
More Punctuation \“:\”

: = format operator
:nD = round to n decimals
:nT = truncate to n characters
:nTL = truncate to n characters, left pad
:nTR = truncate to n characters, right pad
For zero padding: %Z.zero.fill() see your mouse pad for details
Look at MT source code:

- Magic  F(4) \Name of Program
- C/S F(5) DPM, then procedure
- Lookup available
- Arguments usually at top of program
INIT report from standard

- “The more you need it, the less likely it is to work”
- List to report to paper
- Create report with same selects/segments
Copy MEDITECH Macros

INIT report from standard

• Copy macros with F(4) MAGIC
• Exact name match needed in MAGIC
  F(5) C/S
• Put “M” at procedure prompt in C/S
Syntax Issues

- Take @ sign out of @Next subscripts
  \[
  \text{DO}\{@\text{Next}(\@dx)\ \text{change to } @\text{Next}(dx)\}
  \]

- Loop instead of Killing
  \[
  K(/STUFF) \ @\text{Kill}(/STUFF) \ @K(^/STUFF)
  \]

- \[
  \text{DO}\{>/STUFF[SUB]^SUB ""^>/STUFF [SUB]\}}
  \]
• Writing to @ 141^@Z.last.key
• Change to 141^/Z
• Writing to /. 1^@.pha.site ".PHA.SITE"^XXX, 1^/[XXX]
IF{condition statement}

IF{condition statement; Nextcond statement; Nextcond statement; Finalvalue}
IF{@age.in.years^AGE<18 "Child";
AGE<65 "Adult";
AGE "Senior";
"No Age Available"}
• Left hand value returned if comparison is true
• Nil returned if comparison is false
• Forgetting LEFT TO RIGHT (!)
• Not knowing that
  ! = minimum (not really OR)
  & = maximum (not really AND)
IF{@age.in.years^AGE<18!AGE>65 "Y"} 
IF{17<18!17>65 "Y"} 
IF{17!17 returns 17 
IF{17>65 "Y"} fails 

Solution: Add Parenthesis 
IF{@age.years^AGE<18!(AGE>65) "Y"}
+ is most essential operator in MAGIC

Moves thru structures one subscript level at a time

List next doctor (or first if DOC var is initially nil)

+\GU[DOC]^DOC
Sense of +

- + pushes subscript into variable
- Value of this expression goes from first doctor to last doctor, then to nil

$+GU[DOC]^DOC$
Combine + and DO and you have a report writer

DO{+\GU[DOC]^DOC N(DOC)^#}
• \texttt{DO\{while.true<space>DOSTUFF\}}
• So list Doctors from start to "C"
\texttt{DO\{+\textbackslash GU[DOC]^DOC"D" N(DOC)^#\}}
Loop on c.location, build list of locations in string and print on report
Difference between + and @Next

- @Next is for structures in the Data Definition
- @Next(dx)
- @Next(room,@room.bed.index)
@Next uses data definition to figure out the subscripts and structure to loop on:

- @Next(dx)
- +?DZ[dz]DX[dxN]^dxN
- @Next(room,@room.bed.index)
- +:AARB[ggb,ggrB]^ggrB]
- @Next(subscript) or @Next (subscript,index)
Scheduled and registered patients on same report

- Loop on Registration Index AND
- Loop on Scheduled Index "yourself"
- Put urns into list in slash
+(/STUFF[SUB],DATA)^SUB
+ on subscript and get value of node in one operation
@Next.get(bed,@room.bed.index,urn)
>:AARB[SUB]^SUB
>@room.bed.index[SUB]^SUB

SUB#0S = facility
SUB#1S = room
SUB#1S = bed

You can use > in Next Get syntax

>(@room.bed.index[SUB],DATA)^SUB
There is also physical previous (<)
No translator operator for > and <
i.e. nothing like: @Phys.Next(STUFF)
We Can Help!

Joe Cocuzzo
Vice President, NPR Services
Iatric Systems, Inc.
(978) 805-4115
Joe.Cocuzzo@iatric.com

Attend our free monthly webcasts.
Subscribe to our newsletter.
Thank you.