VistA M Routine Analyzer

Release 1.00

Ray Group International

February 9, 2012

Abstract

This paper describes a set M tags that can be used to analyze M routines and Fileman dictionaries in VistA installations for package to package dependencies and package entry tags.

Contents

1. Introduction

VistA code is organized in terms of packages. These packages depend on each other in variety of ways. Understanding these dependencies is important both for discussions on iEHR architecture and current VistA refactoring effort. The tags discussed in this paper can be used to generate various reports on these dependencies.

The code here is an improved and extended version of XINDEX utility which has been long used to validate new and edited routines for VistA. The code essentially parses all the routines in a VistA installation and stores relevant information in M Globals. The Global structures in this version closely mimic what was used in XINDEX. The Globals are then used to write various reports.

The following are the information available in the reports.

1. For any tag
    a. Formal Parameters.
    b. Input/Output Parameters.
    c. Directly used Globals.
    d. Globals used through Fileman calls.
    e. Number of Read/Write commands.
    f. Number of Indirections used.
    g. Number of Xecute statements used.
2. All entry tags in a package that is being called from other packages.
3. All tags in other packages that is being called from a package.
4. All tags in a package that is used in Options File.
5. All tags in a package that is used in RPC calls.

2. Repository

The code is being hosted in the following Git repository:

- https://github.com/OSEHR/M-RoutineAnalyzer

3. Code Walkthrough

The code is based on the XINDEX utility. A few procedures have been added in order to tailor the XINDEX output to our needs. The most important tags are described in the following chapter.

3.1.MAIN^ZZRGND14(GLB): Prepare data for reports

This is the first command to be run in order to prepare the data for the reports. This procedure parses all VistA codebase and creates a global containing indexed information about different patterns found in source code. By default the indexed data is stored in ^ZZRG. It calls the indexing procedures in the required order.

Code

```
MAIN(GLB) ;Parses VistA codebase and extracts relevant information
 S:$G(GLB)="" GLB="^ZZRG"
 D CRTBASE(GLB)
 D NDXFI(GLB)
 D NDXOPT(GLB)
 D NDXRPC(GLB)
 D NDXIN(GLB,1)
 Q
 ;
```

3.1.1. CRTBASE^ZZRGND14(GLB): Parse VistA codebase

This tag parses all the routines in VistA installation and stores the information in input parameter

GLB. It will store the commands found in code, the locals used and whether they are newed, initialized or killed, globals and their usage and the lines where naked globals are used. To do that it calls the original XINDEX parser in “silent” mode.

Code

```
CRTBASE(GLB) ;Parses codebase & extracts relevant info
 N CNT,RTN,INP,NRO,DA,INDDA
 S:$G(GLB)="" GLB="^ZZRG"
 K @GLB
 D NEWINP(.INP)
 S INP(10)=9.4,INP(1)=1,DA=-1,INDDA=DA
 S CNT=0,RTN=""
 K ^UTILITY($J)
 F  S RTN=$O(^ROUTINE(RTN)) Q:RTN=""  S CNT=CNT+1,^UTILITY($J,RTN)=""
 S ^UTILITY($J,0)=CNT_";ROU",NRO=CNT
 D MAIN^ZZRGNDX(1)
 M @GLB=^UTILITY($J)
 K ^UTILITY($J)
 Q
 ;
```

Output

```
   @GLB@(0)=”N;ROU” where N is the number of routines.
   @GLB@(1,RTN,0)=…..
   @GLB@(1,RTN,0,0)=Number of lines
   @GLB@(1,RTN,0,LINE_NO,0)=Source code
   @GLB@(1,RTN,”COM”,LINE_NO)=Line label and commands separated by <TAB>
   @GLB@(1,RTN,”L”,GLOBAL)=Flags (~=Newed,*=Initialized,!=Killed)
   @GLB@(1,RTN,”G”,GLOBAL,0)=Labels+offsets where GLOBAL is used
separated by “,”
   @GLB@(1,RTN,”I”,INTERNAL_TAG,0)=Labels+offset where internal tags are
called
   @GLB@(1,RTN,”L”,LOCAL)=Flags (~=Newed,*=Initialized,!=Killed)
   @GLB@(1,RTN,”L”,LOCAL,0)=Labels+offset where the local is used.
   @GLB@(1,RTN,”N”,”^(“,0)=Labels+offset where naked global is used.
   @GLB@(1,RTN,”RSUM”)=
   @GLB@(1,RTN,”T”,TAG)=Label
   @GLB@(1,RTN,”X”,TAG ROUTINE,0)=Labels+offset where external tags are
called.
```

3.1.2. NDXFI^ZZRGND13(GLB): Index calling tags

The tag uses the output of `CRTBASE^ZZRGND14` and generates a map of all tags for a package that are called from other packages.

Code

```
NDXFI(GL) ;
```

```
 N RTN,TAGS,TAG,PKG,T,TRTN,TTAG,TPKG,I,J
 S RTN=""
 F  S RTN=$O(@GL@(1,RTN)) Q:RTN=""  D
 . S PKG=$$GETPKG^ZZRGND19(RTN,GL)
 . S:'$D(@GL@(4,PKG)) @GL@(4,PKG)=0
 . S T=""
 . F  S T=$O(@GL@(1,RTN,"X",T)) Q:T=""  D
 . . S TRTN=$P(T," ",1)
 . . S TTAG=$P(T," ",2)
 . . Q:TTAG=""
 . . S TPKG=$$GETPKG^ZZRGND19(TRTN,GL)
 . . Q:PKG=TPKG
 . . I '$D(@GL@(4,TPKG,PKG))
S @GL@(4,TPKG)=+$G(@GL@(4,TPKG))+1,@GL@(4,TPKG,PKG)=""
 . . S I=""
 . . F  S I=$O(@GL@(1,RTN,"X",T,I)) Q:I=""  D
 . . . S TAGS=@GL@(1,RTN,"X",T,I)
 . . . F J=1:1:$L(TAGS,",") D
 . . . . S TAG=$P(TAGS,",",J)
 . . . . S @GL@(8,PKG,TPKG,TRTN,TTAG)=""
 . . . . S @GL@(9,TPKG,TRTN,TTAG,PKG,TAG_"^"_RTN)=""
 Q
 ;
```

Output

```
@GLB@(9,TPKG,TRTN,TTAG,PKG,TAG^RTN)=""
```

TPKG	Target package namespace
TRTN	Target routine
TTAG	Target tag
PKG	Calling package
TAG^RTN	Calling tag

Example

```
^ZZRG(9,"GMPL","GMPLSAVE","EN","OR","EDSAVE+16^ORQQPL1")=""
^ZZRG(9,"GMPL","GMPLSAVE","EN","OR","^ORQQPL1")=""
```

3.1.3. NDXOPT^ZZRGND13(GL): Index OPTION calls

This procedure generates a map of all tags for a package that are called through OPTION tags.

Code

```
NDXOPT(GL) ; Indexes dependencies gathered from OPTION file
 N PKG,RTN,TAG,OPT,OPTNAME,CMD,I
 F I=1:1:+$P(^DIC(19,0),"^",3) D:$D(^DIC(19,I,0))
 . S OPT=^DIC(19,I,0)
 . Q:$P(OPT,"^",4)'="R"  Q:'$D(^DIC(19,I,25))
 . S OPTNAME=$P(OPT,"^",1)
 . S RTN=$P(^DIC(19,I,25),"^",2)
```

```
 . Q:$G(RTN)=""
 . S TAG=$P(^DIC(19,I,25),"^",1)
 . S PKG=$$GETPKG^ZZRGND19(RTN,GL)
 . S @GL@(13,PKG,RTN,TAG,OPTNAME)=""
 Q
 ;
```

Output

```
@GL@(13,PKG,RTN,TAG,OPTNAME)=””
```

PKG	Package namespace
RTN	Routine name
TAG	Tag name
OPTNAME	OPTION tag name from ^DIC(19

Example

```
^ZZRG(13,"OR","ORAMSET","SETCLIN","ORAM CLINIC PARAMETERS")=""
^ZZRG(13,"OR","ORAMSET","SETDIV","ORAM DIVISION PARAMETERS")=""
```

3.1.4. NDXRPC^ZZRGND13(GL): Index RPC calls

This procedure generates a map of all tags for a package that are called by the RPC broker.

Code

```
NDXRPC(GL) ; Indexes tags called by RPC Broker
 N I,ENTRY,RPC,PKG,RTN,TAG
 S I=""
 F  S I=$O(^XWB(8994,I)) Q:I=""  D
 . Q:+I=0
 . S ENTRY=^XWB(8994,I,0)
 . S RPC=$P(ENTRY,"^",1)
 . S RTN=$P(ENTRY,"^",3)
 . Q:RTN=""
 . S TAG=$P(ENTRY,"^",2)
 . S PKG=$$GETPKG^ZZRGND19(RTN,GL)
 . S @GL@(14,PKG,RTN,TAG)=I
 . S @GL@(14,PKG,RTN,TAG,RPC)=""
 Q
 ;
```

Output

```
@GL@(14,PKG,RTN,TAG,RPC)=""
```

PKG	Package namespace
RTN	Routine name
TAG	Tag name

RPC Remote Procedure Call tag name

Example

```
^ZZRG(14,"OR","ORAMSET","GET","ORAMSET GET")=""
^ZZRG(14,"OR","ORAMSET","GETCLINS","ORAMSET GETCLINS")=""
^ZZRG(14,"OR","ORAMSET","INDICS","ORAMSET INDICS")=""
```

3.1.5. NDXIN^ZZRGND13(GL,CLEAN): Index usage information per tag

This procedure goes through the tree generated by `CRTBASE^ZZRGND14` and indexes the information in a format ready to be used by `REPORTAPI^ZZRGND13`. To do that it goes through the indices created by the previous procedures (direct calls, OPTION calls, RPC calls) and outputs all the dependencies found in a structure ready to be processed by `REPORTAPI^ZZRGND13`.

Output

```
^ZZRG(7,PKG,RTN,TAG,"CMD","@")= Number of tags called through indirection
^ZZRG(7,PKG,RTN,TAG,"CMD","R")= Number of Read commands encountered
^ZZRG(7,PKG,RTN,TAG,"CMD","W")= Number of Write commands encountered
^ZZRG(7,PKG,RTN,TAG,"CMD","F",PARAMETER_NAME)= Formal parameters
^ZZRG(7,PKG,RTN,TAG,"CMD","FMG",GLOBAL_NAME)= Globals used through
FileMan calls
^ZZRG(7,PKG,RTN,TAG,"CMD","G",GLOBAL_NAME)= Globals used directly by this
tag
^ZZRG(7,PKG,RTN,TAG,"CMD","GL",GLOBAL_NAME)= Package globals
^ZZRG(7,PKG,RTN,TAG,"CMD","I",VARIABLE_NAME)= Input variables
^ZZRG(7,PKG,RTN,TAG,"CMD","O",VARIABLE_NAME)= Output variables
```

Example

```
^ZZRG(7,"GMPL","GMPLX1","PARAMS","CMD","W")=3
^ZZRG(7,"GMPL","GMPLX1","PARAMS","FMG","^GMPL(125.99")=""
^ZZRG(7,"GMPL","GMPLX1","PARAMS","FMG","^ORD(101")=""
^ZZRG(7,"GMPL","GMPLX1","PARAMS","G","^GMPL(125.99")=""
^ZZRG(7,"GMPL","GMPLX1","PARAMS","G","^ORD(101")=""
^ZZRG(7,"GMPL","GMPLX1","PARAMS","GL","^GMPL(125.99")=""
^ZZRG(7,"GMPL","GMPLX1","PARAMS","GL","^ORD(101")=""
^ZZRG(7,"GMPL","GMPLX1","PARAMS","I","U")=""
```

3.2.REPORTAPI^ZZRGND13(GLB)

Prints the tags from a particular package that are called from other packages, OPTION file or RPC broker. For each tag it lists the formal as well as input and output parameters, directly used globals, globals used through FileMan calls and package dependencies from faux routines.
The information is organized as follows:

CALLING PACKAGES: List of packages that call this tag
CALLING RPC's: List of RPC Broker tags that call this tag
CALLING OPTIONS: List of menu OPTIONs that call this tag
FORMAL: List of formal parameters
INPUT: List of input parameters
OUTPUT: List of output parameters
GLBS: List of directly used globals
GLS MDEP: List of globals used by faux routines called from this tag (M-code dependencies)
FM GLBS: List of globals touched through FileMan calls
READ: Number of Read commands
WRITE: Number of Write commands

Code

```
REPORTAPI(GLB) ;
 S:$G(GLB)="" GLB="^ZZRG"
 D READPKGS^ZZRGND19(GLB,0)
 D RGDEP(GLB)
 D WAPI(GLB)
 Q
 ;
WAPI(GL) ;
 N PKG,I,NAME,RTN,TAG
 S PKG="",I=0,RTN="",TAG=""
 R !,"ROUTINE TO BE INDEXED //All: ",RTN
 R:RTN'="" !,"TAG TO BE INDEXED //All: ",TAG
 I RTN="" D  Q
 . F  S PKG=$O(@GL@(4,PKG)) Q:PKG=""  S I=I+1 D
 . . W !,"-----------------------------------------------------------------
",!
 . . S NAME=$$GPKGNAME^ZZRGND19(PKG,GL)
 . . I @GL@(4,PKG)>40 W !,I,". COMMON SERVICE NAME: "_NAME I 1
 . . E  W !,I,". PACKAGE NAME: "_NAME
 . . I '$D(@GL@(7,PKG)) W !!,"Not used by other packages",! I 1
 . . E  D RPCETAG^ZZRGND16(PKG,GL),WPKGAPI(GL,PKG,I,"","")
 . . W !,"-----------------------------------------------------------------
",!
 . W !
 S PKG=$$GETPKG^ZZRGND19(RTN,GL)
 D WPKGAPI(GL,PKG,1,RTN,TAG)
 Q
 ;
WPKGAPI(GL,PKG,I,RTN,TAG) ;
 N P,SRCS
 S TAG=$G(TAG)
 I RTN="" D  Q
 . F  S RTN=$O(@GL@(7,PKG,RTN)) Q:RTN=""  D
 . . S TAG=""
 . . F  S TAG=$O(@GL@(7,PKG,RTN,TAG)) Q:TAG=""  D WRTNAPI(GL,PKG,RTN,TAG)
 I TAG'="" D WRTNAPI(GL,PKG,RTN,TAG) Q
 F  S TAG=$O(@GL@(7,PKG,RTN,TAG)) Q:TAG=""  D WRTNAPI(GL,PKG,RTN,TAG)
 Q
```

```
 ;
WRTNAPI(GL,PKG,RTN,TAG)
 W !!," "_TAG_"^"_RTN
 D REPORTSR(GL_"(9,"""_PKG_""","""_RTN_""","""_TAG_""")"," CALLING
PACKAGES : ",GL)
 D REPORTRP(TAG_"^"_RTN," CALLING RPC's : ")
 D REPOROPT(TAG_"^"_RTN," CALLING OPTIONS : ")
 D REPORTGL(" FORMAL:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""F"")",0)
 D REPORTGL(" INPUT:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""I"")",1)
 D REPORTGL(" OUTPUT:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""O"")",1)
 D REPORTGL(" GLBS:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""G"")",0)
 D REPGLPKG(" GLS MDEP:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""G"")",PKG,GL)
 D REPORTGL(" FM GLBS:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""FMG"")",0)
 D REPGLPKG(" FM G MDEP:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""FMG"")",PKG,GL)
 D REPORTCMD(" READ:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""CMD"")","R")
 D REPORTCMD(" WRITE:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""CMD"")","W")
 W !
 D REPORTGL(" PKG GLBS:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""GL"")",0)
 D REPORTCMD(" PKG EXEC:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""CMD"")","X")
 D REPORTCMD(" PKG IND:
",GL_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""CMD"")","@")
 Q
 ;
```

Example

```
GETFLDS^GMPLEDT3
  CALLING PACKAGES : ORDER ENTRY/RESULTS REPORTING,CLINICAL CASE
                     REGISTRIES
  CALLING RPC's    : --
  CALLING OPTIONS  : --
  FORMAL: DA
  INPUT: DA*,DUZ*,GMPLMGR,GMPORIG*,GMPROV,GMPVAMC,U
  OUTPUT: %,C,DA*,DI,DIL,DIQ0,DIQ1,DIQ2,DRS*,GMPFLD*,GMPORIG*,J,X,Y
  GLBS: ^AUPNPROB(,^DD(,^DI(,^DIC(
  GLS MDEP: --
  FM GLBS: ^AUPNPROB("
  FM G MDEP: --
  READ: 0
  WRITE: 0

  PKG GLBS: ^AUPNPROB(
  PKG EXEC: 0
  PKG IND: 0
```

3.3.REPORTAPIO^ZZRGND13

This procedure lists the external tags that are called by a particular package.

Code

```
REPORTAPIO(GLB) ;
 S:$G(GLB)="" GLB="^ZZRG"
 D READPKGS^ZZRGND19(GLB,0)
 D WAPIO(GLB)
 Q
 ;
WAPIO(GL) ;
 N PKG,I,NAME,RTN,TAG
 S PKG="",I=0,RTN=""
 F  S PKG=$O(@GL@(8,PKG)) Q:PKG=""  S I=I+1 D
 . W !,"------------------------------------------------------------------",!
 . S NAME=$$GPKGNAME^ZZRGND19(PKG,GL)
 . I @GL@(4,PKG)>40 W !,I,". COMMON SERVICE NAME: "_NAME,! I 1
 . E  W !,I,". PACKAGE NAME: "_NAME,!
 . D WPKGAPIO(GL,PKG,I)
 . W !,"------------------------------------------------------------------",!
 W !
 Q
 ;
WPKGAPIO(GL,SPKG,I) ;
 N RTN,TAG,P,SRCS,PKG,ATAG
 S PKG=""
 F  S PKG=$O(@GL@(8,SPKG,PKG)) Q:PKG=""  D
 . S RTN=""
 . F  S RTN=$O(@GL@(8,SPKG,PKG,RTN)) Q:RTN=""  D
 . . S ATAG=""
 . . F  S ATAG=$O(@GL@(8,SPKG,PKG,RTN,ATAG)) Q:ATAG=""  D
 . . . W !," "_ATAG_"^"_RTN_" ("_$$GPKGNAME^ZZRGND19(PKG,GL)_")"
 Q
 ;
```

Example

```
26. PACKAGE NAME: PROBLEM LIST

 $$GETSTAT^DGMSTAPI (REGISTRATION)
 $$GETCUR^DGNTAPI (REGISTRATION)
 7^VADPT (REGISTRATION)
 DEM^VADPT (REGISTRATION)
 OERR^VADPT (REGISTRATION)
 $$SITE^VASITE (REGISTRATION)
 CREIXN^DDMOD (VA FILEMAN)
 WAIT^DICD (VA FILEMAN)
 FILE^DICN (VA FILEMAN)
 YN^DICN (VA FILEMAN)
 DQ^DICQ (VA FILEMAN)
 $$GET1^DID (VA FILEMAN)
 IX1^DIK (VA FILEMAN)
 $$GET1^DIQ (VA FILEMAN)
```

```
EN^DIQ1 (VA FILEMAN)
BREAK^GMTSUP (HEALTH SUMMARY)
CKP^GMTSUP (HEALTH SUMMARY)
…
```

3.4.USES^ZZRGND15(GL,PKG)

This procedure prints all globals used by routines in a specific package as well as the file name and the package that owns that file. It depends on the output from `CRTBASE^ZZRGND13`.

Code

```
USES(GLB,PKG) ; Prints globals used by a package and their owner
 S:$G(GLB)="" GLB="^ZZRG"
 N GPKG,SUB,RTN
 D RDOWNER^ZZRGND19(GLB,0)
 W !,"Directly Used Globals By ",$$GPKGNAME^ZZRGND19(PKG,GLB),!
 S RTN=""
 F  S RTN=$O(@GLB@(1,RTN)) Q:RTN=""  D
 . Q:$$GETPKG^ZZRGND19(RTN,GLB)'=PKG
 . S G=""
 . F  S G=$O(@GLB@(1,RTN,"G",G)) Q:G=""  D USESGLB(GLB,PKG,RTN,G)
 W !,"Globals used through FileMan calls by
",$$GPKGNAME^ZZRGND19(PKG,GLB),!
 Q
 ;
USESGLB(GL,PKG,RTN,G)
 N GINFO,SUB,TAG
 Q:$E(G,1,4)="^TMP"
 Q:$E(G,1,8)="^UTILITY"
 Q:$E(G,1,6)="^%ZOSF"
 S GINFO=$$GLBINFO(GL,G)
 W !,"Routine ",RTN,": ",G," "
 W:PKG'=$P(GINFO,"^",1) "(Dependency to ",$P(GINFO,"^",2),",
",$P(GINFO,"^",3)," File)"
 Q
 ;
GLBINFO(GL,G)
 N GINFO,GLB,SUB
 S GLB=$P(G,"(",1)
 S SUB=$P($P(G,"(",2),",",1)
 S GINFO=""
 S:SUB'="" GINFO=$G(@GL@(20,GLB,SUB))
 S:GINFO="" GINFO=$G(@GL@(20,GLB))
 Q GINFO
 ;
```

Example

```
Directly Used Globals By PROBLEM LIST

Routine GMPL: ^AUPNPROB(
Routine GMPL: ^GMPL(125
Routine GMPL: ^VA(200 (Dependency to KERNEL, NEW PERSON File)
```

```
   Routine GMPL1: ^AUPNPROB(
   Routine GMPL1: ^ICD9("AB" (Dependency to DRG GROUPER, ICD
DIAGNOSIS File)
   Routine GMPL1: ^XUSEC("GMPL ICD CODE" (Dependency to Kernel,
AUDIT LOG FOR OPTIONS File)
   …
```

3.5.USED^ZZRGND15(GL,PKG)

This tag prints a list of all packages that use globals owned by the PKG package. It depends on the output from `CRTBASE^ZZRGND14`.

Code

```
USED(GLB,PKG) ; Prints packages that use globals owned by PKG
 N GINFO
 S RTN="",CNT=0
 F  S RTN=$O(@GLB@(1,RTN)) Q:RTN=""  D
 . Q:$$GETPKG^ZZRGND19(RTN,GLB)=PKG
 . S G=""
 . F  S G=$O(@GLB@(1,RTN,"G",G)) Q:G=""  D
 . . S GINFO=$$GLBINFO(GLB,G)
 . . Q:$P(GINFO,"^",1)'=PKG
 . . W !,"Routine ",RTN,": ",G, "
(",$$GPKGNAME^ZZRGND19($$GETPKG^ZZRGND19(RTN,GLB),GLB),")"
 Q
 ;
```

Example

```
Routine ACKQUTL6: ^AUPNPROB( (QUASAR)
Routine ACKQUTL6: ^AUPNPROB("AC" (QUASAR)
Routine ACKQUTL6: ^AUPNPROB("B" (QUASAR)
Routine GMTSPLSZ: ^AUPNPROB( (HEALTH SUMMARY)
Routine GMTSPLSZ: ^GMPL(125.99 (HEALTH SUMMARY)
Routine IBDFBK3: ^AUPNPROB( (AUTOMATED INFO COLLECTION SYS)
Routine IBDFN11: ^AUPNPROB( (AUTOMATED INFO COLLECTION SYS)
Routine LEXLGM3: ^AUPNPROB( (LEXICON UTILITY)
Routine LEXLGM3: ^AUPNPROB(0 (LEXICON UTILITY)
Routine LEXPL: ^AUPNPROB( (LEXICON UTILITY)
Routine LEXPLEM: ^AUPNPROB( (LEXICON UTILITY)
…
```

3.6.RFMCALLS^ZZRGND17(GLB)

This procedure lists all FileMan calls done by a particular M procedure grouped by package, as well as the globals read or written with this calls.

Code

```
RFMCALLS(GLB) ; Prints FileMan calls per M TAG and globals used by these
calls
 N PKG,RTN,TAG,LN
 S:$G(GLB)="" GLB="^ZZRG"
 S PKG="",LN=0
 F  S PKG=$O(@GLB@(7,PKG)) Q:PKG=""  S LN=LN+1 D
 . W !,"-------------------------------------------------------------------",!
 . W !,LN,". PACKAGE NAME: "_$$GPKGNAME^ZZRGND19(PKG,GLB)
 . S RTN=""
 . F  S RTN=$O(@GLB@(7,PKG,RTN)) Q:RTN=""  D
 . . S TAG=""
 . . F  S TAG=$O(@GLB@(7,PKG,RTN,TAG)) Q:TAG=""  D
 . . . Q:'$D(@GLB@(7,PKG,RTN,TAG,"FMG"))
 . . . W !!," "_TAG_"^"_RTN
 . . . D REPORTGL^ZZRGND13(" Globals:
",GLB_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""FMG"")",0)
 . . . D REPORTGL^ZZRGND13(" FileMan calls:
",GLB_"(7,"""_PKG_""","""_RTN_""","""_TAG_""",""FMGC"")",0)
 . W !,"-------------------------------------------------------------------",!
 Q
 ;
```

Example

```
1. PACKAGE NAME: VOLUNTARY TIMEKEEPING

 VIEW^ABSV88B
  Globals: ^ABS(503330,
  FileMan calls: EN1^DIP

 DELETE^ABSVDENT
  Globals: ^ABS(503340,
  FileMan calls: ^DIC,^DIK

 NEW^ABSVDENT
  Globals: ^ABS(503334,;^ABS(503340
  FileMan calls: FILE^DICN,^DIC,^DIE

 VIEW^ABSVDENT
  Globals: ^ABS(503340,
  FileMan calls: EN^DIQ,^DIC

 DATE^ABSVDPNT
  Globals: ^ABS(503340,
  FileMan calls: EN1^DIP

…
```

4. How to use the code

In order to run the reports, the first step is to call `MAIN^ZZRGND14`. This procedure will parse the codebase and will produce the indexed data the reports depend on. You can pass the name of a global to store this data or you can accept the default `^ZZRG`.

```
D MAIN^ZZRGND14()
```

Note

This procedure loads the routine list by Xecuting ^%ZOSF("RSEL") that will be ask for a list of routines. The prompt is different depending on you host.

On Cache installation please select:

```
All Routines? No => Yes
```

On GT.M installations please enter:

```
Routine: *
```

The following reports are available:

- To list calls into packages grouped by M Tag, as well as formal and input parameters, globals used, number of reads and writes:

  ```
  D REPORTAPI^ZZRGND13
  ```

- To list calls made by routines into other packages:

  ```
  D REPORTAPIO^ZZRGND13
  ```

- To list globals owned by other packages that are used by a routine:

  ```
  D USES^ZZRGND15
  ```

- To list procedures that use globals owned by a specific package:

  ```
  D USED^ZZRGND15
  ```

- To list calls to FileMan procedures made by a procedure:

  ```
  D RFMCALLS^ZZRGND17
  ```

- To list all RPC Broker tags, their description, input parameters and return values:

  ```
  D REPORTRPC^ZZRGND13
  ```

- To list all OPTION tags available and the procedures they are calling:

```
D REPOPTAG^ZZRGND13
```

Note

When calling these procedures you will be asked to provide the path to `Packages.csv` and `Ownership.csv`. These files are included in the tool package. The former lists package names and their assigned namespaces, the latter contains a mapping between globals and the packages that own them. When asked, please enter the full path name to the location where they are stored on disk (e.g. `C:\M-RoutineAnalyzer\Packages.csv`)

5. Conclusions

This paper presented a set of utility procedures that can be used during the VistA code refactoring effort. They will be useful in the code analysis phase to find the dependencies between packages so we can get a better idea of what has to be done to make the code more modular.
Another useful information is the list of globals that should be used through the Database Access layer. Last but not least the described reports give us a view of how local variables are used in code, whether they should be newed, initialized or killed by a specific routine.