IVS Working Group IV and the New Open Format Database

John Gipson

Abstract IVS Working Group IV was established to 'design a data structure that meets current and anticipated requirements for individual VLBI sessions including a cataloging, archiving, and distribution system. Further, it will prepare the transition capability through conversion of the current data structure as well as cataloging and archiving softwares to the new system'. Working Group IV successfully met these goals and was disbanded in March 2013. I describe the new VLBI data format, the use of the format by various software packages, and plans to transition to the new format.

Keywords Data structures, VLBI

1 Introduction and Brief History

At the 15 September 2007 IVS Directing Board meeting I proposed establishing a 'Working Group on VLBI Data Structures'. This proposal was unanimously accepted, and the Board established IVS Working Group 4 on VLBI Data Strucures (IVS-WG4). The first meeting of IVS-WG4 was at the IVS General Meeting in St. Petersburg in 2008 [1]. In 2009 I circulated a draft proposal within IVS-WG4, and this was presented to the wider VLBI community at the 2010 General Meeting in Hobart [2]. At that meeting I also solicited input on naming the data structure. At the IVS GM in Madrid [3] I presented a progress report and announced that the new structure would be called 'openDB'. The name acknowledges the long history of Mark III databases, and emphasizes the open nature of the new structure. The final report of IVS-WG4 was presented to the IVS Directing Board in March 2013. With its acceptance WG4 was disbanded. Since it turned out that the name 'openDB' is already taken, the format was renamed to vgosDB. A slightly modified version of the final report reflecting the name change was published in the 2013 IVS Annual Report [4].

It often happens that something that makes sense in theory does not work well in practice. To ensure that this did not happen, Calc/Solve and VieVS were modified to use 'draft' versions of the proposed structure. In 2010 I wrote a utility to convert a subset (the data contained in NGS format) of the data in a Mark III-database (MK3-DB) to the new format. This subset was chosen because many software packages use NGS format as input. VieVS was successfully modified to use this format. Solve was also modified to take some input from the new format, taking the rest of its input from Solve 'superfiles'. The original utility was expanded to convert more of the data in MK3-DB to vgosDB format, and by 2012 the vgosDB format could be used as a replacement for Solve 'superfiles'. Timing tests (see Figure 1) showed that the new format is faster for large sessions. Concurrently, nuSolve was modified to be able to read and write the new format. In the remainder of 2012 and the beginning of 2013 programs were written to handle other stages in the processing chain. By June 2013 it was possible to process a VLBI session in the vgosDB format, starting at correlator output and ending with a 'version 4' vgosDB where the ambiguities were resolved and the data edited. The first public release of Calc/Solve which used vgosDB was made in February 2014.

NVI, Inc.

Alpha-testing of the new format using Calc/Solve, nuSolve, and VieVS was very important. Although there were no fundamental changes in the overall scheme, there were many incremental changes that improved the final product. For example: we changed variable names to make them more consistent; we modified which variables were stored in which files to reflect our experience in processing the data (with the original specification, the standard processing would have required updating some files—something we wanted to avoid if possible); we made some files more self-contained, and we also reduced redundancy for variables which are sometimes constant for a session (e.g., ambiguity spacing) by introducing the 'Repeat' attribute.

The remainder of this note describes vgosDB and concludes with a transition plan. In a note of this size it is impossible to do more than present an overview of the vgosDB format. A fuller description of an earlier version of vgosDB can be found in [5]. This documentation is in the process of being updated to reflect the final version and should be available via anonymous ftp at the same location later in 2014.

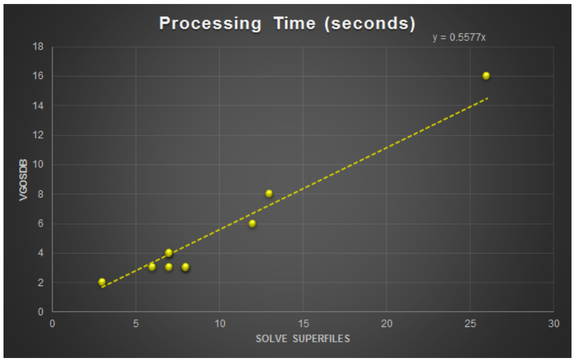

Fig. 1 Large sessions run much faster in the vgosDB format.

2 Design Goals

Table 1 summarizes some of the important design goals of IVS-WG4 and how they were achieved. We divide the goals into two classes: format is related to how the data is stored and structure to how the data is organized. This is an important distinction, because these are independent. For example, T. Hobiger [6] wrote routines which store the information contained in a MK3-DB in netCDF files while preserving the original organization of the data. This format would have met most of the goals in the top of the table, but not those in the bottom.

Table 1 Design Goals.

Goal	Format	Structure	How
Low Level Goals			
Reduce Redundancy		X	Scope
Ease of Access	X		NetCDF
Speed of Access	X		NetCDF
Open format	X		NetCDF
Many Languages, Platforms	X		NetCDF
High Level Goals			
Flexibility		X	Wrappers Separation
Exchange subsets of data	X	X	Separation
Separate observables, models, theoreticals		X	Separation
Data at different levels of abstraction		X	Completeness, Separation

Scope is how broadly applicable the data is. The Mark III databases (MK3-DB) recognize two scopes: 1) session-dependent data is valid for the entire session (e.g., station positions and names) and is stored in type 1 lcodes, and 2) observation-dependent data is valid for a given observation and is stored in type 2 and type 3 lcodes. Everything that is not session-dependent is automatically observation-dependent. This results in great redundancy, since some items, such as station meteorological data are really station/scan-dependent (that is, are constant for a given station in a scan), and not observation-dependent. For an N-station scan, this information is currently stored N–1 times in the MK3-DB and NGS format. vgosDB enlarges the concept of scope to include the following additional categories: 3) scan-dependent, where the data depends only on the scan (e.g., EOP) and 4) station/scan-dependent (Met data, cable-cal), where the data depends only on the station and a scan. Separating data by scope reduces redundancy at the cost of extra book-keeping.

NetCDF is a commonly used binary format used to store scientific data. Figure 2 is a schematic depiction of a netCDF file. NetCDF was designed for fast data access. It is open-source, and there are interfaces for most common computer languages (e.g., C, C++, FORTRAN, Matlab, Java) and most common operating sys-

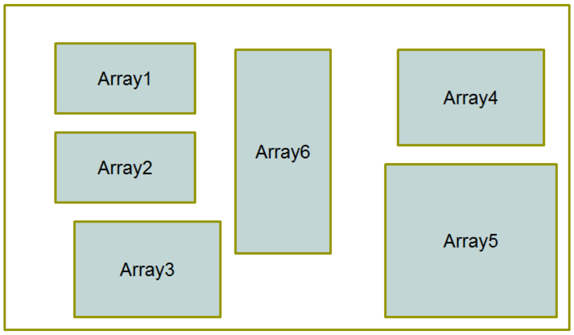

Fig. 2 A netCDF file can be viewed as a container for arrays.

tems (Linux, Windows, Mac OS). There are other data-storage formats with similar characteristics, and there are utilities for converting from one format to another. We chose netCDF because several members of IVS-WG4 had prior experience with it. All data in vgosDB is stored in either netCDF or ASCII files. This ensures that it can be read by anyone.

Separation is the concept that data should be split up depending on its origin and use. This is in contrast to MK3-DB and NGS format where in principle all of the data required to analyze a session is stored in a single file. In contrast, in the vgosDB format, observables (which should never be modified) are kept separate from theoreticals, partials, and the results of derived quantities such as ionosphere corrections.

An advantage of Separation is that it allows you to do a partial update of the data, for example, using alternative met-data, without updating all of the files. This is in contrast to the MK3-DB format where changing a single variable requires an entirely new database.

Completeness is the concept that we should include sufficient data so that the the complete processing chain can be redone from scratch. The proposed format does allow for this to happen, but this would require archiving the raw correlator output. In the current implementation we do not go quite this far, but we do include all data present in Version 1 MK3-DB. This allows other software packages besides Calc/Solve to do data editing and ambiguity resolution. Because many session log-files from 1990 onwards are available at the IVS Data Centers, other packages can also add cable-calibration and meteorological data.

Wrappers are special ASCII files that organize the data. These are necessary because, in contrast to MK3-DB or NGS format where all of the data is one file,

```
Begin History
Begin Program Calc/Solve Processing
Version   Mixed
CreatedBy John M. Gipson
Default_dir History
RunTimeTag 2014/04/21 17:21:51
History    10JAN04XU_kMK3DB_V005.hist
End Program Calc/Solve Processing
...
End History
!
Begin Session
Session I10004
AltSessionId 10JAN04XU
Head.nc
Default_Dir Apriori
Eccentricity.nc
Antenna.nc
Station.nc
Source.nc
Default_Dir CrossReference
StationCrossRef.nc
SourceCrossRef.nc
End Session
!
Begin Station KOKEE
Default_Dir KOKEE
TimeUTC.nc
Met.nc
AzEl_V005.nc
End Station KOKEE
!  ... Wettzell omitted
Begin Observation
!
Default_Dir Observables
TimeUTC.nc
DataFlag_bS.nc
DataFlag_bX.nc
AmbigSize_bS.nc
AmbigSize_bX.nc
Baseline.nc
GroupDelay_bS.nc
Source.nc
GroupDelay_bX.nc
Default_Dir ObsEdit
NumGroupAmbig_bX.nc
NumGroupAmbig_bS.nc
GroupDelayFull_bX.nc
GroupDelayFull_bS.nc

Default_Dir CrossReference
ObsCrossRef.nc
Default_Dir ObsDerived
EffFreq_bS.nc
EffFreq_bX.nc
End Observation
```

Fig. 3 Wrapper grammar is simple and human-readable.

vgosDB splits the data into many files. The wrapper contains pointers to all of the different pieces.

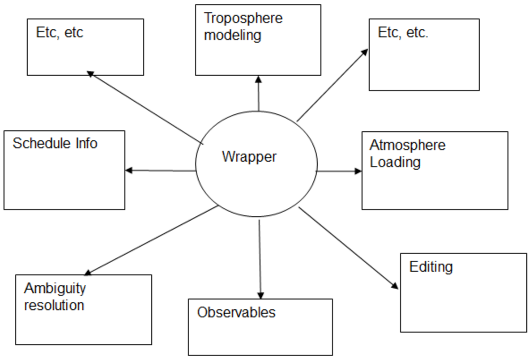

Fig. 4 A wrapper organizes vgosDB data.

3 Organization

In this section, we give an overview of how the vgosDB data is organized.

3.1 Directories

vgosDB sessions are organized by year and then by session. The name of the session directory is the same as the associated MK3-DB name. For example, the directory *vgosDB*/2010/10*JAN*04*XA* and its sub-directories contains all of the data for 10JAN04XA. There is one sub-directory for each station (e.g., *KOKEE*) which contains all of the station/scan-dependent data. The *Session* and *Scan* sub-directories contain session-dependent and scan-dependent data, respectively. There are several observation-dependent directories containing different types of data: *Observables* contains observables; *ObsEdit* contains the data editing flags; *ObsCal* and *ObsPart* contain calibration and partial information; and so on. The *History* sub-directory contains ASCII files summarizing the history of the experiment. In addition there is a provision for software specific directories, e.g., *Solve* or *VieVS*, which contain information of interest only to a particular analysis package.

3.2 vgosDB Variable

A vgosDB variable is an array of data together with associated meta-data. Roughly speaking, a variable corresponds to an MK3-DB lcode. The meta-data is stored using netCDF attributes, some examples of which are shown in Table 2. Each variable name is almost unique: it is permissible to use the same name if the variables differ in Station, Band or Kind. (Kind is described more fully below.) For example, the temperature data at all stations is called TempC. The X and S band group delay are both called GroupDelay but reside in different files.

Table 2 Some vgosDB variable attributes.

Name	Comment
Definition	Required
Units	Required if appropriate
Origin	Optional
Repeat	Indicates repeating value
Lcode	Optional
CreateTime	Optional

3.3 Time Dependence

There are two kinds of time dependence for vgosDB variables. If the time dependence is explicit, then the time information must be contained in the same file as the variable. If it is implicit, then the time dependence is contained in external files called *TimeUTC.nc*. This file contains two arrays: 1) YMDHM is an integer array that contains the Year, Month, Day, Hour and Minute part of the time-tag and 2) Second is a double precision array that contains the seconds part of the time-tag. Each station directory contains a *TimeUTC.nc*. The files *Scan/TimeUTC.nc* and *Obs/TimeUTC.nc* contain the time tags for the scan-dependent and the observation-dependent data, respectively.

3.4 Files

Related vgosDB variables are stored together in NetCDF files. These files have header information which provides information about when the files were made, their contents, and some information used for consistency checks. An example is given in Table 3.

Table 3 Sample vgosDB header information.

Name	Example
Stub	Cal-Cable
CreateTime	2013/06/27 17:56:41
CreatedBy	John Gipson NVI, Inc.
Program	db2vgosDB 2014Feb04
Subroutine	write_station_file 2012Dec11
DataOrigin	Data extracted from i1004kk.log
TimeTag	StationScan
TimeTagFile	TimeUTC.nc
Session	I1004
Station	KOKEE

Data is grouped together depending on scope, origin, how the data is used, and how the data is processed. One guiding principle in deciding where to put the data was that in routine data processing, although it may be necessary to make a new NetCDF file, you should avoid having to update an existing one. Most of the vgosDB files contain only a few vgosDB variables. For example, the meteorology data for each station usually comes from an on-site sensor. This data is stored in the file Met.nc in the appropriate station directory. Met.nc files contain pressure, temperature, and humidity of a site. The GroupDelay_bX.nc file contains the measured group delay and sigma. (A notable exception to the rule that 'each file contains only a few variables' is the CorrInfo*.nc file which contains the output of the fringing process and has $\sim$ 100 variables, most of which are seldom or rarely used.)

Table 4 vgosDB file-naming conventions.

<table>
<tr><td colspan="3">Stub_kAAAA_vBBBB_iCCCC_bDD.nc
Fields are separated by _
Field type indicated by k, v, i, b
Field length is arbitary</td></tr>
<tr><td>Part</td><td>Field</td><td>Comment</td></tr>
<tr><td>Stub</td><td>First</td><td>Specifies type of data</td></tr>
<tr><td>_k</td><td>Kind</td><td>e.g., NMF, VMF, ...</td></tr>
<tr><td>_v</td><td>Version</td><td>Arbitrary version control</td></tr>
<tr><td>_i</td><td>Institution</td><td>Individual/Institution</td></tr>
<tr><td>_b</td><td>Band</td><td>e.g., X, S, Ku..</td></tr>
<tr><td colspan="3">Examples:
Met_kNMF.nc
Cal-Cable.nc
Part-NutationEQX_kIAU2000
GroupDelay_bX.nc</td></tr>
</table>

Table 4 summarizes the filenaming convention. The first part of the file-name is the *stub* and completely specifies the type of data in the file. Files with the same stub are 'plug compatible', i.e., you can replace one file with another and the analysis software will continue to work correctly. The *kind* field differentiates between different functionally equivalent models, e.g., mapping functions such as VMF1 and GMF.

4 Transition Plans and Next Steps

The first Calc/Solve release, using vgosDB, was made in February 2014. This release can use vgosDB to replace superfiles. Also, nuSolve, which was released at the same time, can read and write the vgosDB format. VieVS is able to use vgosDB instead of NGS cards. The Goddard VLBI group has developed programs to create vgosDB files from correlator output and to use these files in all other stages in the processing chain.

Currently the Calc/Solve Analysis Centers (ACs) are responsible for producing the 'Version 4' databases which serve as the basis for all VLBI analysis. (Other software packages use NGS cards which are derived from the MK3-DB.) In 2014 the Calc/Solve ACs will transition to producing vgosDB files and submitting them to the IVS Data Center. Until this transition is complete the Goddard VLBI group will make available all IVS sessions in vgosDB format. By the end of 2014 the Calc/Solve ACs will stop producing MK3-DB, and we will fully transition to the new format.

References

1. J. Gipson, IVS Working Group 4 on VLBI Data Structures, in IVS 2008 General Meeting Proceedings, pp. 143–152.
2. J. Gipson, IVS Working Group 4: VLBI Data Structures, in IVS 2010 General Meeting Proceedings, NASA/CP-2010-215864, pp. 187–191.
3. J. Gipson, IVS Working Group 4: VLBI Data Structures, in IVS 2012 General Meeting Proceedings, NASA/CP-2012-217504, pp. 212–221.
4. J. Gipson et al., Final Report of IVS Working Group 4 on Data Structures, in IVS 2013 Annual Report, in press.
5. J. Gipson, OpenDB Manual, ftp::/gemini.gsfc.nasa.gov/pub/misc/jmg/VLBI_Structure_2013Jun11.pdf
6. T. Hobiger, Y. Koyama, T. Condo, MK3TOOLS and NetCDF-Storing VLBI data in a machine independent oriented data format, Proceedings 2007 EVGA Meeting.