John Gipson: IVS Working Group 4: VLBI Data Structures, IVS 2012 General Meeting Proceedings, p.212–221
http://ivscc.gsfc.nasa.gov/publications/gm2012/gipson.pdf

IVS Working Group 4: VLBI Data Structures

John Gipson

NVI, Inc., Code 698, NASA Goddard Space Flight Center, Greenbelt, MD, 20771
e-mail: john.m.gipson@nasa.gov

Abstract

I present an overview of the "openDB format" for storing, archiving, and processing VLBI data. In this scheme, most VLBI data is stored in NetCDF files. NetCDF has the advantage that there are interfaces to most common computer languages including Fortran, Fortran-90, C, C++, Perl, etc, and the most common operating systems including Linux, Windows, and Mac. The data files for a particular session are organized by special ASCII "wrapper" files which contain pointers to the data files. This allows great flexibility in the processing and analysis of VLBI data. For example it allows you to easily change subsets of the data used in the analysis such as troposphere modeling, ionospheric calibration, editing, and ambiguity resolution. It also allows for extending the types of data used, e.g., source maps. I present a roadmap to transition to this new format. The new format can already be used by *VieVS* and by the *global* mode of *solve*. There are plans in work for other software packages to be able to use the new format.

1. Introduction

At the 15 September 2007 IVS Directing Board meeting I proposed establishing a "Working Group on VLBI Data Structures". The thrust of the presentation was that, although the VLBI database system has served us very well these last 30 years, it is time for a new data structure that is more modern, flexible, and extensible. This proposal was unanimously accepted, and the Board established IVS Working Group 4 (WG4). Quoting from the IVS Web site [1]: *"The Working Group will examine the data structure currently used in VLBI data processing and investigate what data structure is likely to be needed in the future. It will 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."*

Table 1. Original membership in Working Group 4.

Chair	John Gipson
Analysis Coordinator	Axel Nothnagel
Correlator Representative	Roger Cappallo
GSFC/Calc/Solve	David Gordon Leonid Petrov
JPL/Modest	Chris Jacobs Ojars Sovers
Occam	Oleg Titov Volker Tesmer
TU Vienna	Johannes Boehm
IAA	Sergey Kurdobov
Steelbreeze/MAO	Sergei Bolotin
Observatoire de Paris/PIVEX	Anne-Marie Gontier
NICT	Thomas Hobiger Hiroshi Takiguchi

Any change to the VLBI data format affects everyone in the VLBI community. Therefore, it is important that the working group have representatives from a broad cross-section of the IVS community. Table 1 lists the original members of WG4 together with their affiliation at that time. The initial membership was developed in consultation with the IVS Directing Board. On the one hand, we wanted to ensure that all points of view were represented. On the other hand, we wanted to make sure that the size did not make WG4 unwieldy. The composition and size of WG4 is a reasonable compromise between these two goals. My initial request for participation in WG4 was enthusiastic: everyone I contacted agreed to participate with the exception of an individual who declined because of retirement.

Since the formation of Working Group 4 there have been several changes. I am sad to note the passing of Anne-Marie Gontier. Leonid Petrov has also left the Goddard VLBI group and is no longer active in WG4. The work of both of these scientists served as inspiration for initial discussions of WG4, and many of their ideas live on in the work of this group. The affiliation of other scientists has changed, and some have retired.

2. History of Working Group 4

WG4 held its first meeting at the IVS 2008 General Meeting in St. Petersburg, Russsia. This meeting was open to the general IVS community. Roughly 25 scientists attended: ten WG4 members and 15 others. This meeting was held after a long day of proceedings. The number of participants and the lively discussion that ensued is strong evidence of the interest in this subject. A set of design goals, displayed in Table 2, emerged from this discussion. In some sense the design goals imply a combination and extension of the current VLBI databases, the information contained on the IVS session Web pages, and much more information [2].

During the next year the working group communicated via email and telecon and discussed how to meet the goals that emerged from the St. Petersburg meeting. A consensus began to emerge about how to achieve most of these goals.

The next face-to-face meeting of WG4 was held at the 2009 EVGA meeting in Bordeaux, France. This meeting was also open to the IVS community. At this meeting a proposal was put forward to split the data contained in the current Mark III databases (MK3-db) into smaller files which are organized by a special ASCII file called a wrapper. Some of the characteristics and advantages of this approach were summarized. Overall the reaction was positive.

In the summer of 2009 I worked on elaborating these ideas, and in July I circulated a draft proposal to Working Group 4 members. At the same time I began a partial implementation of these ideas and wrote software to convert a subset of the data in a Mark III database into the new format. This particular subset included all data in NGS cards and a little more. The subset was chosen because many VLBI analysis packages including *Occam*, *Steelbreeze*, and *VieVS* can use NGS cards as input. In August 2009 three VLBI sessions in the new format were made available via anonymous FTP: an Intensive, an R1, and an RDV.

Andrea Pany of the Technical University of Vienna developed an interface to *VieVS* working with the draft proposal. This work was later modified by Matthias Madzak. During this process the definition of a few of the data items needed to be clarified, which emphasizes the importance of working with the data hands on. At NASA's Goddard Space Flight Center, Sergei Bolotin interfaced a variant of this format to *Steelbreeze*. *Steelbreeze* uses its own proprietary format, and one motivation for interfacing to the new format was to see if there was a performance penalty

Table 2. Key Goals of New Format.

Goal	Description
Provenance	Users should be able to determine the origin of the data and what was done to it.
Compactness	The data structure should minimize redundancy, and the storage format should emphasize compactness.
Speed	Commonly used data should be able to be retrieved quickly.
Plaform/OS/ Language Support	Data should be accessible by programs written in different languages running on a variety of computers and operating systems.
Extensible	It should be easy to add new data types.
Open	Data should be accessible without the need for proprietary software.
Decoupled	Different types of data should be separate from each other.
Multiple data levels	Data should be available at different levels of abstraction. For example, most users are interested only in the delay and rate observables. Specialists may be interested in correlator output.
Completeness	All VLBI data required to process (and understand) a VLBI session from start to finish should be available: schedule files, email, log-files, correlator output, and final 'database'.
Web Accesible	All data should be available via the Web

associated with using the new format. Bolotin found that there was a performance penalty of 40 μs of processing time per observation[1]. At that time there were about 6 million VLBI observations, which translates into an extra six minutes to process all of the VLBI data. This is a modest price to pay for the many advantages the format brings.

Most of the work of 2010-2012 has been devoted to implementing the ideas of the Working Group 4 and making them concrete. This has taken much longer than anticipated. There were two areas of emphasis.

1. Major efforts were devoted to completing the utility, *db2openDB* to convert from MK3-db to openDB format. Mark III databases contain over 400 different L-codes, and it was necessary to figure out what to do with each of them. In this process the L-codes were tabulated and circulated to members of the working group to solicit feedback. Many L-codes, such as the speed of light or the value of π, were discarded because they were no longer needed. Others were discarded because they are obsolete and not used. Most were kept. Development of this utility is a neccessary part of the transition to the openDB format and is explicitly mentioned in the terms of reference for WG4.
2. *Solve* was modified to use the openDB format as a replacement for superfiles. This was done in part to gain experience using the new format. However, as discussed in more detail later, any discussion of converting to openDB has to include a discussion of modifying *calc/solve* because all VLBI data is initially processed using *calc/solve*.

These major hurdles are now over, and we are optimistic that the remaining steps described at the end of this paper will occur in a timely fashion.

[1]No effort was made to optimize the interface. With optimization this figure should be less.

3. Overview of New Organization

3.1. Modularization

A solution to many of the design goals of Table 2 is to modularize the data, that is to break up the data associated with a session into smaller pieces. These smaller pieces are organized by 'type', that is the kind of data: group delay observable; meteorological data; editing criteria; station names; station positions; etc. In many, though not all, cases, each 'type' corresponds to a MK3-db L-code. Different data types are stored in different files, with generally only one or a few closely related data types in each file. For example, it might be convenient to store all of the meteorological data for a station together in a file. However, there is no compelling reason to store the meteorological data together with pointing information. Splitting the data in this way has numerous advantages, outlined below. The first three of these directly address the design goals. The last two were not originally specified but are consequences of this design decision.

1. **Separable.** Users can retrieve only that part of the data in which they are interested.
2. **Extensible.** As new data types become used, for example, source maps, they can be easily added without having to rewrite the whole scheme. All you need to do is specify a new data type and the file format.
3. **Decoupled.** Different kinds of data are separated from each other. Observables are separated from models. Data that won't change is separated from data that might change.
4. **Flexible.** Since data is kept in separate files, it is possible to keep and use several alternatives of the same data type. For example, you might have different sources of met data, or files that contain editing flags from different analysts or groups.
5. **Partial Data Update.** Instead of updating the entire database, as is currently done, you only need to update that part of the data that has changed[2].

Data is also organized by 'scope'. Scope is how broadly applicable the data is — whether it holds for the entire session, for a particular scan, for a particular scan and station, or for a particular observation. The current Mark III database is observation-oriented: all data required to process a given observation is stored once for each observation. This results in tremendous redundancy for some data. For example, in an N station scan, each station will participate in $N-1$ observations. Station dependent data, such as pointing data, or met-data, is stored once for each observation, instead of once for the scan. Organizing data by scope allows you to reduce redundancy.

3.2. NetCDF as Default Storage Format

Data storage is independent of data organization. WG4 reviewed a variety of formats including NetCDF, HCDF, CDF, and FITS . In some sense, all of these formats are equivalent—there exist utilities to convert from one format to another. Ultimately we decided to use NetCDF because it has a large user community, and because several members of the Working Group have experience with using NetCDF. At its most abstract, NetCDF is a means of storing arrays in files as pictured schematically in Figure 1. The arrays can be of different sizes and contain different kinds of data –

[2]This is done by making a new version of the relevant file, keeping the old one intact.

strings, integer, real, double, etc. Most VLBI data can be considered an array, which makes using NetCDF a natural choice. Since NetCDF files can contain ASCII strings it is easy to incorporate information about the provenance and processing history of the data.

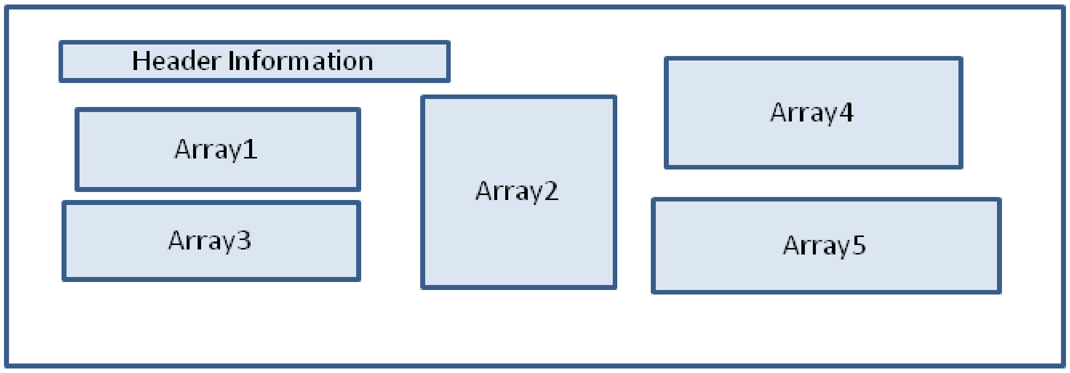

Figure 1. A NetCDF file is a container for arrays.

Storing data in NetCDF format has the following advantages, some of which are answers to the design goals from Table 2.

1. **Platform/OS/Language Support.** NetCDF has interface libraries to all commonly used computer languages running on a variety of platforms and operating systems.
2. **Speed.** NetCDF is designed to access data fast.
3. **Compactness.** The data is stored in binary format, and the overhead is low. This means that a NetCDF file is much smaller than an ASCII file storing the same information.
4. **Open.** NetCDF is an open standard, and software to read/write NetCDF files is freely available.
5. **Transportability.** NetCDF files use the same internal format regardless of the machine architecture. Access to the files is transparent and handled by libraries.
6. **Large User Community.** Because of the large user community, there are many tools developed to work with NetCDF files.

3.3. Organizing Data by Wrappers

The main disadvantage of breaking up the Mark III database into many smaller files is that you need some way of organizing the files. This is where the concept of a wrapper comes in. A wrapper is an ASCII file that contains pointers to VLBI files associated with a session. The wrapper concept is illustrated schematically in Figure 2. An appendix gives a concrete example of a wrapper for an IVS Intensive session. The wrapper can serve several purposes:

1. The wrapper can be used by analysis programs to specify what data to use.
2. The wrapper allows analysts to experiment with ‘what if’ scenarios—for example, to use alternative data editing. All you need to do is modify the wrapper to point to the alternate file, as illustrated in Figure 3.

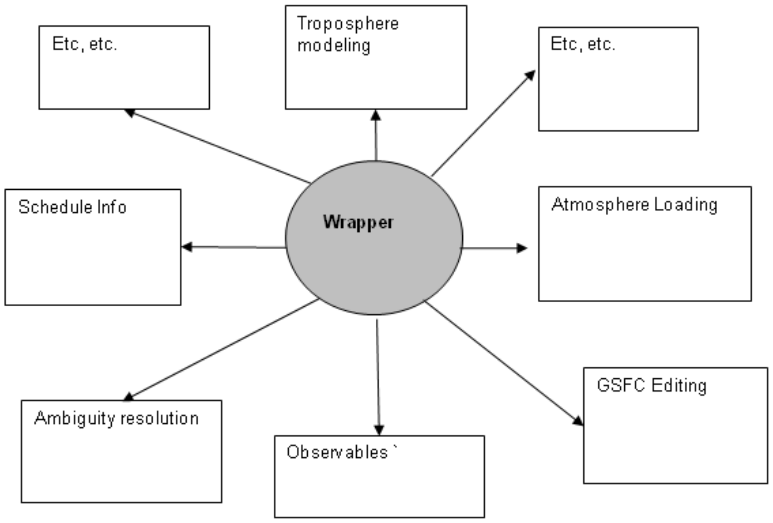

Figure 2. ASCII wrapper files organize the VLBI data and contain pointers to file locations.

3. Wrappers allow users to exchange subsets of the data—for example, editing criteria or meteorological data, or trophosphere modeling.

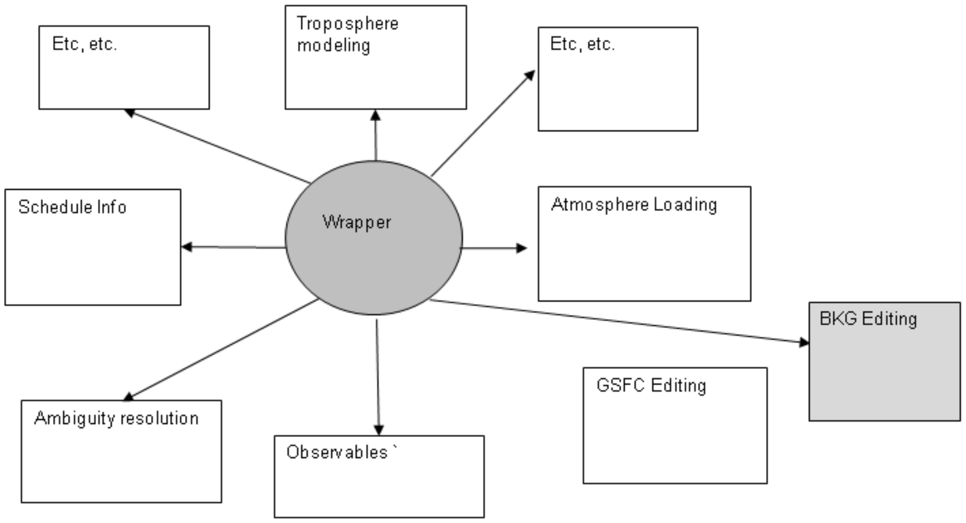

Figure 3. Wrappers allow you to explore 'what-if' scenarios by editing the wrapper file and pointing to alternative data-items.

4. Different analysis packages can define custom data-items and refer to them in wrappers.
5. If some of the VLBI data changes, you only have to update that part that has changed and write a new wrapper.

There are many other features of the new format which space prohibits from describing. All of these are discussed in the draft proposal, which has a complete specification of the wrapper grammar, together with a specification of the file format.

4. Transition to OpenDB Format and Calc/Solve Issues

Figure 4 shows the standard processing of VLBI data from the correlator to the version 4 database. The version 4 database, either in MK3-db format, or as a subset of the data exported to NGS cards, is the starting point of many VLBI analysis packages. Here are the steps involved:

1. *Dbedit* takes the individual correlator output fringe files (one for each observation and band) and knits them into a version 1 database for each band.

2. *Calc* computes the theoretical delay (used by many analysis packages) and other quantities primarily of interest to *solve* users (e.g., various partials), producing a version 2 database.

3. *Dbcal* extracts cable-cal and meteorological information from the log files and inserts it into the database producing a version 3 database.

4. *Interactive-Solve* combines the X- and S-band databases and allows the analyst to do data-editing, ambiguity resolution and ionosphere calibration. The result is saved as a version 4 database. The version 4 database is exported to an IVS Data Center where it is available for download.

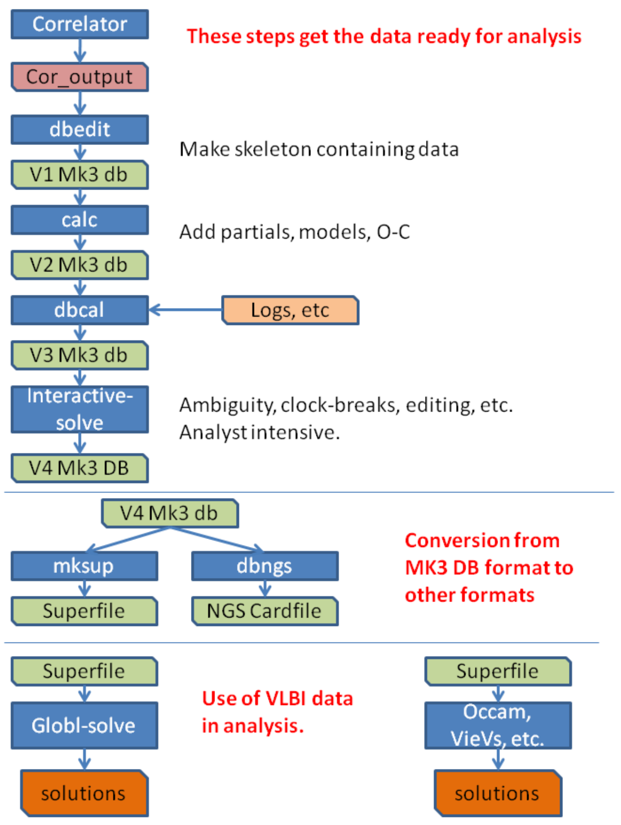
Correlator
These steps get the data ready for analysis
Cor_output
dbedit
Make skeleton containing data
V1 Mk3 db
calc
Add partials, models, O-C
V2 Mk3 db
dbcal
Logs, etc
V3 Mk3 db
Interactive-solve
Ambiguity, clock-breaks, editing, etc. Analyst intensive.
V4 Mk3 DB
V4 Mk3 db
mksup
dbngs
Conversion from MK3 DB format to other formats
Superfile
NGS Cardfile
Superfile
Use of VLBI data in analysis.
Superfile
Globl-solve
Occam, VieVs, etc.
solutions
solutions

Figure 4. Standard VLBI data path. The starting point for most VLBI analysis is version 4 databases which are converted to different formats depending on the analysis package.

At this point the data flow branches depending on the analysis software.

1. For *calc/solve* users, *mksup* converts a MK3-database into a 'superfile' — a special binary file containing much of the information in a MK3-database. This is used in *globl-solve*, which processes several databases jointly to obtain, for example, estimates of station position or source coordinates. This step is neccessary because the I/O involved in accessing the information in a MK3-database directly is prohibitive if you are going to be analyzing hundreds or thousands of sessions.

2. For users of other software packages, such as *Occam*, *Steelbreeze*, *VieVS*, etc., *dbngs* extracts a subset of the data contained in the MK3-database and writes it to an 'NGS cardfile' which is a special ASCII file. Note that since this is only a subset of the data you are limited in

what you can do. As a specific example, it is impossible to resolve ambiguities using only the data present in the NGS cards. Because of this most software packages take this data 'as-given.' If there are problems with the ambiguity resolution there is little you can do about it.

A key point of the above discussion is that to make a version 4 database, the starting point of many data-analysis packages, you need to process the VLBI data through *calc/solve*. Any discussion of converting to a new format of neccessity involves a discussion of modifying *calc/solve* to use the new format. This problem is made more difficult because *calc/solve* is an operational program which is central to the IVS data-flow. You have to keep the normal data-flow working while implementing an alternative datapath. The situation is very much like building a new bridge for a road. First you build a bridge in parallel to the original bridge. Then you route traffic over the new bridge. Finally you tear down the old bridge.

In our case there are several 'bridges' with each bridge corresponding to taking a database from one version to the next. Our approach is to start at the last stage of the data path and work backwards towards the correlator output as illustrated in Figure 5. Our first goal was to write a utility db2openDB to convert from MK3-format to the openDB format. This is a neccessary step in transitioning to the new format. Work began on this in July 2009 and was completed in July 2012. A beta version in openDB format of all sessions through 2011 is publicly available at `ftp://gemini.gsfc.nasa.gov/pub/openDB`.

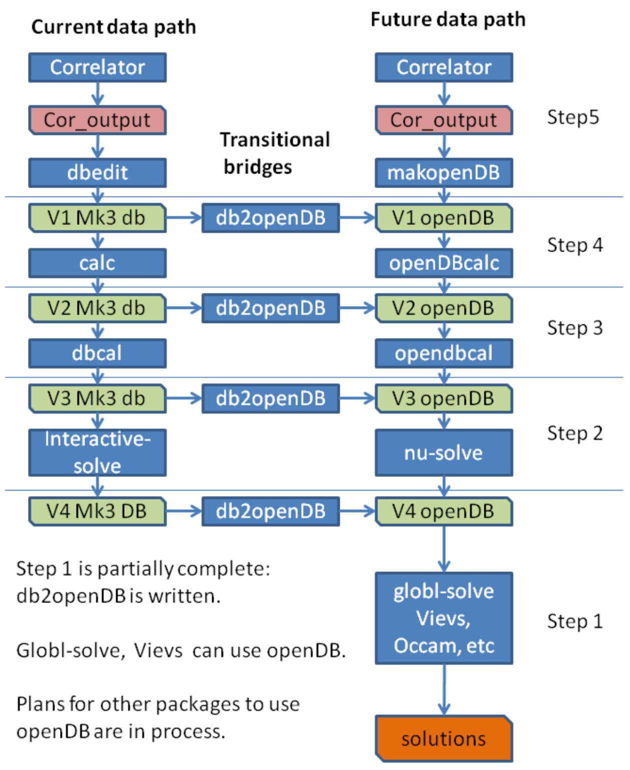

Figure 5. Data path of VLBI data with hybrid Mark III databases and openDB format. The goal is to transition from the left-hand-side to the right-hand-side of this chart by working upwards from the bottom.

While developing db2openDB the VLBI group at Goddard worked on modifying *globl-solve* to use openDB files instead of superfiles, and the VLBI group at the Technical University of Vienna modified *VieVS* to use openDB. Feedbacks from these efforts resulted in 'tweaking' the organization of the data and resulted in defining some data-items which were not originally MK3-database L-codes. The NGSQualityflag, which is identicaly to the NGS quality code in NGS cardfiles was introduced as well as some other items specific to *solve*. This illustrates one of the strengths of this approach—it is easy and straightforward to add or modify data items.

The Goddard VLBI group is actively involved in writing replacements for earlier stages in the data analysis. *nu-Solve*, our replacement for interactive solve, is being modified to read and write openDB files. *OpenDBcal*, a replacement for *dbcal* that works directly with openDB files, is currently in alpha testing. *openDBcalc*, a calc replacement, and *makopenDB*, are currently under development. We expect all of these programs to be complete in early 2013. At this stage we will no longer need Mark III databases.

Timing tests done at Goddard indicate that sessions involving a small amount of data, such as the Intensives, take longer to process in using the openDB format instead of superfiles. We speculate that this is because of: A) The extra overhead involved in processing many files instead of one; and B) Some intermediate conversion of the openDB data that happens as *solve* reads it in. On the other hand, large sessions such as the RDVs and T2s are processed much faster using the openDB format instead of superfiles. In processing a solution using all available VLBI data these two effects balance out, and there is negligible difference in run time between a large solve run using superfiles and openDB format files. Since the amount of data in each session continues to increase, we believe that in the future openDB format will be superior in terms of execution time for *calc/solve*.

5. Next Steps and Conclusions

Currently only two software packages can use the openDB format: *Solve* in its global mode and *VieVS*. For the new format to become the geodetic VLBI standard it is neccessary that other VLBI analysis programs be able to read and write openDB format. The developers of both *C5++* and *Occam* are planning on doing so.

The draft proposal written in 2009 needs to be modified and updated to reflect the latest changes to the openDB format. This proposal was written prior to trying to use the format in analysis packages. Although the heart of the proposal remains intact—storing the VLBI data items in 'small' NetCDF files, and organizing these files by an ASCII 'wrapper' file—some of the details were modified as a result of experience gained in using the format in a real-world situation.

Lastly, we plan on writing a final report which we will submit to the IVS Directing Board in March of 2013 at the EVGA meeting. At this point the working group will be formally dissolved.

6. Appendix

This appendix displays the wrapper for an Intensive session. This wrapper can easily be read and modified by a text editor.

```
! Information contained in NGS cards.
Begin History
Created 2012/07/19 13:48:30
Createdby John Gipson  NVI, Inc./GSFC
Program  db2openDB         2012Jul17
History  10JAN04XU_V004.hist
End History
! **** Start of Session Section *****
Begin Session
Session I10004
AltSessionId 10JAN04XU
Head.nc
```

```
Default_Dir Session
ScanIndex.nc
TimeUTC.nc
End Session
! **** Start of Station Section *****
Begin Station KOKEE
Default_Dir KOKEE
AzEl.nc
Met.nc
Cal_kCable.nc
TimeUTC.nc
End Station KOKEE
!
! *** WETTZELL omitted because of space****
!
! **** Start of Observation Section ****
Begin Observation
Default_Dir Obs
GroupDelay_bX.nc
AmbigSize_bX.nc
GroupRate_bX.nc
GroupRate_bS.nc
GroupDelay_bS.nc
AmbigSize_bS.nc
Source.nc
Baseline.nc
ObsIndex.nc
!
Default_Dir ObsEdit
GroupDelayFull_bS_V004.nc
GroupDelayFull_bX_V004.nc
NGSQualityFlag_V004.nc
End Observation
```

References

[1] http://ivscc.gsfc.nasa.gov/about/org/documents/ivsTOR.html.

[2] Gipson, J., IVS Working Group 4 on VLBI Data Structures, In: Measuring the Future, Proceedings of the Fifth IVS General Meeting, St. Petersburg, Nauha, 2008, A. Finkelstein and Dirk Behrend (eds.), 143-152, 2008.

[3] Gipson, J., IVS Working Group 4: VLBI Data Structures, In: IVS 2010 General Meeting Proceedings, NASA/CP-2010-215864, D. Behrend and K. D. Baver (eds.), 187-191, 2010.