Working Group 4
VLBI Data Structures

John Gipson

19th EVGA Working Meeting
March 24-25, 2009

Working Group Members

Chair	John Gipson
Analysis Coordinator	Axel Nothnagel
Haystack/Correlator Representative	Roger Cappalo
GSFC/Calc/Solve	David Gordon Dan MacMillan
IAA/QUASAR	Sergey Kurbodov Elena Skurhina
JPL/Modest	Chris Jacobs
Occam	Oleg Titov
Vienna	Johannes Boehm
Main Astronomical Observatory/Steelbreeze	Sergei Bolotin
Observatorie de Paris/PIVEX	Anne-Marie Gontier
NICT	Thomas Hobiger Hiroshi Takiguchi

Working Group Charter

From the IVS website:

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 software** to the new system.

Current Format

Mk3 database. Currently 30+ years old. Used to archive and transmit IVS sessions.

A product of its time:

- Designed to run on systems with 20k (!!) memory
- Designed before Fortran had strings

Furthermore…

- Hard to port
- Slow
- Baseline oriented → Tremendous redundancy
- Theoretical and observation data mixed.
- Limited user community (20 users?)

Current Format

Mk3 database.

In spite of its flaws, it has served us well.

- Lasted 30 years—testament to good design.
- Self describing data format.
- Can add new datatypes.

Some Goals

0. Handle current and anticipated VLBI data.
1. Reduce redundancy
2. Ease of access
3. Flexibility
4. Separation of "observations" from "models" and "theory"
5. Ability to access data at different levels of abstraction
6. Ability to easily access most common parts of the data
7. Consistency
8. Completeness ???

Some Issues

1. How should data be organized within a session?
2. How should data be stored? (Related to ease of access)
3. How should data be organized across sessions?
4. What impact do these choices have on data flow?

Current Data Types

Current database format has two types of data:

1. Session data (type 1-lcodes). Scope= entire session
   A. Stations
   B. Sources
   C. Correlator
   D. …

Current Data Types

Current database format has two types of data:

1. Session data (type 1-lcodes). Scope= entire session
   A. Stations
   B. Sources
   C. Correlator
   D. …
2. Observation data (type 2&3 lcodes). Scope = observation
   A. Observables
   B. Ambiguities and Editing
   C. Station Az-El
   D. Loading corrections
   E. Calibrations
   F. EOP
   G. …

Redundancy

1. Many (most?) of the type 2&3 lcodes are really scan and station dependent, and not observation dependent:
   A. Station Az-El
   B. Loading corrections
   C. Calibrations
   D. Atmospheric delay

2. Others are only scan dependent:
   A. Source
   B. EOP

This makes the data tremendously redundant.

Redundancy of R1360

#Stats	#Scans			
2	366			
3	232			
4	158			
5	71			
6	35			
7	0			

Redundancy of R1360

#Stats	#Scans	scans*stats		
2	366	732		
3	232	696		
4	158	632		
5	71	355		
6	35	210		
7	0	0		
Total		**2625**		

Redundancy of R1360

#Stats	#Scans	scans*stats	#BL	
2	366	732	1	
3	232	696	3	
4	158	632	6	
5	71	355	10	
6	35	210	15	
7	0	0	21	
Total		**2625**		

Redundancy of R1360

#Stats	#Scans	scans*stats	#BL	#BL*2*#scans
2	366	732	1	732
3	232	696	3	1392
4	158	632	6	1896
5	71	355	10	1420
6	35	210	15	1050
7	0	0	21	0
Total		**2625**		**6490**

2xNumber of observations

Redundancy of R1360

#Stats	#Scans	scans*stats	#BL	#BL*2*#scans
2	366	732	1	732
3	232	696	3	1392
4	158	632	6	1896
5	71	355	10	1420
6	35	210	15	1050
7	0	0	21	0
Total		**2625**		**6490**
Redundancy				**2.47**

Redundancy of RDV73

RDV73				
#Stats	#Scans	scans*stats	#BL	#BL*2*#scans
2	134	268	1	268
3	149	447	3	894
4	55	220	6	660
5	42	210	10	840
6	30	180	15	900
7	24	168	21	1008
8	33	264	28	1848
9	34	306	36	2448
10	40	400	45	3600
11	60	660	55	6600
12	39	468	66	5148
13	22	286	78	3432
14	16	224	91	2912
15	11	165	105	2310
Total		4266		32868
Redundancy				**7.70**

Redundancy of stat16_6_2p1D0ln

stat16_6_2p1D0ln				
#Stats	#Scans	scans*stats	#BL	#BL*2*#scans
2	164	328	1	328
3	36	108	3	216
4	138	552	6	1656
5	423	2115	10	8460
6	725	4350	15	21750
7	1282	8974	21	53844
8	1377	11016	28	77112
9	1391	12519	36	100152
10	533	5330	45	47970
11	50	550	55	5500
12	9	108	66	1188
13	0	0	78	0
Total		**45950**		**318176**
Redundancy				**6.92**

Redundancy of Stat32_6_2p1D0ln

stat32_6_2p1D0ln				
#Stats	#Scans	scans*stats	#BL	#BL*2*#scans
2	431	862	1	862
3	261	783	3	1566
4	145	580	6	1740
5	80	400	10	1600
6	49	294	15	1470
7	15	105	21	630
8	6	48	28	336
9	34	306	36	2448
10	97	970	45	8730
11	200	2200	55	22000
12	232	2784	66	30624
13	362	4706	78	56472
14	516	7224	91	93912
15	596	8940	105	125160
16	601	9616	120	144240
17	683	11611	136	185776
18	639	11502	153	195534
19	274	5206	171	93708
20	82	1640	190	31160
21	10	210	210	4200
22	0	0	231	0
Total		69987		1002168
Redundancy				**14.32**

How to Reduce Redundancy

Introduce two new types of data:

1. Station-scan data depends only on the station and the scan.
2. Scan data depends only on the scan.

This requires modest additional bookkeeping:

1. A table that connects observations to scans.
2. A table that connects scans to stations.

How to Reduce Redundancy

Introduce two new types of data:

1. Station-scan data depends only on the station and the scan.
2. Scan data depends only on the scan.

This requires modest additional bookkeeping:

1. A table that connects observations to scans.
2. A table that connects scans to stations.

We could do this using the *present* Mark3 database format.

Ease of Access

- Ability to easily access data on different platforms.
- Ability to use different languages.
- Speed

Ease of Access

- Ability to easily access data on different platforms.
- Ability to use different languages.
- Speed

There are many data storage formats that meet these goals: NetCDF, CDF, HCDF.

Ease of Access

- Ability to easily access data on different platforms.
- Ability to use different languages.
- Speed

There are many data storage formats that meet these goals: NetCDF, CDF, HCDF.

I recommend using NetCDF.

Ease of Access

- Ability to easily access data on different platforms.
- Ability to use different languages.
- Speed

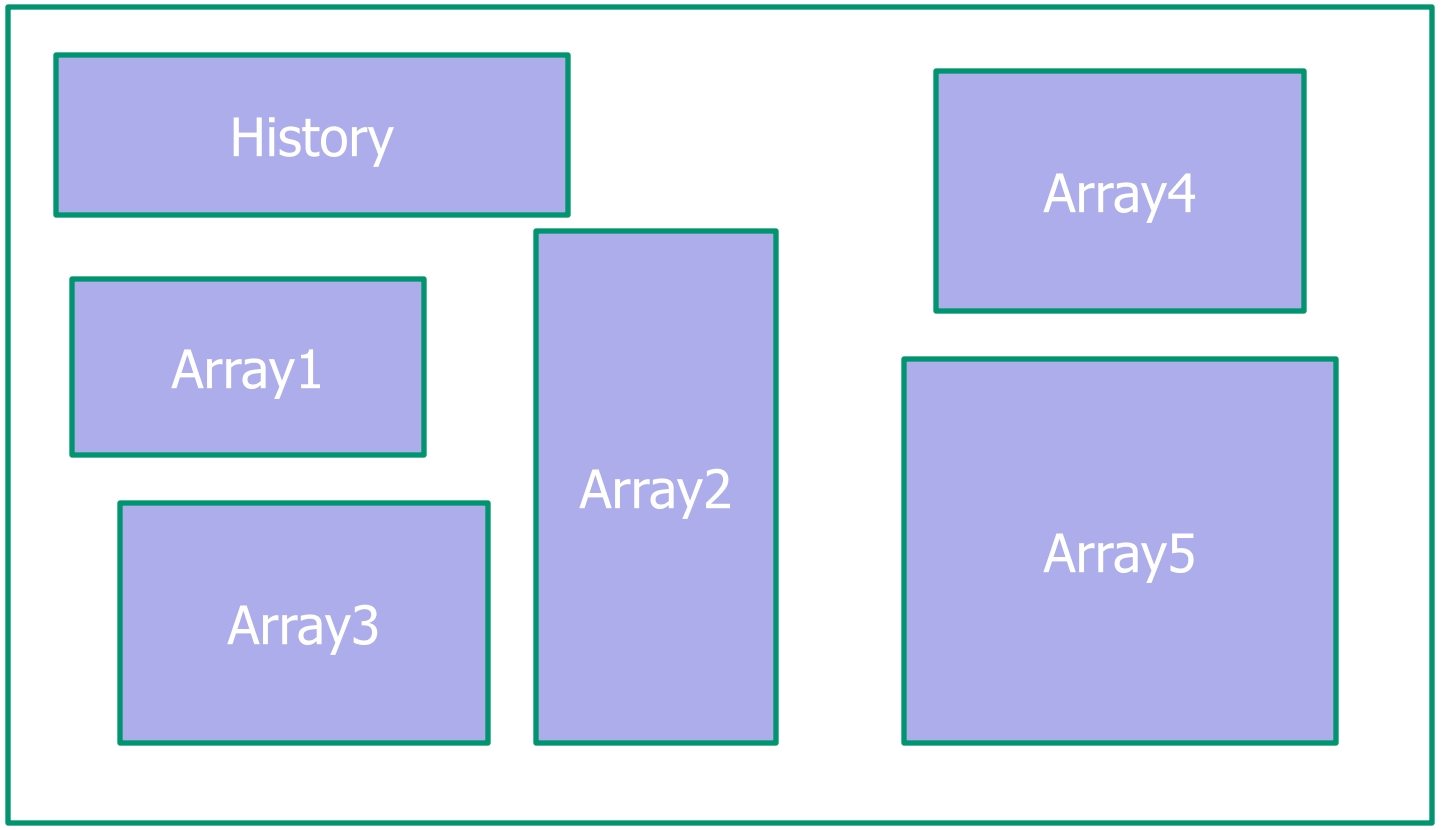
There are many data storage formats that meet these goals: NetCDF, CDF, HCDF.

I recommend using NetCDF.
This also makes it possible to access sub-sets of the data.

Why NetCDF

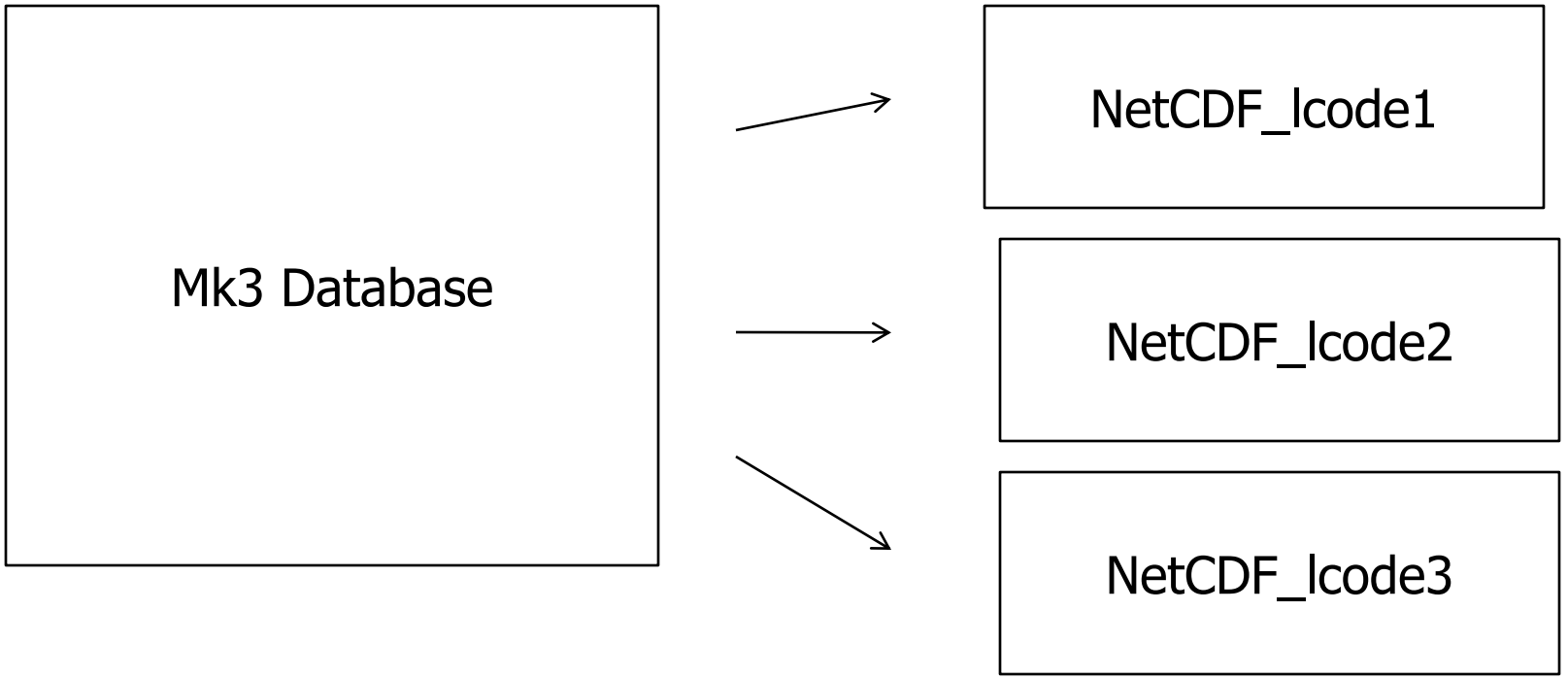
1. Meets stated goals.
2. Self-describing data format.
3. Large user community.
4. Many tools.
5. Well established conventions.
6. Flexible.

What does a NetCDF File Look Like?



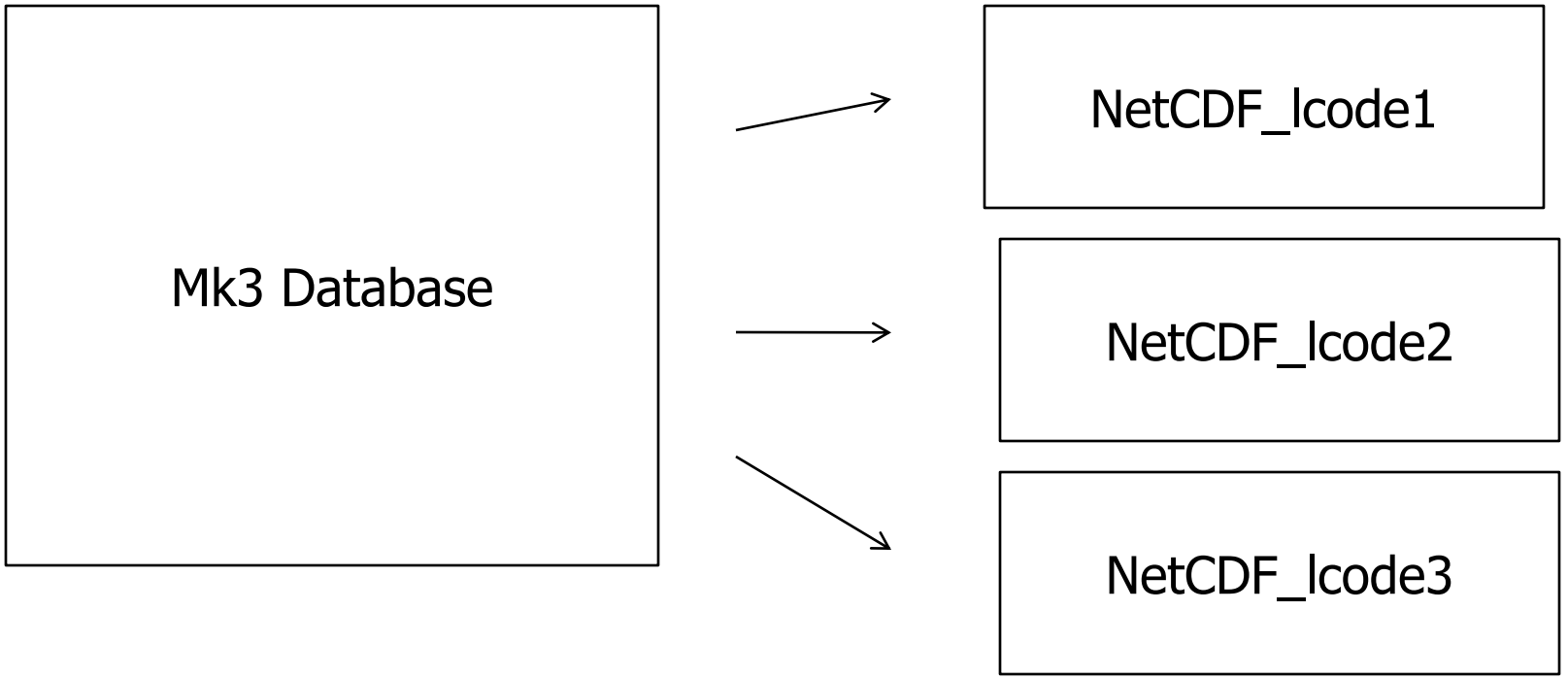
A NetCDF file can contain an arbitrary number of arrays. The arrays can differ in dimensions and type (byte, short, integer, real, double). The arrays can have attributes like name, unit, long-name, description associated with them.

Hobiger's NetCDF database implementation



There is a 1-1 mapping between lcodes and NetCDF arrays.

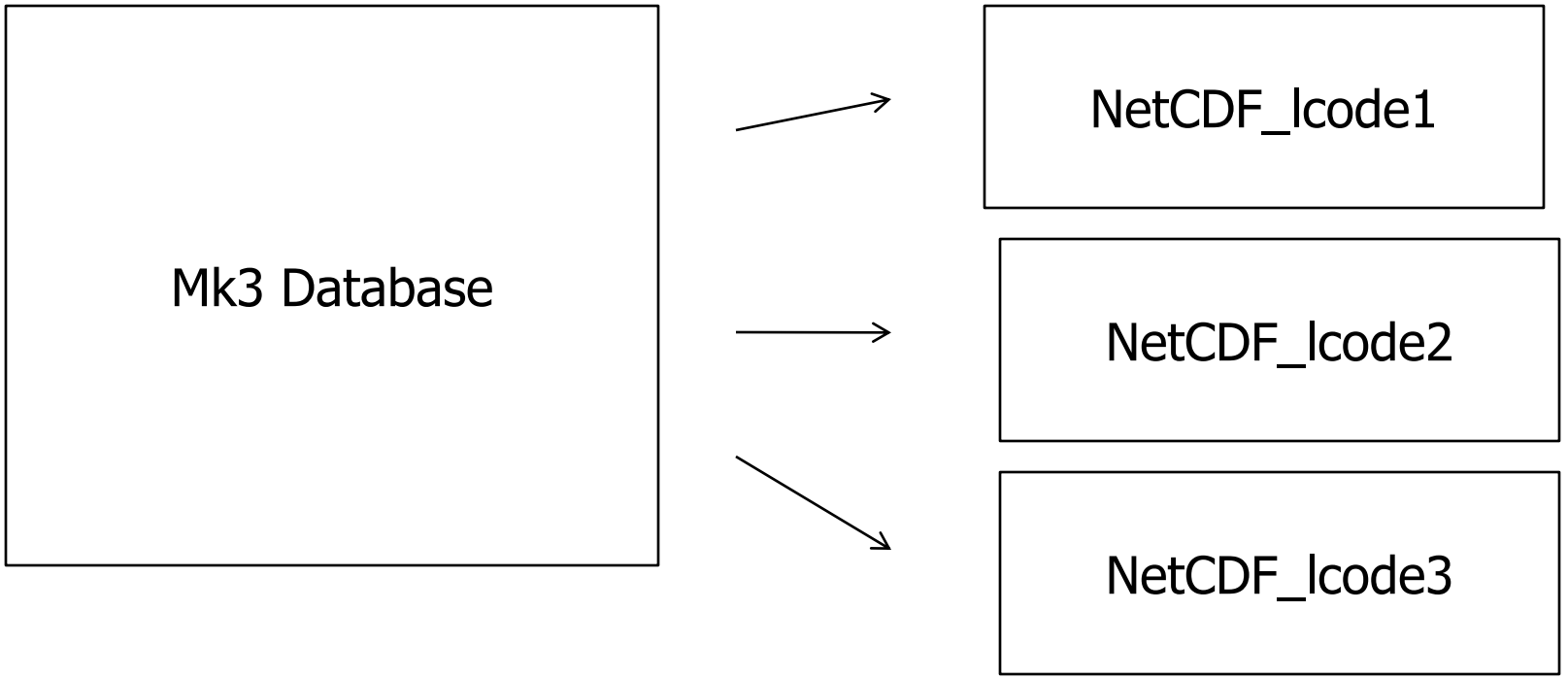
Hobiger's NetCDF database implementation



There is a 1-1 mapping between lcodes and NetCDF arrays.

Can also go from NetCDF files to Mk3 database.

Hobiger's NetCDF database implementation



There is a 1-1 mapping between lcodes and NetCDF arrays.

Can also go from NetCDF files to Mk3 database.

This would meet the design goals of accessibility and speed.

Splitting the VLBI Session data

The Mark3 database format was designed so that *all* data pertaining to a session resides in one file.

Addvantage: "one-stop-shopping".

Disadvantages:

1. Anytime anything changes—calibrations, ambiguities, models—you need a new version of the database.
2. Anytime something is added to the database, you need a new version of the database.
3. The database now contains lots of obsolete information that is no longer used.

Splitting the VLBI session data

Proposal: Gather data that is similar in scope, origin, physical effect, frequency of change. Store in its own file.

Splitting the VLBI session data

Proposal: Gather data that is similar in scope, origin, physical effect, frequency of change. Store in its own file.

1. Experiment info: everything known about experiment beforehand.
2. Atmospheric delay
3. Met data
4. Calibrations
5. Physical and geophysical effects calculable beforehand: relativity, tidal ocean loading, etc.
6. Physical and geophysical effects calculable afterwards: atmosphere loading, hydrological loading, etc.
7. Observables and commonly used observation related data.
8. Editing and Ambiguity
9. Less commonly used observation related data

Splitting the VLBI session data

Advantages:

1. Items that are not expected to change are separated from items that may change.
2. Data is separated from models.
3. This approach lends itself to building up the session piece by piece.
4. We delay discussion of what the VLBI2010 observable format should look like.
5. Commonly used data is separated from less commonly used data.
6. This enables easy testing of new models.
7. As models improve, they can be easily incorporated.

Organizing the Data With Wrappers

Corresponding to a database is a wrapper. This is a special file that contains pointers to the associated data files:

R1345.wrp
! Standard IVS session.
!More info about what is in here.
R1345_sess_GSFC_0001.nc
R1345_atm_GSFC_0001.nc !NMF mapping function
R1345_srcmap_GSFC_0001.nc !Source maps to use.
…
R1345_obsc_HAYS_0001.nc
R1345_obsu_HAYS_0001.nc
R1345_amed_GSFC_0001.nc

Advantages and Uses of Wrappers

1. Can specify "IVS-standard" session.
2. Can incorporate alternative models by replacing a generating alternative NetCDF file and pointing to it in the wrapper file. E.g.

   R1345_atm_GSFC_0001.nc →R1345_atm_VIEN_001.nc

3. Researchers can use their own "private" wrappers to test alternative models.
4. Groups can swap editing and ambiguity information.
5. Can easily add new data types.
6. Can use this to preserve history of processing.

Next Steps

- Solicit feedback
- Refine approach
- Write detailed design document
- Write software to convert from MK3 to new format.
  - I have written a routine that converts a subset of a Mark3 database to NetCDF.
- Write software to make new format from scratch.