Status of netCDF-3, netCDF-4, and CF Conventions

Russ Rew

Community Standards for Unstructured Grids Workshop, Boulder

2006-10-16

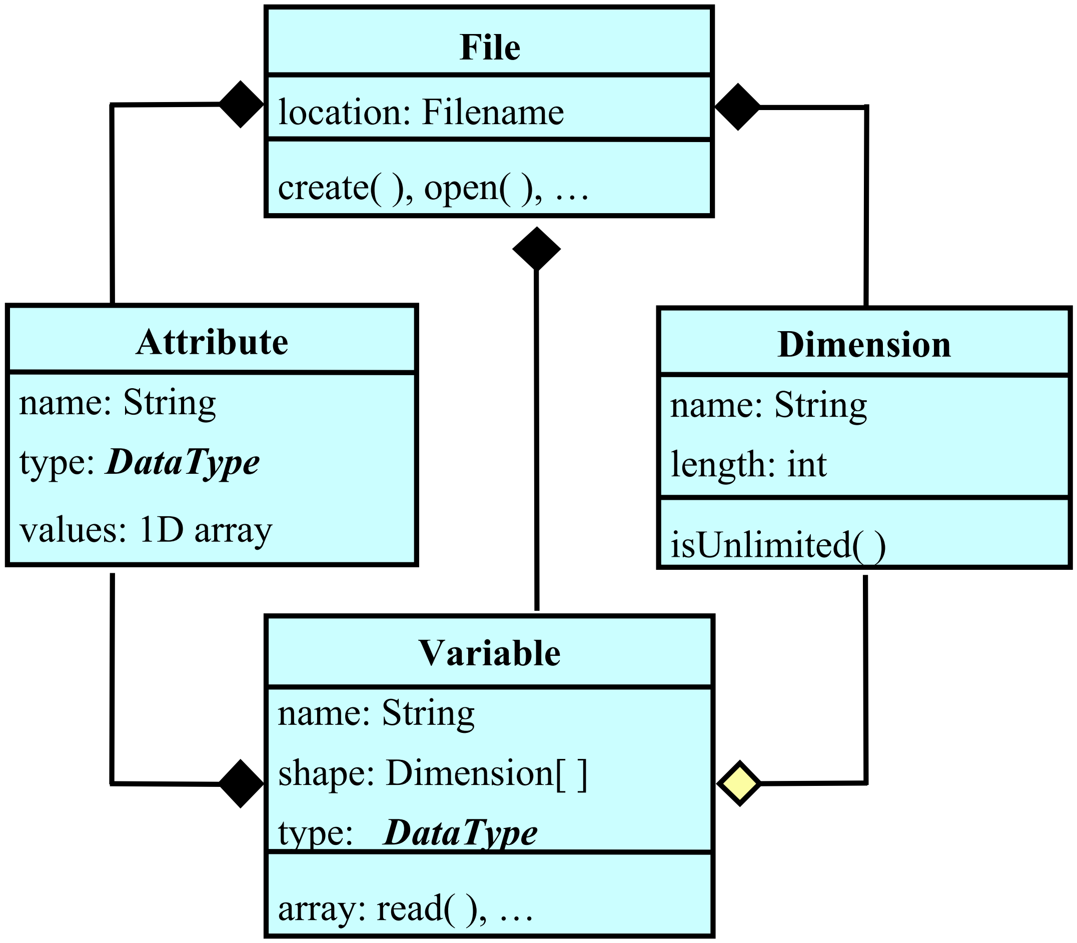
Status of netCDF-3 Work

- NetCDF 3.6 (C, Fortran, C++) eliminated most 2 GiByte size limitations
  - Supports 64-bit offset file format variant
  - Other improvements not very relevant to unstructured grids: shared libraries, portability, performance, C++ improvements, …
- NetCDF Java (nj22) continues to advance
  - Through a Common Data Model interface, reads netcdf-3, HDF5 (most), OPeNDAP, GRIB1, GRIB2, BUFR, …
  - Provides CF conventions compliance, coordinate systems:
  - I/O provider framework for adding new data formats
- libcf under development for a CF conventions API based on netCDF-3

Status of NetCDF-4

- NetCDF-4.0-alpha17 currently available for testing
  - Files created with alpha release use unsupported artifacts
  - We're still seeking feedback on performance and functionality
  - Early users have obtained 4:1 compression and 7x speedups
- NetCDF-4.0-beta waiting for HDF5 1.8-beta
  - Will finalize file format, eliminate necessity for artifacts
  - Expected within a few weeks of HDF5 1.8-beta release
- HDF5 1.8 currently expected in January 2007
  - Has enhancements specifically for netCDF-4: variable creation order, Unicode names, dimension scales, on-the-fly numeric conversions
- Plans for netCDF-4.1 and beyond on netCDF-4 web site

NetCDF-3 Data Model



File

location: Filename

create( ), open( ), …

Attribute

name: String

type: ***DataType***

values: 1D array

Dimension

name: String

length: int

isUnlimited( )

Variable

name: String

shape: Dimension[ ]

type: ***DataType***

array: read( ), …

Variables and attributes have one of six primitive data types.

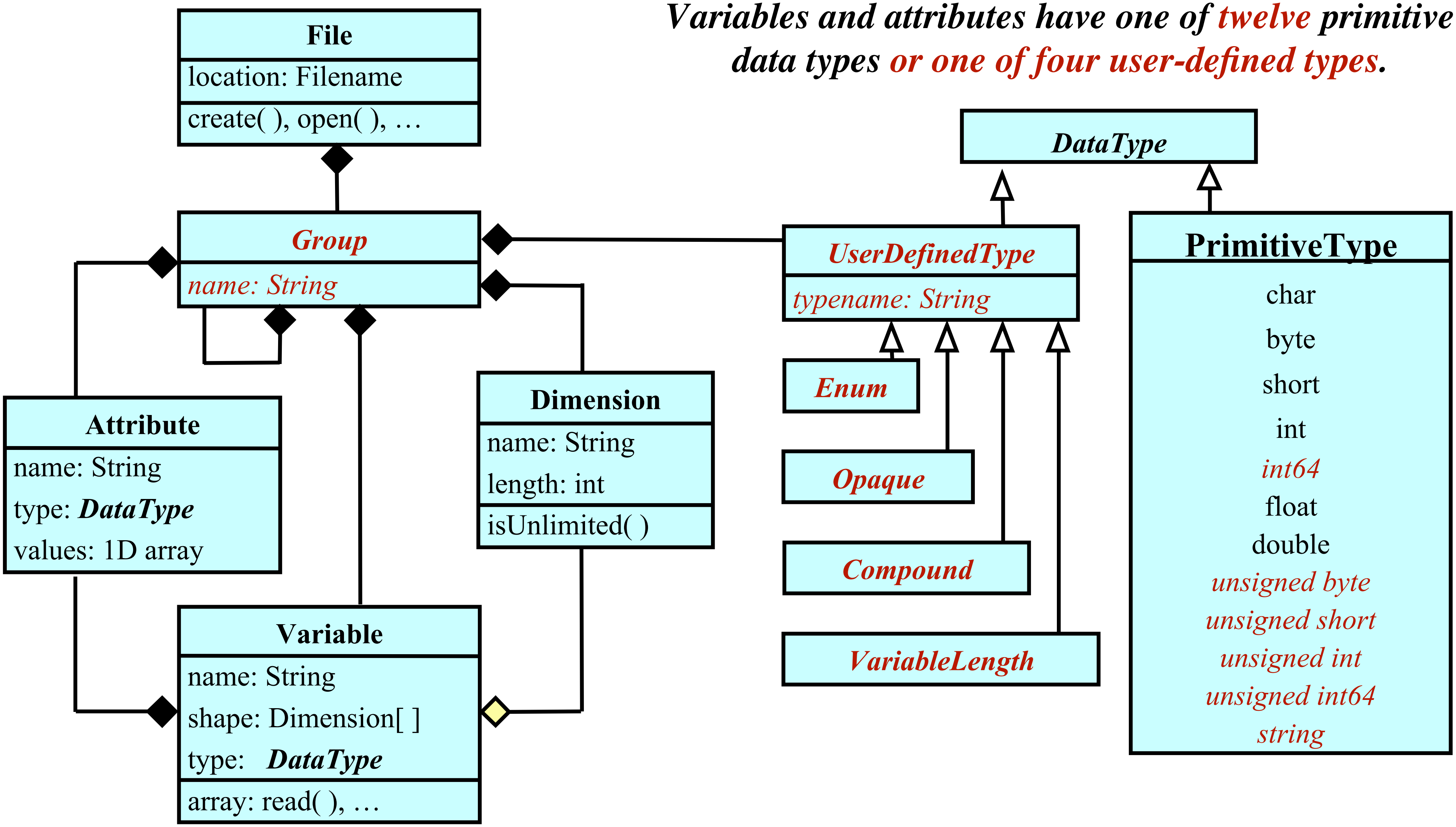
DataType

char
byte
short
int
float
double

A file has named variables, dimensions, and attributes. Variables also have attributes. Variables may share dimensions, indicating a common grid. One dimension may be of unlimited length.

NetCDF-4 Data Model (Common Data Access Model)

Variables and attributes have one of twelve primitive data types or one of four user-defined types.



A file has a top-level unnamed group. Each group may contain one or more named subgroups, variables, dimensions, attributes, and types. Variables also have attributes. Variables may share dimensions, indicating a common grid. One or more dimensions may be of unlimited length.

Some netCDF-3 Limitations

- Relevant to representing unstructured grids:
  - No data structures, just scalars and multidimensional arrays
  - No ragged arrays or nested structures
  - Only one shared unlimited dimension
  - Flat name space for dimensions and variables
- Not relevant (?) for unstructured grids:
  - No strings, just arrays of characters
  - Limited numeric types
  - Only ASCII characters in names
  - Changes to file schema can be expensive
  - Efficient access requires reads in same order as writes
  - No built-in compression
  - Only serial I/O

New Features of netCDF-4

- Relevant to representing unstructured grids:
  - User-defined compound types (portable structs)
  - User-defined variable-length types for ragged arrays
  - Groups for nested scopes
  - Multiple unlimited dimensions
- Not relevant (?) for unstructured grids:
  - String type
  - Additional numeric types
  - Unicode names
  - Efficient dynamic schema changes
  - Multidimensional tiling (chunking)
  - Per variable compression
  - Parallel I/O

User-Defined Compound Type

```
types:
    compound ob {
        int station_id;
        double time;
        float temperature;
        float pressure;
    }
variables:
     ob obs(nstations);
```

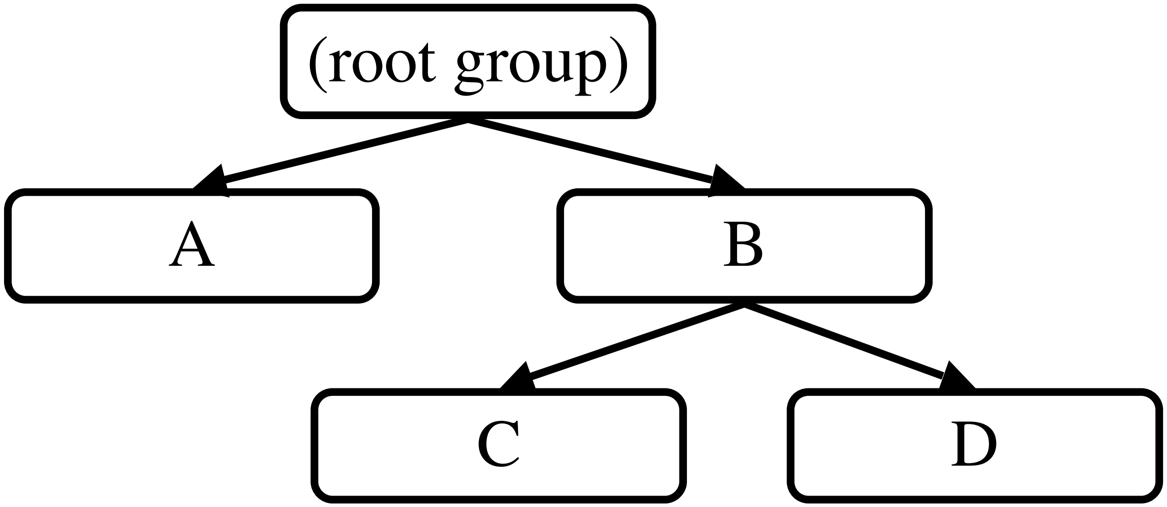
- Like C structs, but portable
- May be nested
- Multiple variables may use same type
- Attributes may be of compound type also (needed for units)
- Efficiency note: members stored close together

User-Defined Variable Length Type

```
types:
        float(*) row_of_floats;
variables:
   row_of_floats ragged_array1(m);
```

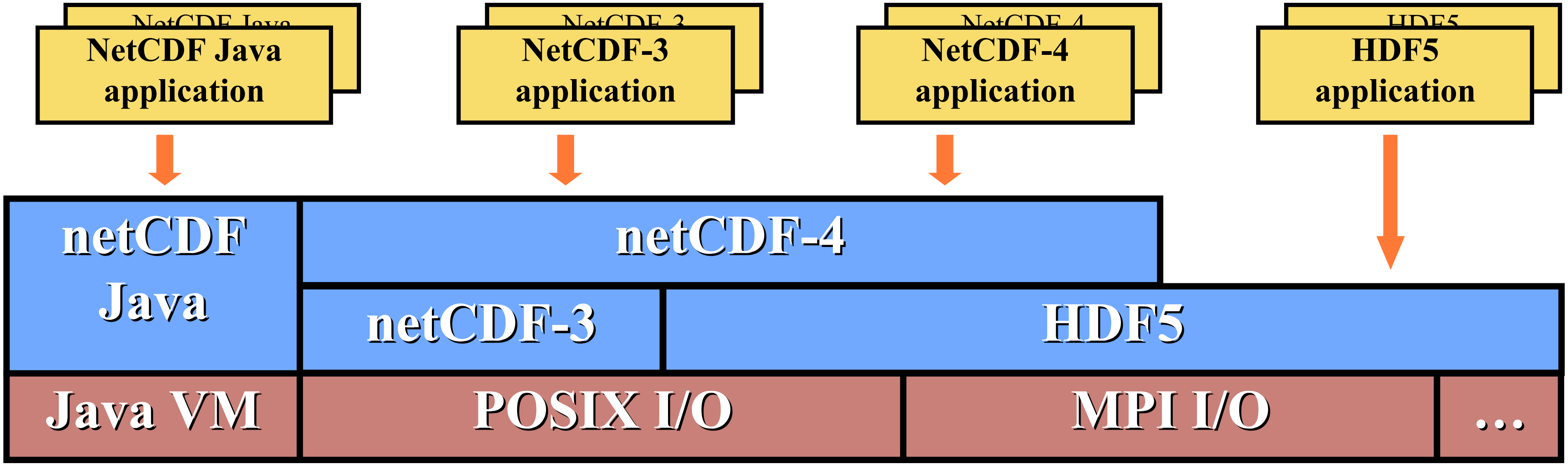
- Has a name and a base type
- Can be used for “ragged arrays”
- Access to a variable-length value is atomic
    - Length and values written or read together
    - Can't know length until value is read
    - In C/Fortran, library allocates memory for value
- Multiple variables may use same type
- May be nested to create multidimensional variable-length types

Groups



- A non-root Group has a name and a parent group
- The root group is unnamed
- A Group may have variables, dimensions, attributes, types, and subgroups
- A Group is analogous to a netCDF-3 file

NetCDF-4 Architecture



- NetCDF-4 uses HDF5 for storage, high performance
  - Parallel I/O
  - Chunking for efficient access in different orders, efficient use of compression
  - Conversion using “reader makes right” approach
- Provides simple netCDF interface to subset of HDF5
- Also supports netCDF classic and 64-bit formats

Status of CF

- White paper available on *"Maintaining and Advancing the CF Standard for Earth System Science Community Data"*, Bryan Lawrence, et al
- CF becoming important to more communities
- New web site set up for discussions, maintenance: http://cf-pcmdi.llnl.gov/ eventually: http://cfconventions.org/
- Funded staff now supporting CF
- CF Governance Panel now in existence (Oct 1), responsible for stewardship not technical content
- Under WMO/WCRP Working Group on Coupled Modeling (WGCM)
- Two CF committees
  - Conventions
  - Standard Names

Some Unstructured Grid Issues

- Is netCDF-3 data model adequate for representing unstructured grids?
- If not, what netCDF-4 features are needed for unstructured grid representations?
- Can needed netCDF-4 features for unstructured grids be emulated in netCDF-3 data model?
- Should means of emulation of particular netCDF-4 features in netCDF-3 be elevated to conventions level?