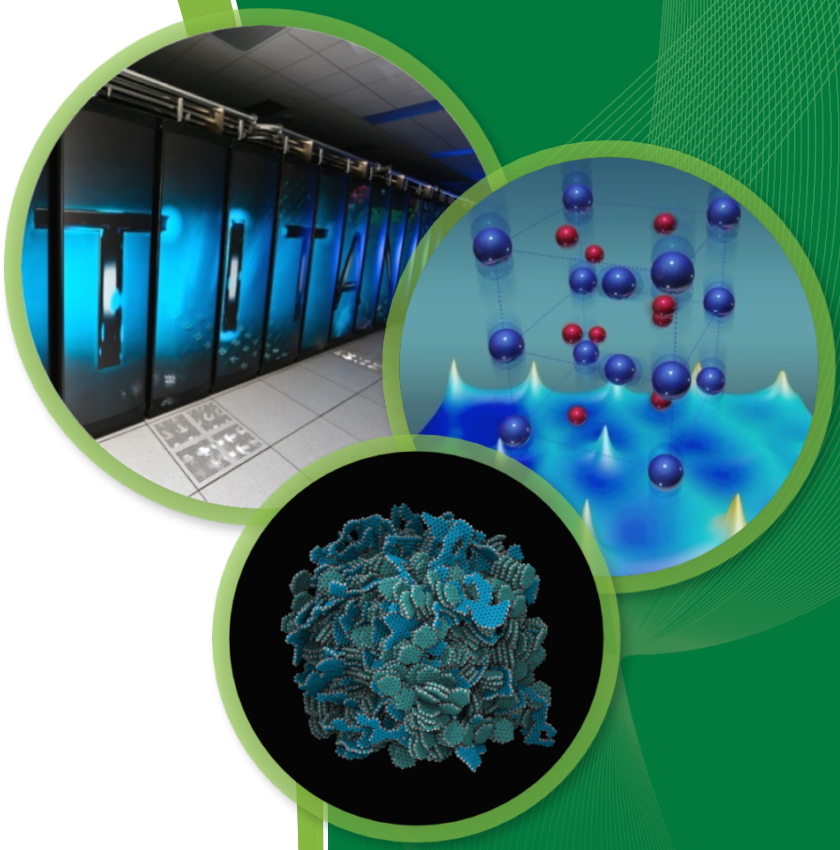
Current Status of Access Routines to ENDF Data in AMPX

Dorothea Wiarda



WPEC 2017 – Subgroup 43

Paris, France
May 16, 2017

ORNL is managed by UT-Battelle
for the US Department of Energy

OAK RIDGE National Laboratory

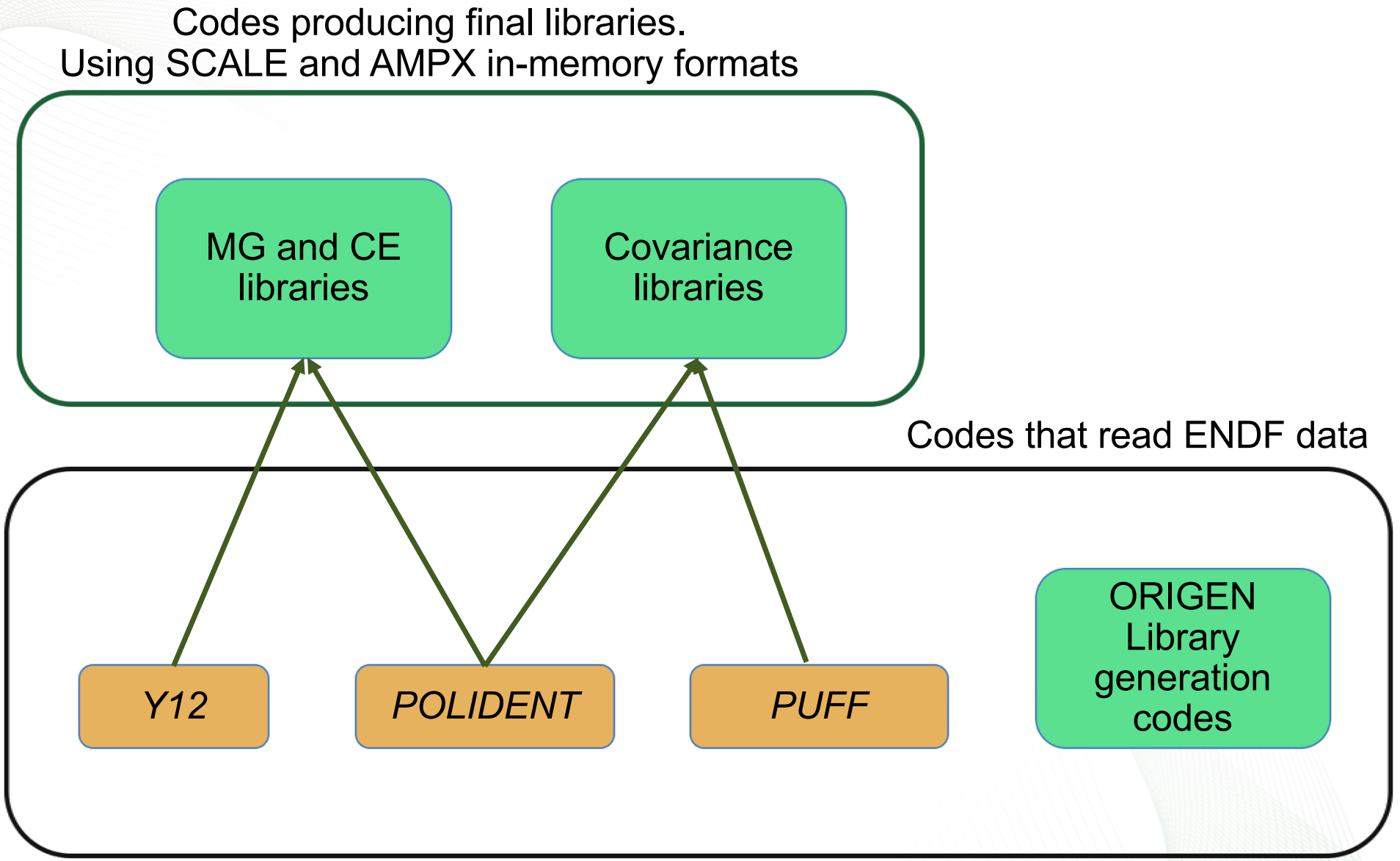
Outline

- Current status of ENDF processing in AMPX
  - In-memory containers for AMPX and SCALE data
  - How do the processing codes use the in-memory containers
  - Interaction between ENDF data and in-memory containers
- Current status of GND basic containers in AMPX
  - Definition for aliases for GND data types
  - Base container classes
  - Values container
  - Other containers

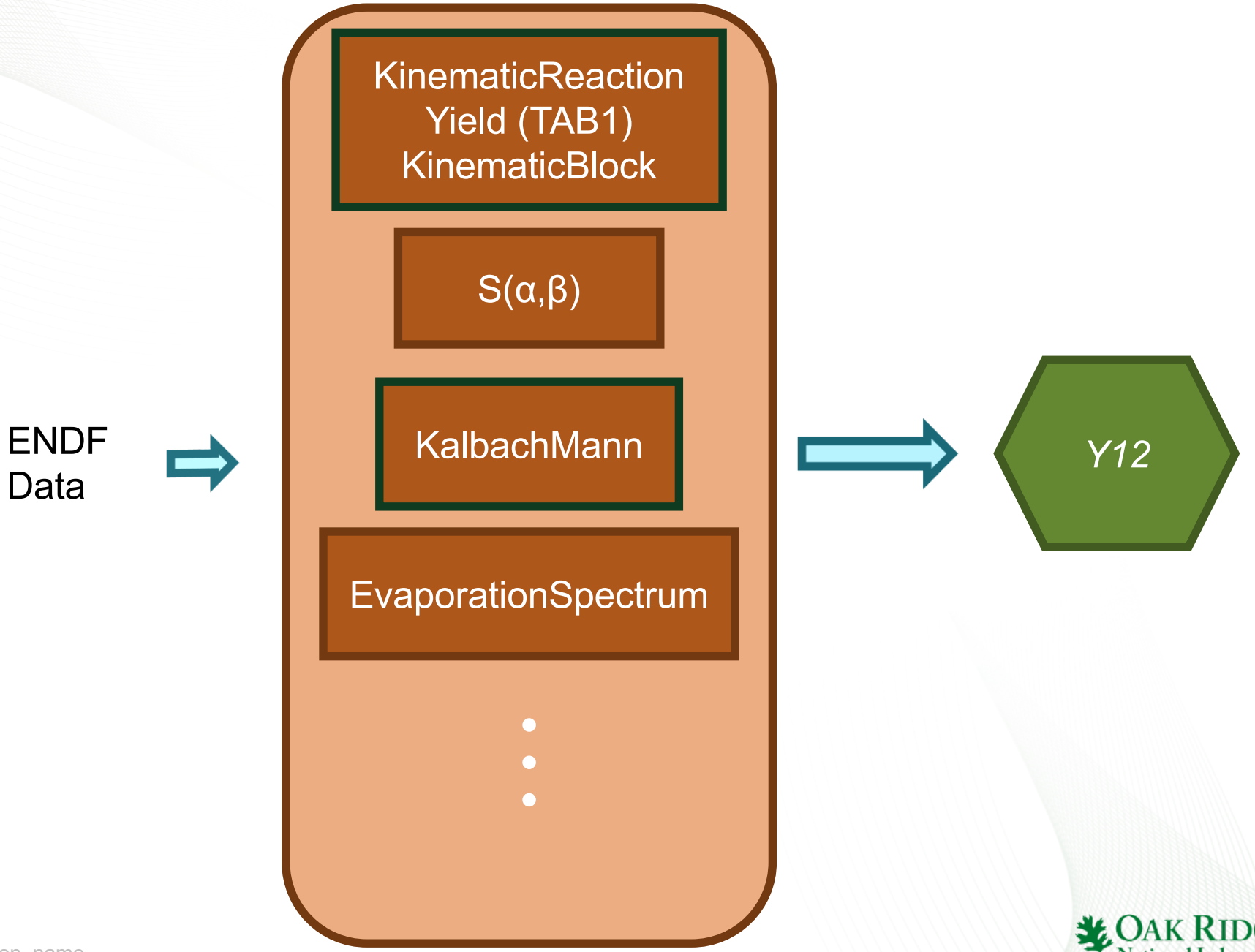
AMPX and SCALE Philosophy

- Use in-memory container for data:
  - C++ resource for MG library
  - C++ resource CE library
  - C++ resource for COVERX formatted covariance libraries
  - C++ resource for resonance parameters
  - and many others
- If possible pass data in-memory not via disk I/O
- Where applicable, in-memory data containers have a FORTAN binding.

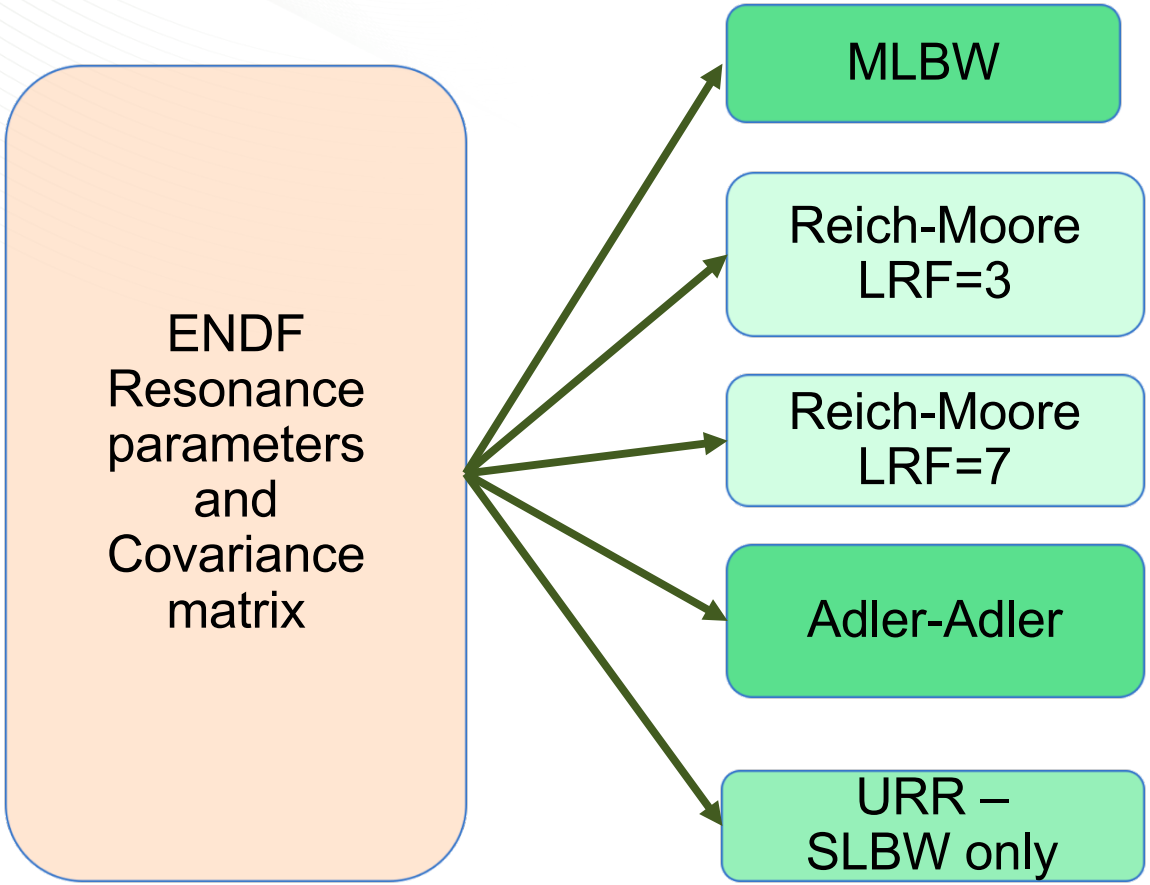
Library generation in AMPX



Processing of kinematic data



Resonance API



- SLBW and MLBW parameter are stored in the same class with a flag indicating which formalism to use
- Resonance parameters for Reich-Moore for LRF=3 are initially stored in a different class, but are converted to a LRF=7 class before calculation
- If derivatives are desired, all formalisms (except URR) are converted to LRF=7 so SAMRML can be used under the hood
- All resonance parameter classes can contain a covariance matrix. If converting to a different formalism, the covariance matrix is re-organized accordingly

Fill AMPX data structures from ENDF

- Processing routines call a function with arguments:
  - type: string of the desired format, currently only ”endf” is allowed
  - out: input stream opened to the file to read
  - Any other arguments that might be needed for the relevant data. For example the kinematic data can be retrieved for exiting neutrons only.
- The result is an in-memory AMPX data container with the relevant ENDF data filled in.
- The above routine determines what class to call to read the data. All legacy *endf* format related classes are collected in a different directory and name space. The *gnd* format has a different name space.
- Examples are:
  - *getKinematicRawData*
  - *getResonanceInfo*
  - *retrieveCovariance*

Structure of the legacy ENDF classes

- Base class (Record) that contains MAT, MT and MF values along with routines to read and write the right part of an ENDF record
- Classes that inherit from Record for *text, list, tab1, tab2 and intg* ENDF records.
- Classes for each ENDF file, that either directly fill an in-memory AMPX class (File 8) or read and store the *text, list, tab1, tab2* and *intg* for later use.
- Classes that combine the varies ENDF file data into the in-memory AMPX record. For example *EndfSetKinematicData* combines data from *File 4, 5, 6, 7,12, 13, 14*, and *15* to generate the data needed for processing in *Y12*
- All the above classes do not do any processing, just sorting of data into the in-memory AMPX class.

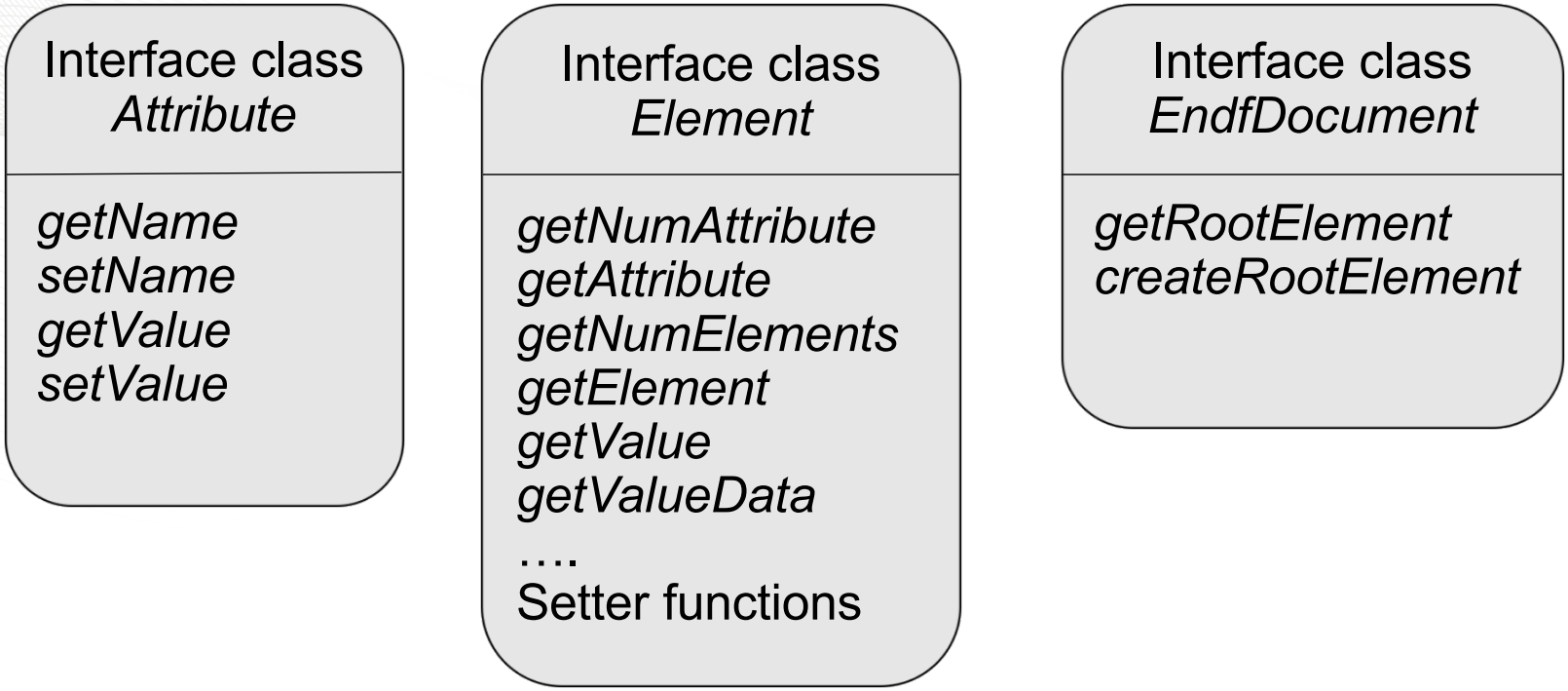
Initial GND processing in AMPX

Definitions for GND general containers

Add a definition file that sets aliases for the different GND data types:

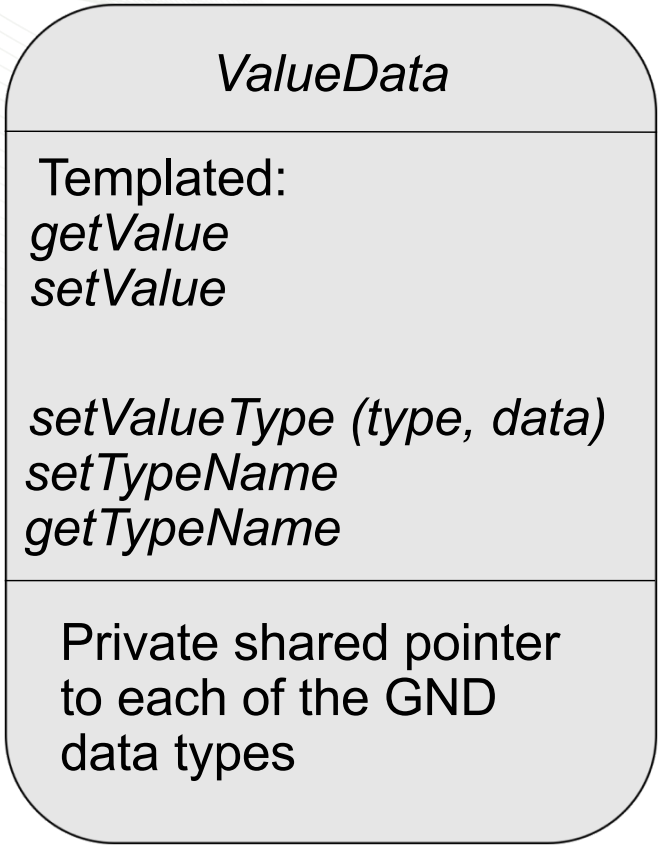
- Link *Float32* to C++ *float* and *Float64* to C++ *double*, etc.
  - In the remaining code always use GND data types
- Add template function to parse GND data types from text and to save to text.
  - Add instantiations for all allowed GND data types
- Add a templated function to get the default value for a GND data type
  - Add instantiations for all allowed GND data types
- Add enumerations for the type of allowed interpolations (*lin-lin, flat*, etc.) with the corresponding text to be used for on-disk representation of GND

Access to the GND files



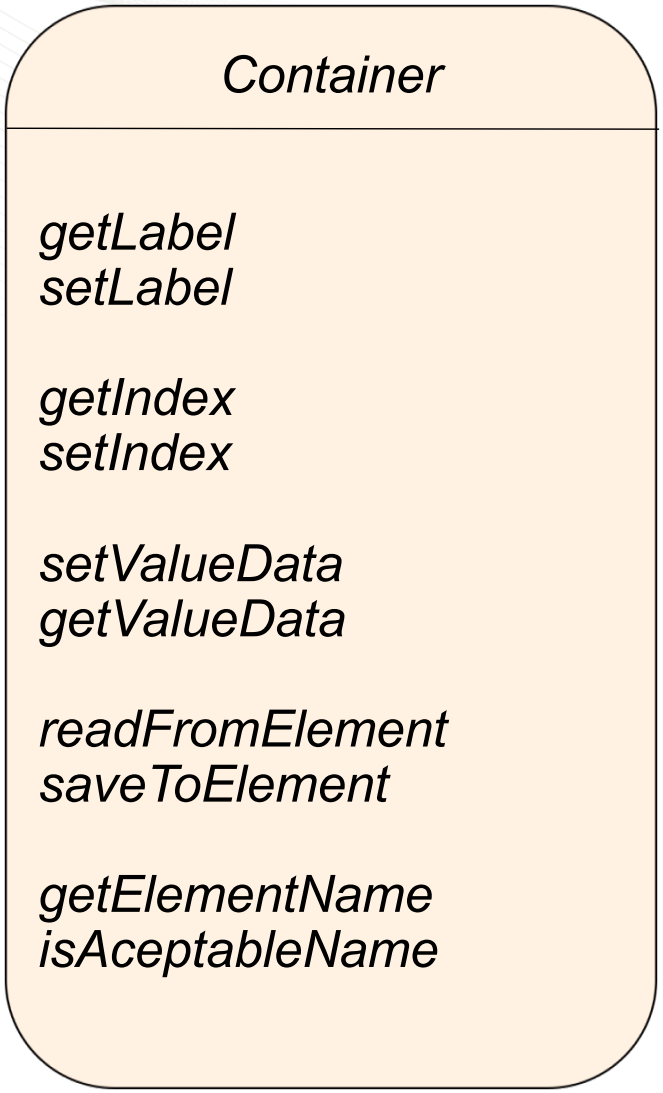
- An implementation for these classes currently exists, using QT XML classes.
- All AMPX GND classes only use the interface class
- The *getValueData* and *setValueData* functions are templated functions to take arrays of a specified GND data types. This should allow future on-disk formats to save the data as binary float or double data to preserve precision.

ValueData class



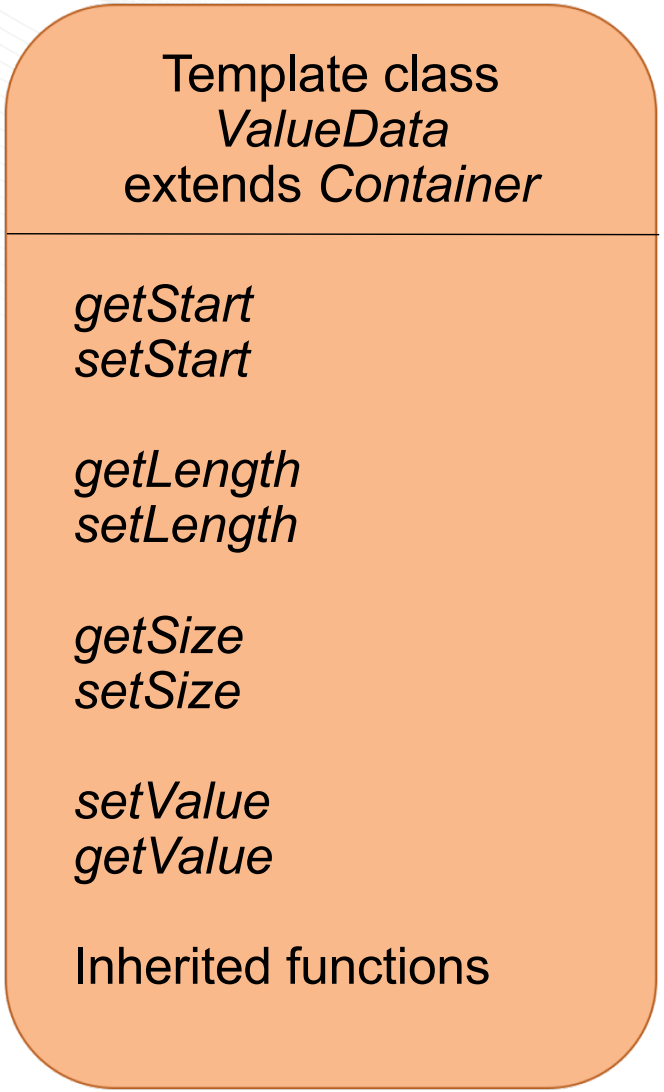
- *ValueData* class represent the optional value and valueType attributes of a functional container.
- One or none of the private shared pointer data to the various GND data types is different from NULL.
- *getValue* returns NULL if the *ValueData* is not of the desired type.

Container base class



- Container class that serves as base class for all other containers
- Combines common attributes of all containers and functional containers.
- Functions to interact with Element and Attribute interface classes. Child classes will first call these functions and then supplement.

ValueData and ValuesContainerHolder



- *ValueData* is a template class that implements the values container
- *ValuesContainerHolder* has a shared pointer to a *ValueData* object for each of the GND data types.
- One or none of the above pointers can be different from NULL.
- Convenience functions with implementations for all GND data types allow to get the underlying *ValueData*.

Remaining general data containers

- Classes for all the remaining general data containers have been added.
- If the container contains a gnd “values container”, one or more shared pointer for *ValuesContainerHolder* objects are added.
- If applicable, templates functions are added to get the data directly. For example:
  - The *ArrayContainer* for the array container has a templated function that takes a vector of indices to get or set the value of the matrix directly.
  - The *TableContainer* for the table container breaks the data into columns and each column in turn contains a *ValuesContainerHolder* object. This allows easy access to the data for a given row and column.
- Unit tests have been added for all classes in the base container package.

Summary

- AMPX uses in-memory structures to hold ENDF data.
- The structures are not dependent on the on-disk ENDF format.
- Processing routines only use the in-memory data, and do not read ENDF data directly.
- An implementation of the GND base containers has been added to AMPX in preparation for processing of the GND file.