Summary of LLNL/LANL/ORNL/BNL discussions on SG38

WPEC SG38, May 9 2016

Caleb M Mattoon

Lawrence Livermore National Laboratory

LLNL-PRES-XXXXXX

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

To speed up progress on SG-38, several labs continued discussions via web- and tele-conference starting early 2016

- Quick summary of progress
  - Revised the 'fission energy release' (MF=1 MT=458) section in GND to be more explicit
  - Better treatment for Coulomb elastic scattering
  - Researched efficiency of HDF as binary version of GND
  - Discussed role of XML schemas
  - ... plus lots of discussion about low-level data containers
  - Meeting notes available on https://ndclx4.bnl.gov/gf/project/gnd/docman/

- Meetings generally held around 09:00 Pacific Time (18:00 C.E.T.) Is there a better time to allow participation from SG38 members outside the U.S.?

Changes to fissionEnergyReleased:

- Previous version (GND-1.7):

```
<outputChannel genre="NBody">
  <Q>...</Q>
  <fissionEnergyReleased>
    <polynomial label="eval" order="2" energyUnit="eV" hasUncertainties="true">
      <promptProductKE> 1.7555e+8 4e+5 -0.4566 0.04566 0.0 0.0</promptProductKE>
      ...
  </polynomial></fissionEnergyReleased>...</outputChannel>
```

- This representation mingles polynomial coefficients with uncertainties – confusing!

Changes to fissionEnergyReleased: now treated as a form of Q-value, with clear distinction between coefficients and uncertainties

- New proposal (implemented in development version) is more verbose, hopefully more clear:

```
<Q>
 <fissionEnergyReleased label="eval">
  <approximateQ>
   <constant1d constant="1.98902e+8"></approximateQ>
  <promptProductKE>
   <polynomial1d domainMin="1e-5" domainMax="2e7">
    <axes>...</axes><values length="3">1.7555e8 -0.4566 0</values>
    <uncertainties>
     <uncertainty>
      <covarianceMatrix type="absolute">
       <axes>...</axes>
       <array shape="3,3" compression="diagonal">
        <values>4e5 0.04566 0</values></array></covarianceMatrix>
...
```

Explicit markup for Coulomb elastic scattering

- GND-1.7 is direct translation from ENDF: MF=3 data are treated as cross sections, MF=6 treated as probability distributions

  - For Coulomb elastic, cross section is infinite, ENDF really stores the nuclear + Coulomb interference portion

  - Needs to be more clear in GND

For MF=6 LAW=5 LTP=1 use 'NuclearAmplitudeExpansion':

```
<reaction outputChannel="H1 + He3" ENDF_MT="2">
  <dCrossSection_dOmega>
    <NuclearAmplitudeExpansion label="eval" productFrame="centerOfMass">
      <Nuclear><XYs2d>...</XYs2d></Nuclear>
      <RealInterference><XYs2d>...</XYs2d></RealInterference>
      <ImaginaryInterference><XYs2d>...</XYs2d></ImaginaryInterference>
    </NuclearAmplitudeExpansion>
  </dCrossSection_dOmega>
  <crossSection>
    <reference label="eval" xlink:href="../../dCrossSection_dOmega"/></crossSection>
  <outputChannel>
     ...
     <product name="H1" label="H1">
       <distribution>
         <reference label="eval" xlink:href="../../../../dCrossSection_dOmega...
```

For MF=6 LAW=5 LTP=12-15, use 'NuclearPlusCoulombInterference':

```
<reaction outputChannel="H1 + Fe56" ENDF_MT="2">
  <dCrossSection_dOmega>
    <NuclearPlusCoulombInterference label="eval">
      <effectiveCrossSection>
        <XYs1d>...</XYs1d></effectiveCrossSection>
      <effectiveDistribution pid="H1" productFrame="centerOfMass">
        <XYs2d>...</XYs2d></effectiveDistribution>
    </NuclearPlusCoulombInterference>
  </dCrossSection_dOmega>
  <crossSection>
    <reference label="eval" xlink:href="../../dCrossSection_dOmega"/></crossSection>
  <outputChannel>
    ...
    <product name="H1" label="H1">
      <distribution>
        <reference label="eval" xlink:href="../../../../dCrossSection_dOmega...
```

- Ignoring LTP=2 (‘residual cross section expansion’) for now since it appears to be an unused option

Discussion of using HDF5 to store evaluations in binary

- HDF5 is a binary hierarchical data format, seems like a natural choice for binary nuclear data.
  - Under active development, comes with access routines in C, Fortran, Java, Python, ... (likely others as well)
- May offer performance advantages: parallel read capability, etc.
- Translating XML – HDF is simple, but we were initially discouraged by much larger file sizes: each new <element> in XML was transformed to ~1 kB in HDF

Recent improvements to HDF make nesting less expensive (although still more expensive than equivalent XML data)

- Space usage has improved in latest libhdf5 (version 1.8.*) :
  - for file with many ‘sister’ nodes, originally 0.7 kB/group, now ~0.31 kB/group
  - for ‘nested’ file, originally ~1 kB/group, now ~0.144 kB/group
  - thanks to Austin McCartney for pointing out these changes

- Even with these changes, deep nesting can be expensive
  - Consider example from ENDF-VII.1 n-074_W_183 (next slide)

- Deeper nesting may impact HDF read performance too, still need to do testing on that.

Impact of nesting on double-differential distribution (W-183 MF=6 LANG=1 MT=91):

```
<XYs3d>
  <axes>...</axes>
  <XYs2d value="489685.0">
    <Legendre value="0.0"><values length="1">2e5</values></Legendre>
    <Legendre value="1e-5"><values length="1">0</values></Legendre></XYs2d>
  <XYs2d value="500000.0">
    <Legendre value="0.0"><values length="1">2e5</values></Legendre>
    <Legendre value="1e-05"><values length="1">0</values></Legendre></XYs2d>
  <XYs2d value="520000.0">
    <Legendre value="0.0"><values length="3">3.2155e-5 1.63337e-7 -2.1354e-7</values></Legendre>
    <Legendre value="62100.0"><values length="3">1.25492e-8 1.23669e-16 0</values></Legendre>
    ...
    <Legendre value="496829.99999"><values length="3">5.04977e-13 3.33006e-21 0</values></Legendre>
    <Legendre value="496830.0"><values length="3">0 0 0</values></Legendre></XYs2d>
  ...
  <XYs2d value="150000000.0">...</XYs2d>
</XYs3d>
```

Impact of nesting on double-differential distribution (W-183 MF=6 LANG=1 MT=91):

```
<XYs3d>
  <axes>...</axes>
  <XYs2d value="489685.0">
    <Legendre value="0.0"><values length="1">2e5
    <Legendre value="1e-5"><values length="1">0<
  <XYs2d value="500000.0">
    <Legendre value="0.0"><values length="1">2e5
    <Legendre value="1e-05"><values length="1">0
  <XYs2d value="520000.0">
    <Legendre value="0.0"><values length="3">3.2
    <Legendre value="62100.0"><values length="3"
    ...
    <Legendre value="496829.99999"><values leng
    <Legendre value="496830.0"><values length="3">0 0 0</values></Legendre></XYs2d>
  ...
  <XYs2d value="150000000.0">...</XYs2d>
</XYs3d>
```

Distribution contains
- 74 <XYs2d> elements, one per incident energy
- 3815 <Legendre> elements, one per outgoing E

HDF about 4 times larger than xml with latest libhdf5 (2.8M vs. 745 kB)

Exercise: wrap each <Legendre> in <curve>, each <XYs2d> inside <surface> and <XYs3d> inside <manifold>
- xml size increases by 7.7%,
- HDF size increases by 25%

Impact of nesting on double-differential distribution (W-183 MF=6 LANG=1 MT=91):

```
<XYs3d>
  <axes>...</axes>
  <XYs2d value="489685.0">
    <Legendre value="0.0"><values length="1">2e5</values></Legendre>
    <Legendre value="1e-5"><values length="1">0</values></Legendre></XYs2d>
  <XYs2d value="500000.0">
    <Legendre value=
    <Legendre value=
  <XYs2d value="520
    <Legendre value="0.0"><values length="3">3.2155e-5 1.63337e-7 -2.1354e-7</values></Legendre>
    <Legendre value="62100.0"><values length="3">1.25492e-8 1.23669e-16 0</values></Legendre>
    ...
    <Legendre value="496829.99999"><values length="3">5.04977e-13 3.33006e-21 0</values></Legendre>
    <Legendre value="496830.0"><values length="3">0 0 0</values></Legendre></XYs2d>
  ...
  <XYs2d value="150000000.0">...</XYs2d>
</XYs3d>
```

Note that Legendre coefficients are not always unitless. Here they represent the distribution for both outgoing energy *and* angle, with units of 1/eV

Discussion of XML schemas

- Suggestion from Austin and Jeremy:
  - The <applicationData> section is meant for storing application-specific data that may not be supported by all users
  - If a file containing <applicationData> is shared with other users, should we require an xml schema describing the application-specific data be distributed as well?
  - Pro: helps users add support for the new data (if they want)
  - Con: how do we enforce the rule?
- Also, should we provide RelaxNG schema as well as W3C?
  - More readable syntax, may not be as well supported by standard tools

Much of the discussion centered around low-level data containers

- Most of this afternoon is reserved for continuing that discussion, so just summarizing my overall impressions here:

  - We seem to be nearing agreement on how to organize data
    - still disagree about the role of <axes>, see next slide
  - Names of containers (XYs1d, XYs2d, regions1d, gridded1d, etc.): we aren't enthusiastic about these names, but they are the most descriptive suggestions so far
  - Data containers need to balance simplicity and consistency... with flexibility
    - For example, when translating MF=4 LTT=3 need to switch representations from Legendre expansion to pointwise.

Main point of disagreement now seems to be ‘where to define the axes’

- LANL argument: low-level data containers define mathematical functions and should not contain any description of axes. Physical meaning, including axes definitions, should be supplied by the parent element that contains the LLDC

- LLNL argument: by adding axes in a math object, we get a physics object with no need for an outer physics wrapper. That reduces nesting and reduces the number of elements that need to be defined.

- Possible compromise: use a ‘template axes’ section to define default axes for all components.

- Expect lots more discussion this afternoon!