WPEC SG50: Developing an Automatically Readable, Comprehensive and Curated Experimental Nuclear Reaction Database

Amanda Lewis[1] Denise Neudecker[2] Arjan Koning[3]

15th International Conference on Nuclear Data for Science and Technology
July 24-29, 2022

[1] Naval Nuclear Laboratory
[2] Los Alamos National Laboratory
[3] International Atomic Energy Agency

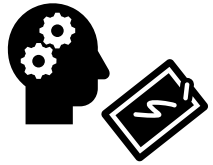
LA-UR-22-26522

“MEDUSAL”: Machine-readable Experimental Data User App & Library

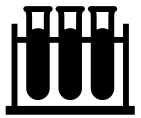
- MEDUSAL goal: to provide the best possible experimental input to support the direct integration of experimental data in the nuclear data pipeline
- The EXFOR database is our basis. The EXFOR database strives to document and preserve experimental nuclear reaction data true to its original publication.
- MEDUSAL is driven by users’ needs and requirements and has three layers to store more types of information than EXFOR

From Wikipedia, CC BY-SA 3.0

Database Users Trigger High Level Requirements

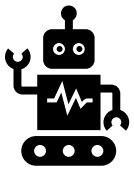


Users



- Nuclear Data Evaluator
- Experimentalist

- Model Development



- Applying ML/AI with database
- Nuclear data library validation

High Level Requirements

- Access Data
  - By search criteria or downloading entire library
  - Storing layers 1-3 & feeding user information in
- Format
  - Easy to read automatically for large amount of data
  - Unique, parsable metadata keys
  - Clear identification of observable
- Data Treatment
  - Renormalization to current standards
  - Estimating missing and total covariances
- Knowledge Management
  - Able to store judgements and versions as used in evaluations
  - Easy for users to upload their judgement and versions

Metadata Requirements and Specifications

6.5 Samples

The `samples` container holds information about all samples measured in the experiment. This includes any sample used as a monitor reaction, to determine the flux, etc. This does not include the targets used in the production of the incident particles.

The requirements of the `samples` container are:

6. `samples` requirements

   6.1 Hold an unlimited number of samples, each in a `sample` container.

   6.2 Identify the configuration of the samples if they are measured simultaneously.

The requirements of the `sample` container are:

7. `sample` requirements

   7.1 Identify the use of the sample in the experiment.

   7.2 If different incident energies are seen by different samples, record incident energy seen and the incident angle.

   7.3 Identify the position of the sample if it measured simultaneously with other samples.

   7.4 Identify the isotopic composition of the sample and quantify any corrections for impurities.

   7.5 Identify the sample material, geometry, and phase.

   7.6 Identify the production method of the sample.

6 Samples

The `samples` container is a list of `sample` containers, each of which describe a single sample used (such as the target(s), and any number of normalization or reference samples).

```
"samples": [
            {sample container},
            {sample container}
           ]
```

The keys allowed in each `sample` container are:

6.1 Sample type: `use`

A data type to describe what the sample is used for - the measurement of the observable, the monitor or standard, or the flux directly.

Specifications for: `use`

- Data type: `choice:list`, which can be one of the allowed values for the `use type`, which are given in Table 10.

JSON examples of `use`

```
{"use": "observable"}
```

6.2 Sample type: `incident_energy`

A data type to hold the incident energy seen by the sample, if different samples are irradiated with different incident energies. This can occur in measurements with multiple samples with charged particle beams which degrade in energy, or neutron generators that have different energy spectra at different angles. If the sample sees the entire energy spectrum described in the `incident_particle` container, this key should not be used.

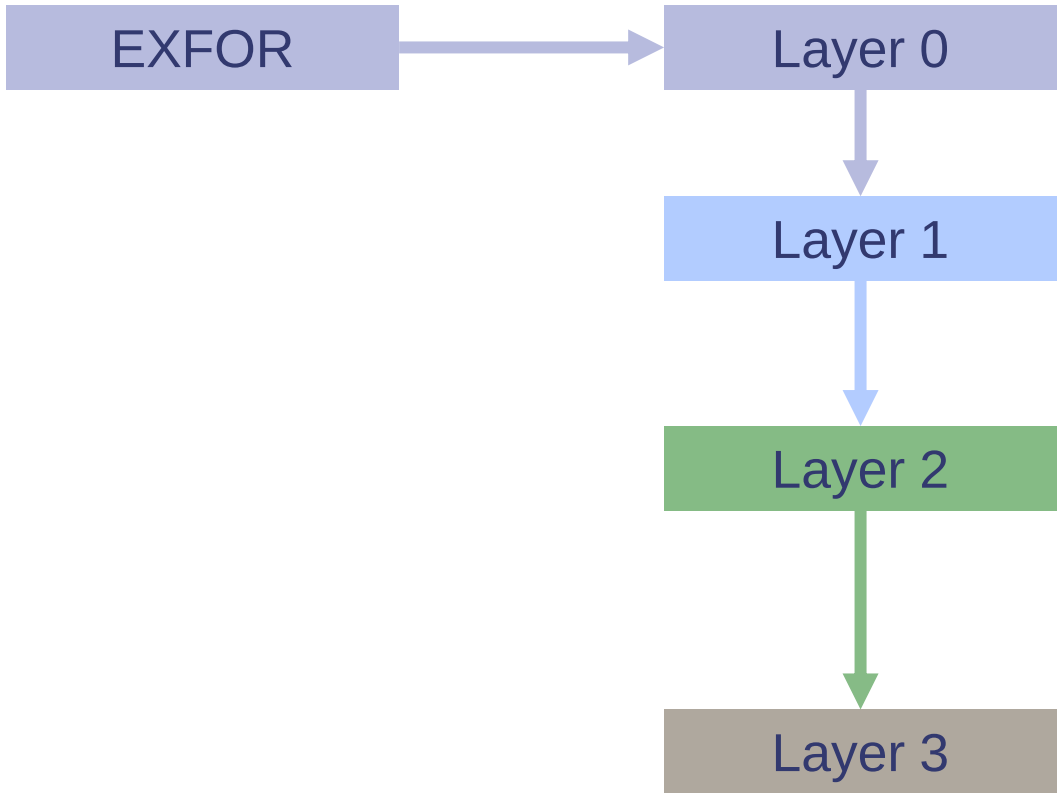
Specifications for: `incident_energy`

- Data type: `measured quantity type`.

JSON examples of `incident_energy`:

```
{"incident_energy": {
                     "value": [2.24e6],
                     "unit": "eV",
                     "type": "mean"
                    }
}
```

Our database will build on EXFOR and store “subjective” corrections to the data sets



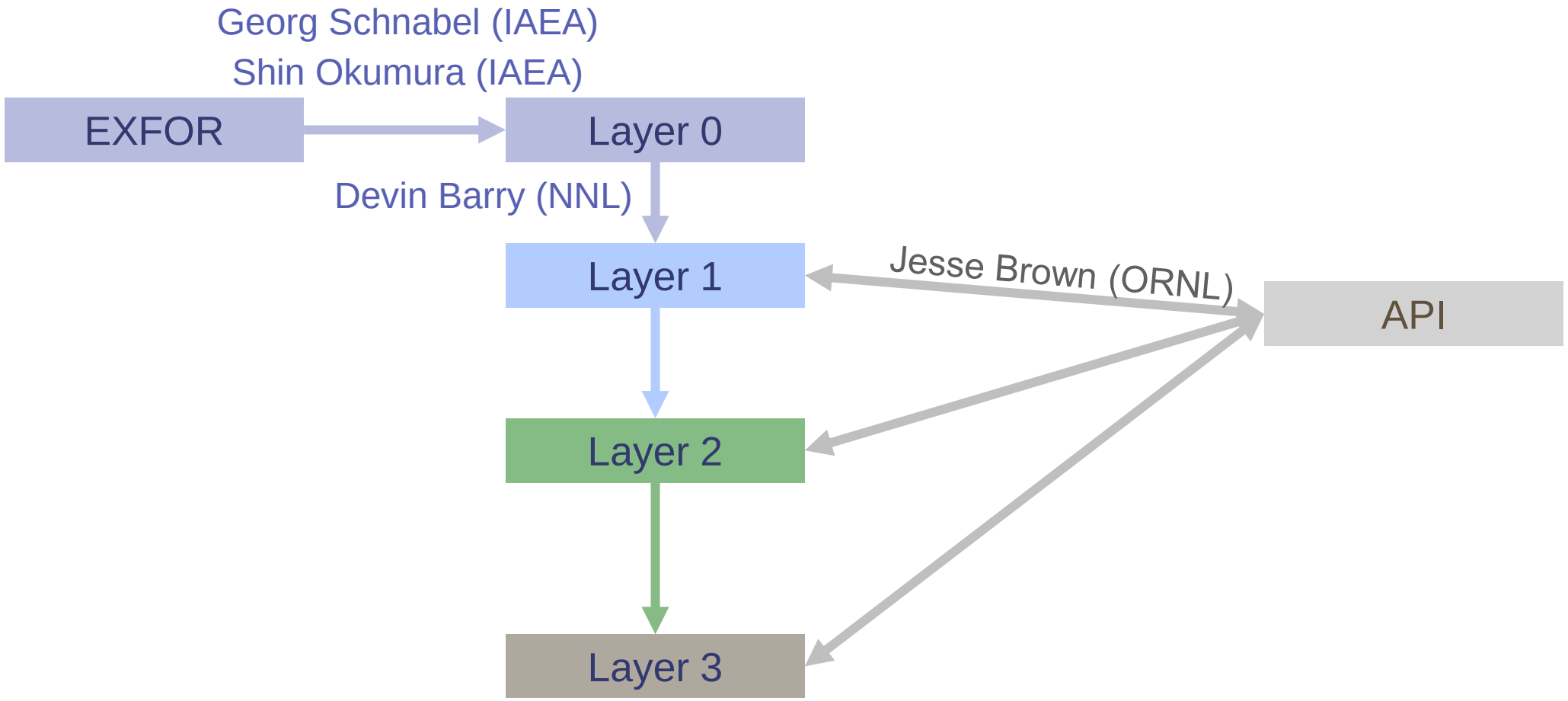
Same information as in EXFOR, but in SG50 format

Information parsed from free text, consistent units, distinction between missing and not applicable keywords

“Objective” Corrections – updating standards, flagging missing uncertainty sources via measurement uncertainty templates, identifying outliers, correcting mistakes in the published experimental report

“Subjective” Corrections – including expert judgement, updating uncertainties via templates - expanding on the current correction system

Our database will build on EXFOR and store “subjective” corrections to the data sets



WPEC SG50 will deliver to the community:

- Database use cases, requirements, and specifications for an automatically readable and curated experimental reaction database
- Example files in JSON format
- Python codes to create example files
- Feedback and input from across the community
- Documentation and examples to enable the creation of a new database

Acknowledgements

- WPEC SG50 has members from all over the international nuclear data community

- The example codes and files are created by the development group
  - Devin Barry (NNL)
  - Jesse Brown (ORNL)
  - Shin Okumura (IAEA)
  - Georg Schnabel (IAEA)