Automatic generation of specifications

D. Brown (BNL)

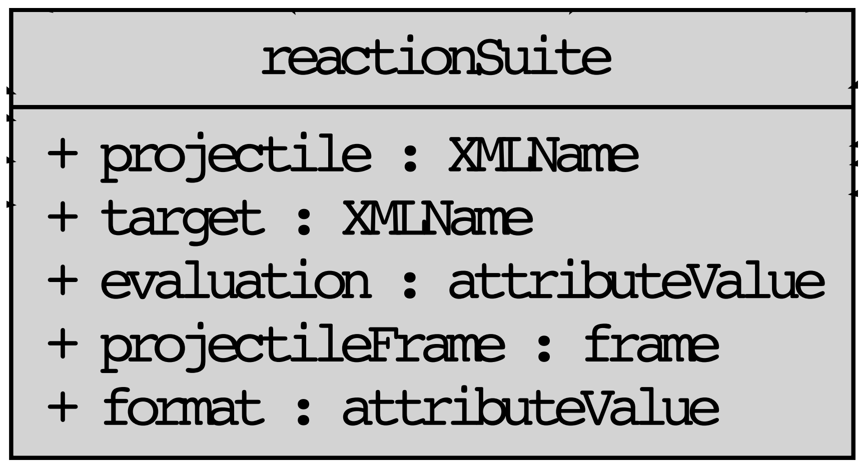
Multiple sources of formatting information

- **Specifications documents** *(GPDC, documentation, top level, PoPs)* — all partially out of date with GNDS-1.9
- **Requirements document** — only source for planned, but not yet properly implemented formats (TSL, FPY)
- **XML schema file** *(gnd.xsd)* — partially out of date with GNDS-1.9, has no descriptive information
- **XML files themselves** — most up to date, including examples of current (not final) implementation of TSL, FPY

**All provide partial information and complement one another.
There is no authoritative reference.**

A technological solution

Develop data structures that contain all information required to describe format



```
reactionSuite
+ projectile : XMLName
+ target : XMLName
+ evaluation : attributeValue
+ projectileFrame : frame
+ format : attributeValue
```

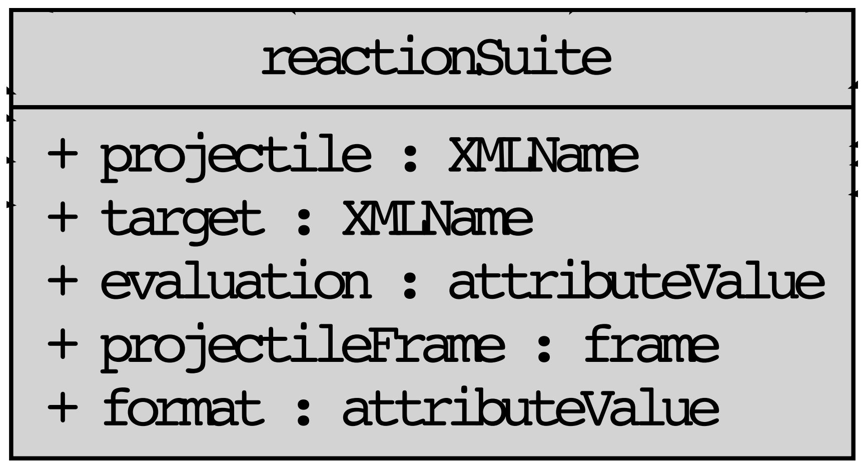
Include:

- Occurrence limits
- Required or not
- Root node or not
- Data type information
- List of child nodes
- Detailed descriptions coded in LaTeX

<u>grokGNDS.py</u>
attributes
nodes
(childNodes)

A technological solution

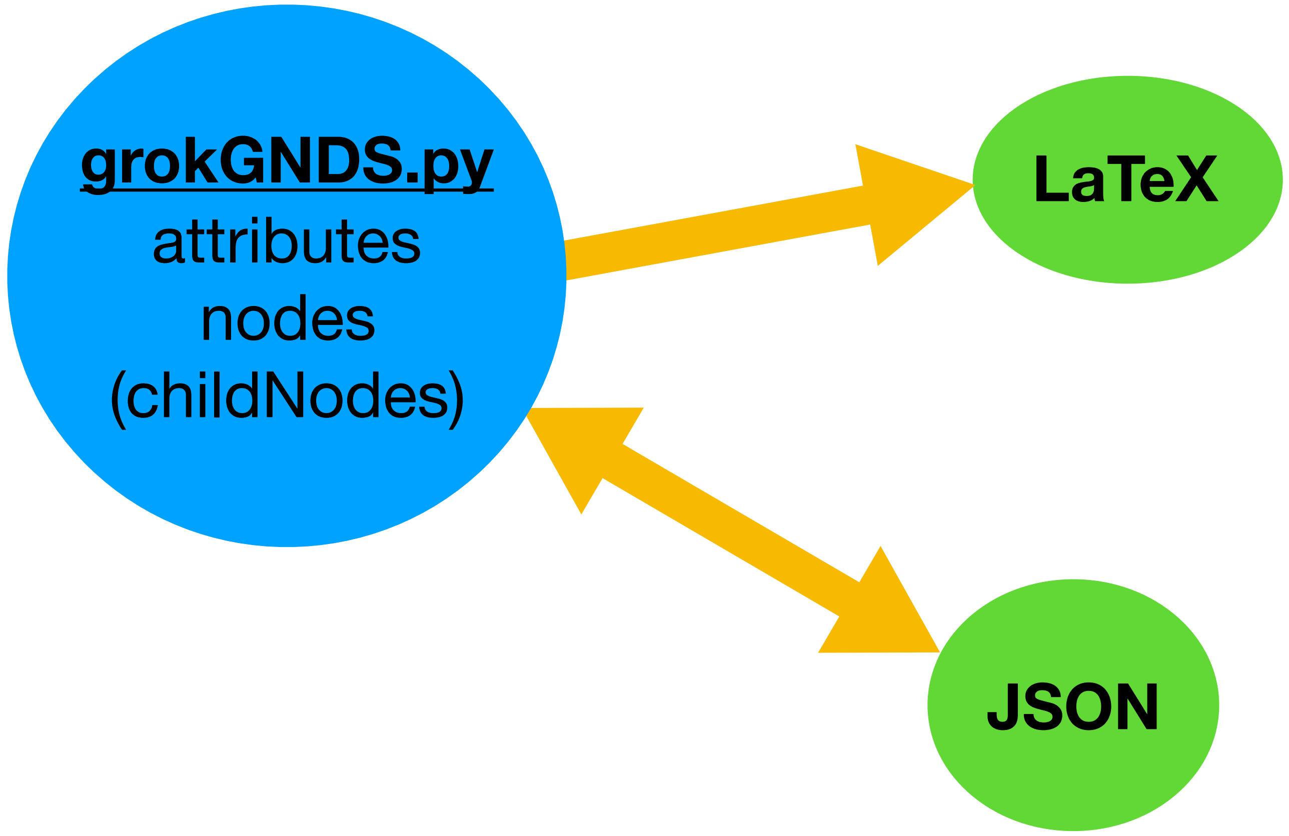
Develop data structures that contain all information required to describe format



```
reactionSuite
+ projectile : XMLName
+ target : XMLName
+ evaluation : attributeValue
+ projectileFrame : frame
+ format : attributeValue
```

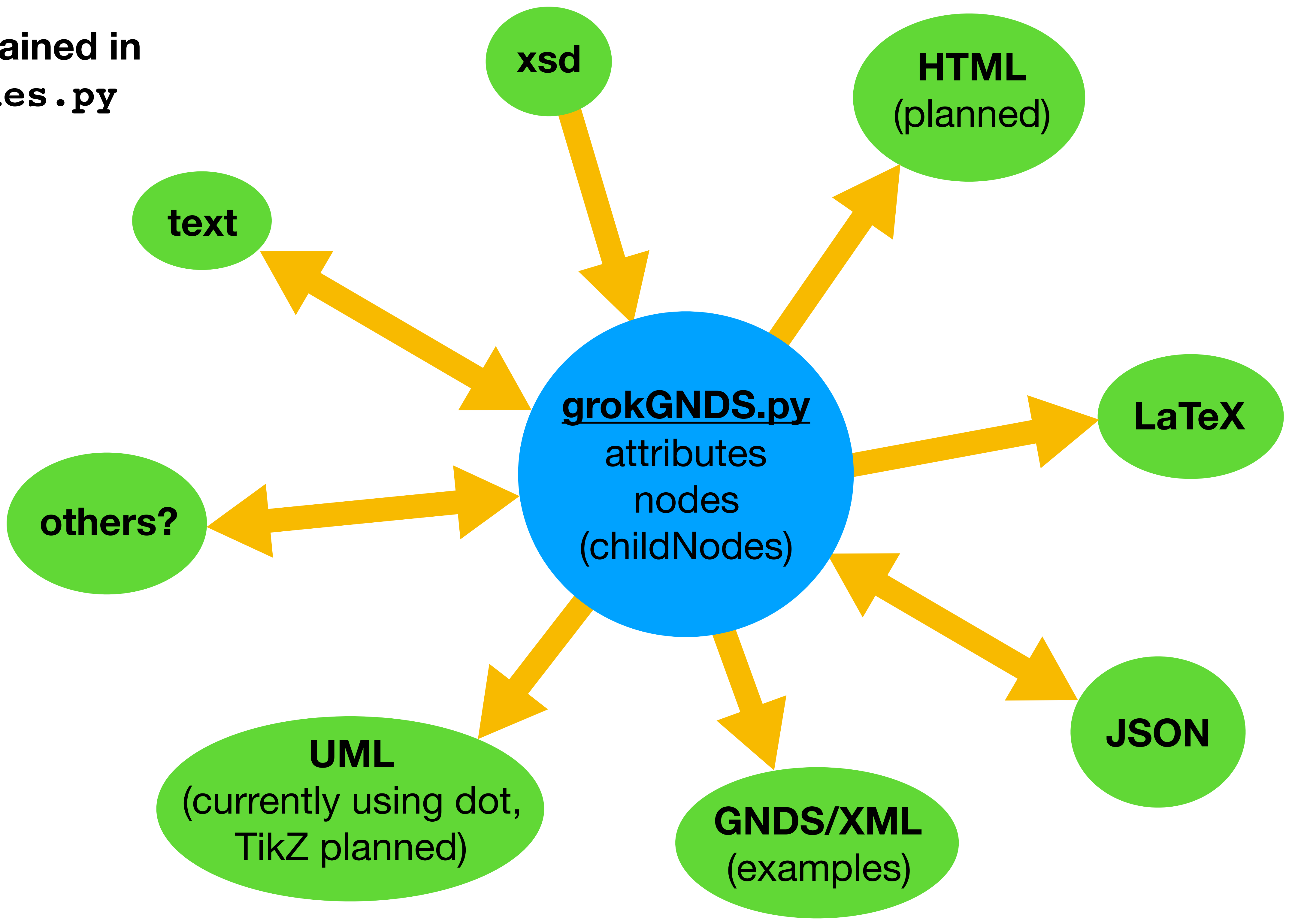
Include:
- Occurrence limits
- Required or not
- Root node or not
- Data type information
- List of child nodes
- Detailed descriptions coded in LaTeX

Additional functionality
- Read/write variety of formats
- LaTeX and/or UML output
- Updating functionality



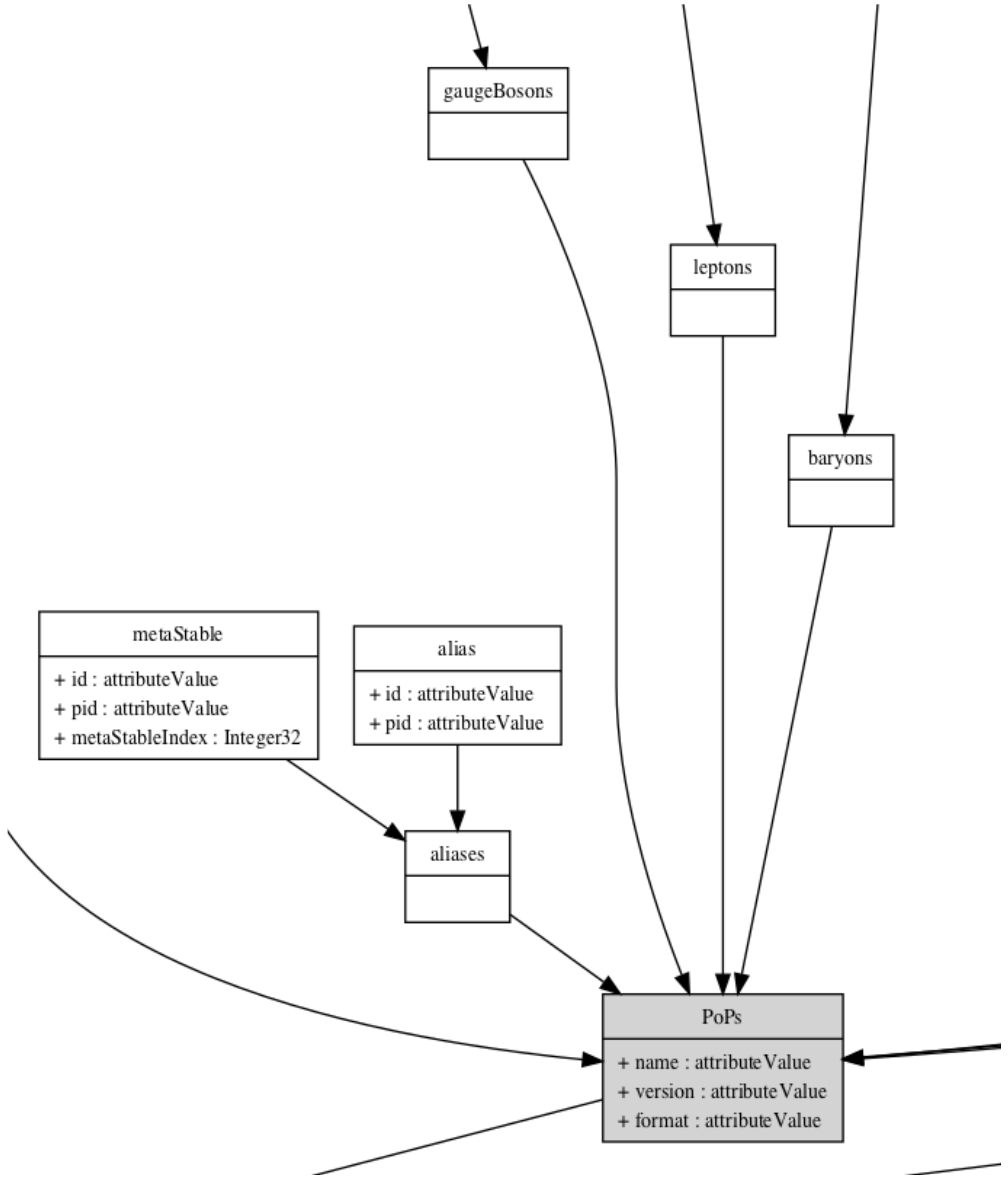
A technological solution

Contained in
`nodes.py`



Key ingredient: a “tree walker”

- Since we’ve designed a data hierarchy, we need need to crawl it to find out what is in it
- Standard computing algorithm: recursive “Tree walking”
- Very easy to implement
- As visit nodes in a given hierarchy, can update node attributes/children



Contained in `grokGNDS.py`

The plan

1. **Initialize database of formats with schema (gnd.xsd)**
2. **Crawl representative sample of XML files to update database**
    - Neutrons (w/ & w/o covariance, fission)
    - Charged particles
    - Photo nuclear
    - Decay
    - Fission product yields
    - Atomic data
    - Processed data
3. **Serialize output to JSON (or equivalent)**
4. **Update descriptions by hand using specifications draft documents**
5. **Serialize result to LaTeX files**
6. **Frame file can be used to organize specifications using \include{}**
7. **If develop xsd back-translator, then can keep specifications and xsd file in sync**

The plan

1. **Initialize database of formats with schema (gnd.xsd)**
2. **Crawl representative sample of XML files to update database**
   - Neutrons (w/ & w/o covariance, fission)
   - Charged particles
   - Photo nuclear
   - Decay
   - Fission product yields
   - Atomic data
   - Processed data
3. **Serialize output to JSON (or equivalent)**
4. **Update descriptions by hand using specifications draft documents**
5. **Serialize result to LaTeX files**
6. **Frame file can be used to organize specifications using \include{}**
7. **If develop xsd back-translator, then can keep specifications and xsd file in sync**

Workflow currently automated with `Makefiles`

The plan

1. Initialize database of formats with schema (gnd.xsd)
2. Crawl representative sample of XML files to update database
   - Neutrons (w/ & w/o covariance, fission)
   - Charged particles
   - Photo nuclear
   - Decay
   - Fission product yields
   - Atomic data
   - Processed data
3. Serialize output to JSON (or equivalent)
4. Update descriptions by hand using specifications draft documents

Contained in `makeSpecs.py`

5. **Serialize result to LaTeX files**
6. **Frame file can be used to organize specifications using \include{}**
7. **If develop xsd back-translator, then can keep specifications and xsd file in sync**

This is what we get

Specifications for `evaluated`

Node name: `evaluated` None
Attributes: The list of additional allowed attributes are:

- `label` [`XMLName`, **required**] None
- `library` [`attributeValue`, **required**] None
- `version` [`Empty`, **required**] None
- `date` [`date`, **required**] None

Child nodes: The list of additional allowed Child nodes are:

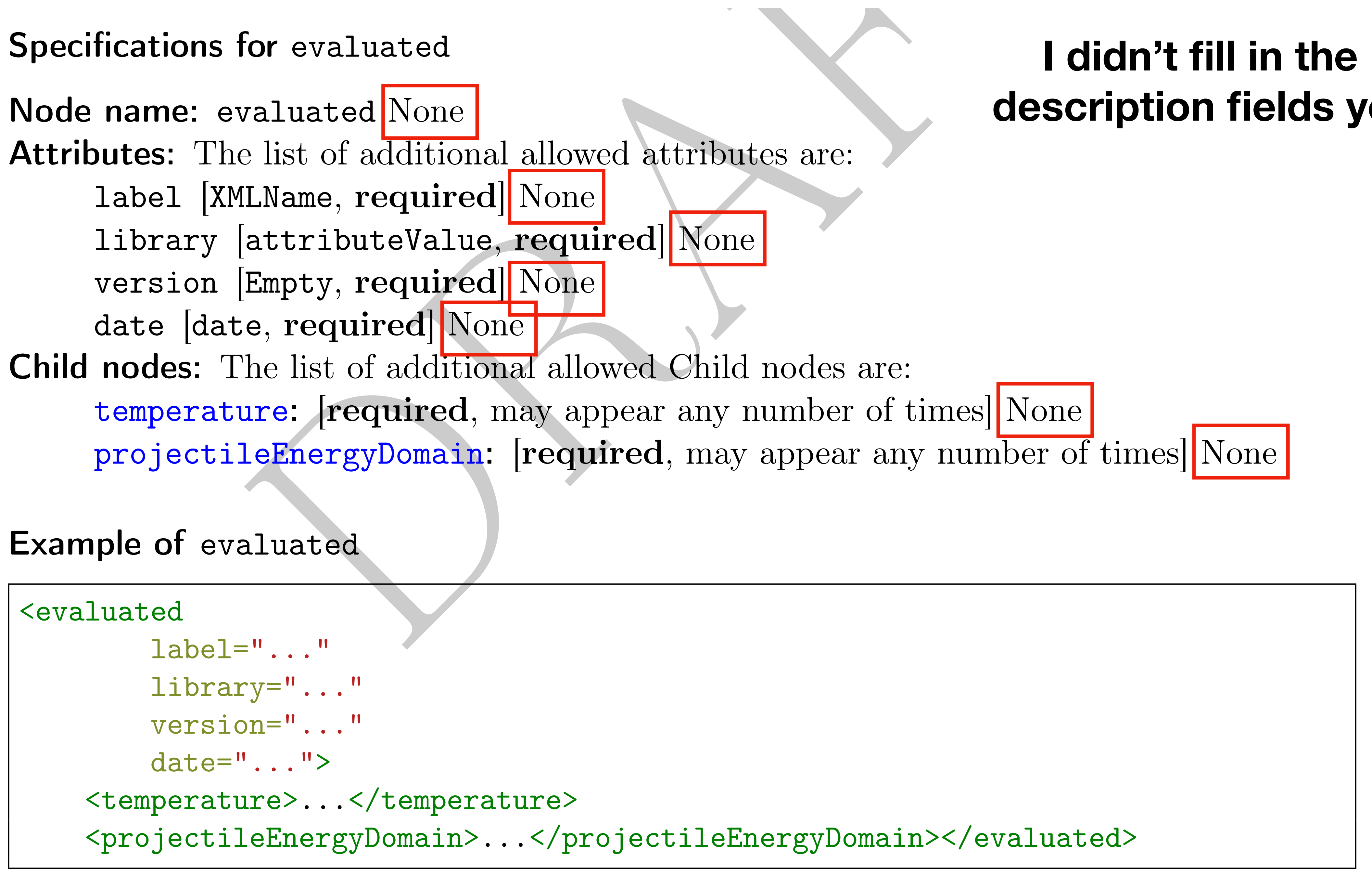
- `temperature`: [**required**, may appear any number of times] None
- `projectileEnergyDomain`: [**required**, may appear any number of times] None

Example of `evaluated`

```
<evaluated
        label="..."
        library="..."
        version="..."
        date="...">
    <temperature>...</temperature>
    <projectileEnergyDomain>...</projectileEnergyDomain></evaluated>
```

This is what we get

I didn't fill in the description fields yet



Specifications for `evaluated`

Node name: `evaluated` None

Attributes: The list of additional allowed attributes are:

- `label` [XMLName, **required**] None
- `library` [attributeValue, **required**] None
- `version` [Empty, **required**] None
- `date` [date, **required**] None

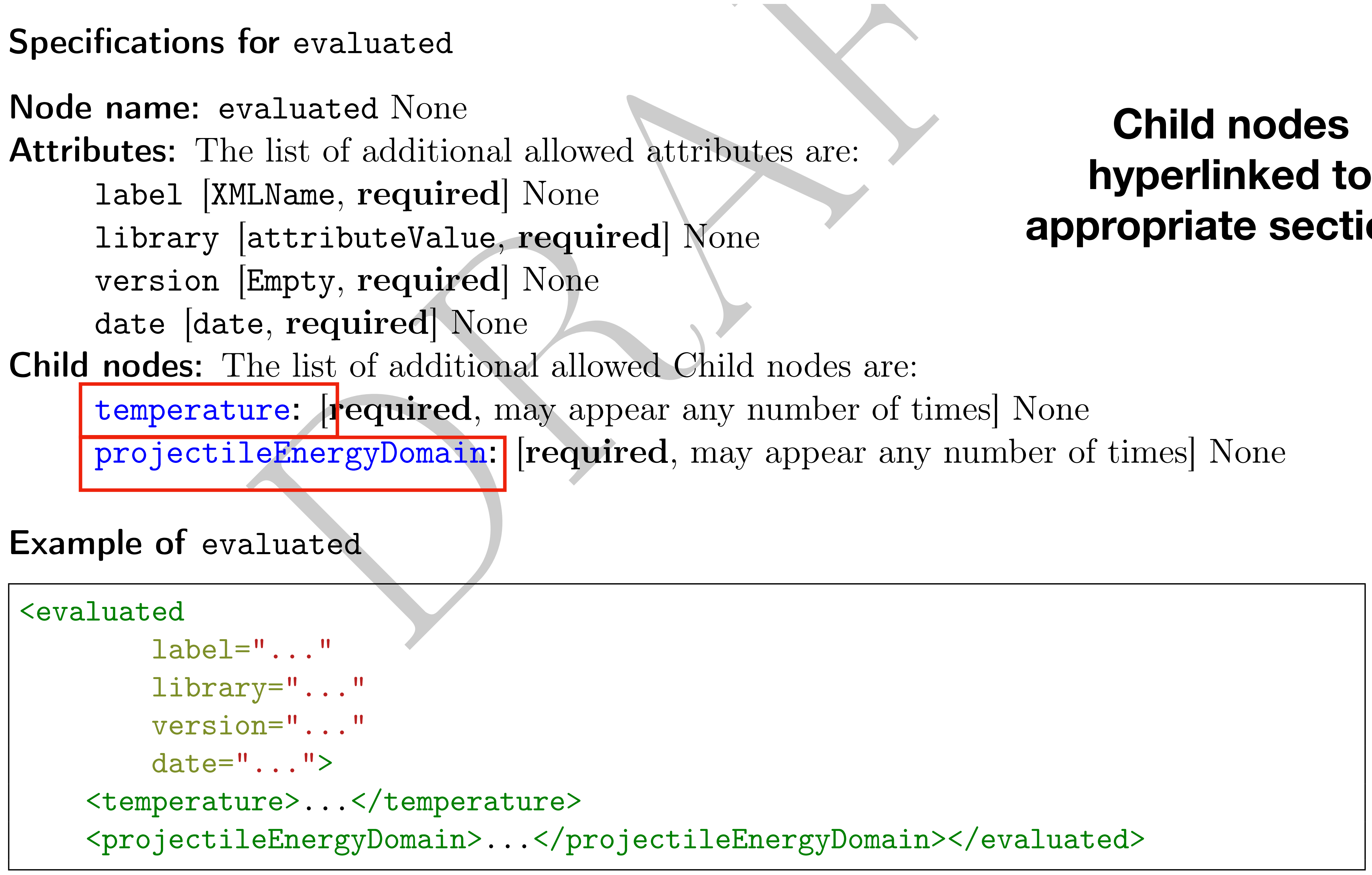
Child nodes: The list of additional allowed Child nodes are:

- `temperature`: [**required**, may appear any number of times] None
- `projectileEnergyDomain`: [**required**, may appear any number of times] None

Example of `evaluated`

```
<evaluated
        label="..."
        library="..."
        version="..."
        date="...">
    <temperature>...</temperature>
    <projectileEnergyDomain>...</projectileEnergyDomain></evaluated>
```

This is what we get



Specifications for `evaluated`

Node name: `evaluated` None
Attributes: The list of additional allowed attributes are:

- `label` [`XMLName`, **required**] None
- `library` [`attributeValue`, **required**] None
- `version` [`Empty`, **required**] None
- `date` [`date`, **required**] None

Child nodes: The list of additional allowed Child nodes are:

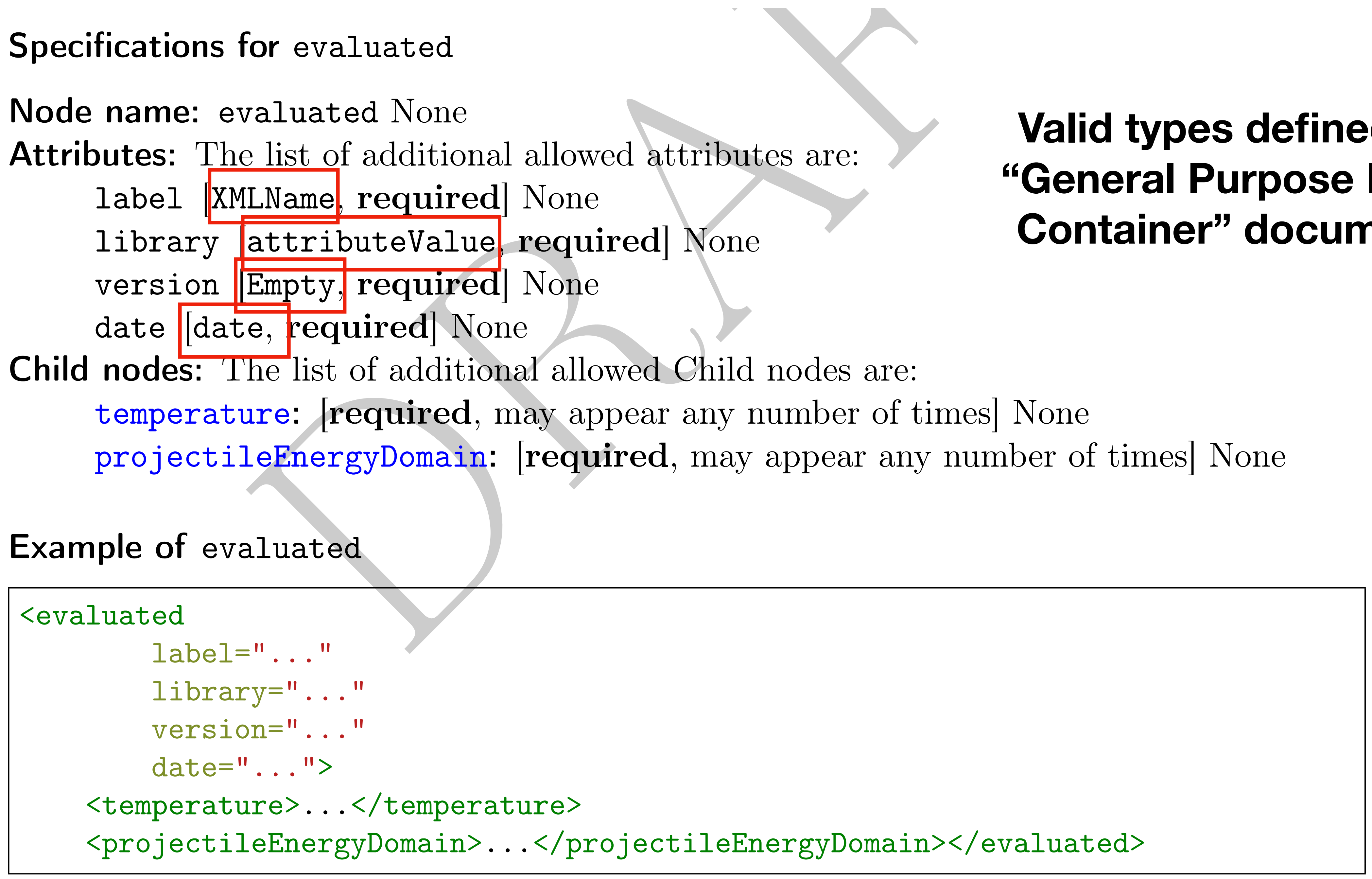
- `temperature`: [**required**, may appear any number of times] None
- `projectileEnergyDomain`: [**required**, may appear any number of times] None

Example of `evaluated`

```
<evaluated
        label="..."
        library="..."
        version="..."
        date="...">
    <temperature>...</temperature>
    <projectileEnergyDomain>...</projectileEnergyDomain></evaluated>
```

Child nodes hyperlinked to appropriate sections

This is what we get



Specifications for `evaluated`

Node name: `evaluated` None

Attributes: The list of additional allowed attributes are:

- `label` [`XMLName`, **required**] None
- `library` [`attributeValue`, **required**] None
- `version` [`Empty`, **required**] None
- `date` [`date`, **required**] None

Child nodes: The list of additional allowed Child nodes are:

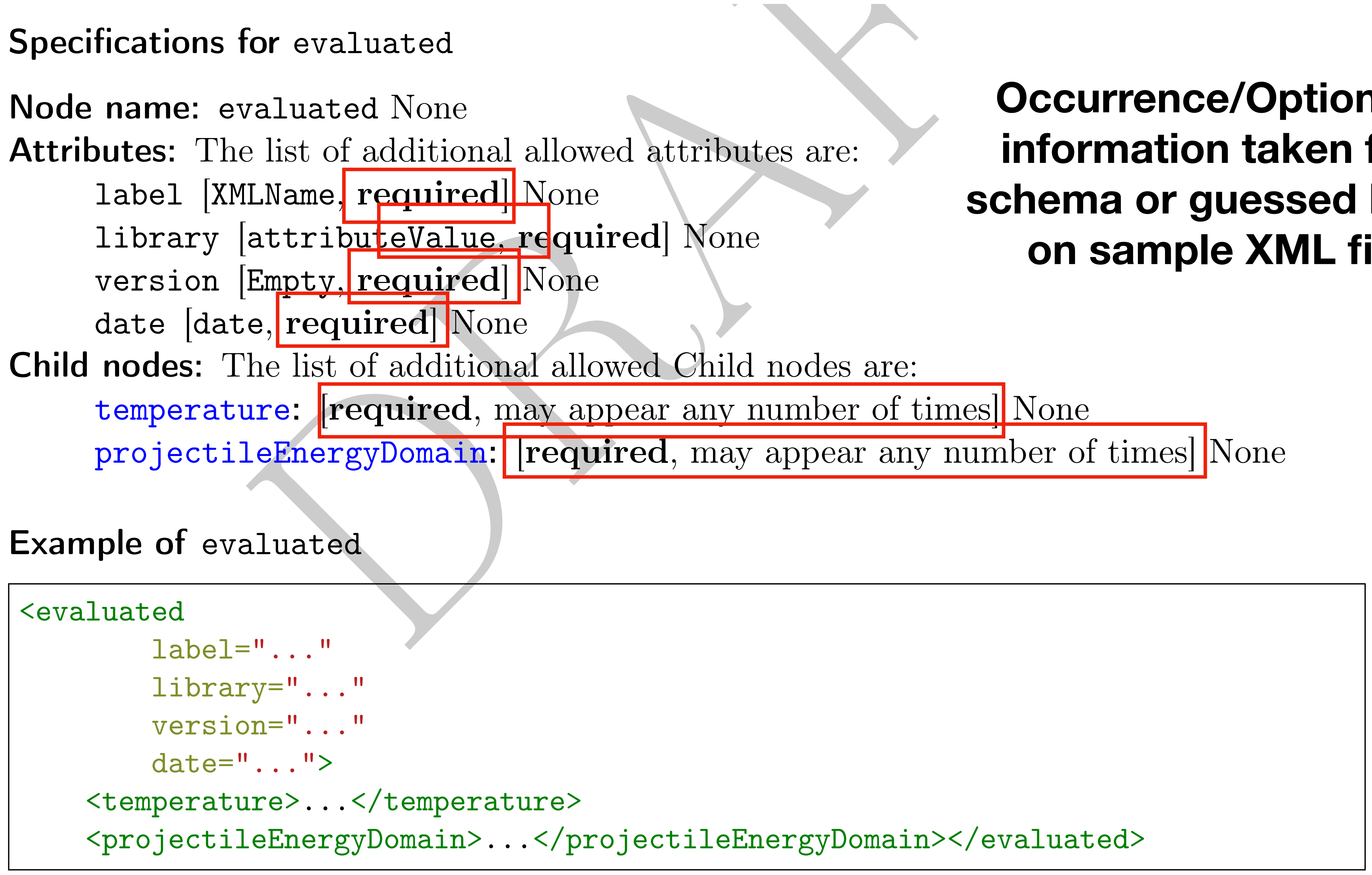
- `temperature`: [**required**, may appear any number of times] None
- `projectileEnergyDomain`: [**required**, may appear any number of times] None

Example of `evaluated`

```
<evaluated
        label="..."
        library="..."
        version="..."
        date="...">
    <temperature>...</temperature>
    <projectileEnergyDomain>...</projectileEnergyDomain></evaluated>
```

Valid types defined in "General Purpose Data Container" document

This is what we get



Specifications for `evaluated`

Node name: `evaluated` None
Attributes: The list of additional allowed attributes are:

- `label` [XMLName, **required**] None
- `library` [attributeValue, **required**] None
- `version` [Empty, **required**] None
- `date` [date, **required**] None

Child nodes: The list of additional allowed Child nodes are:

- `temperature`: [**required**, may appear any number of times] None
- `projectileEnergyDomain`: [**required**, may appear any number of times] None

Example of `evaluated`

```
<evaluated
        label="..."
        library="..."
        version="..."
        date="...">
    <temperature>...</temperature>
    <projectileEnergyDomain>...</projectileEnergyDomain></evaluated>
```

Occurrence/Optionality information taken from schema or guessed based on sample XML files

Special cases: TSL & FPY

- Both focus of Sub Groups with new and moderately complex requirements that go far beyond ENDF-6
- Neither focus of BNL/LANL/LLNL/ORNL efforts to date
- Quick-n-dirty implementation basically quick translation of ENDF, adding no new functionality, mainly to meet ENDF/B-VIII.0 release needs
- Argue for proper implementation of these formats in next GNDS version