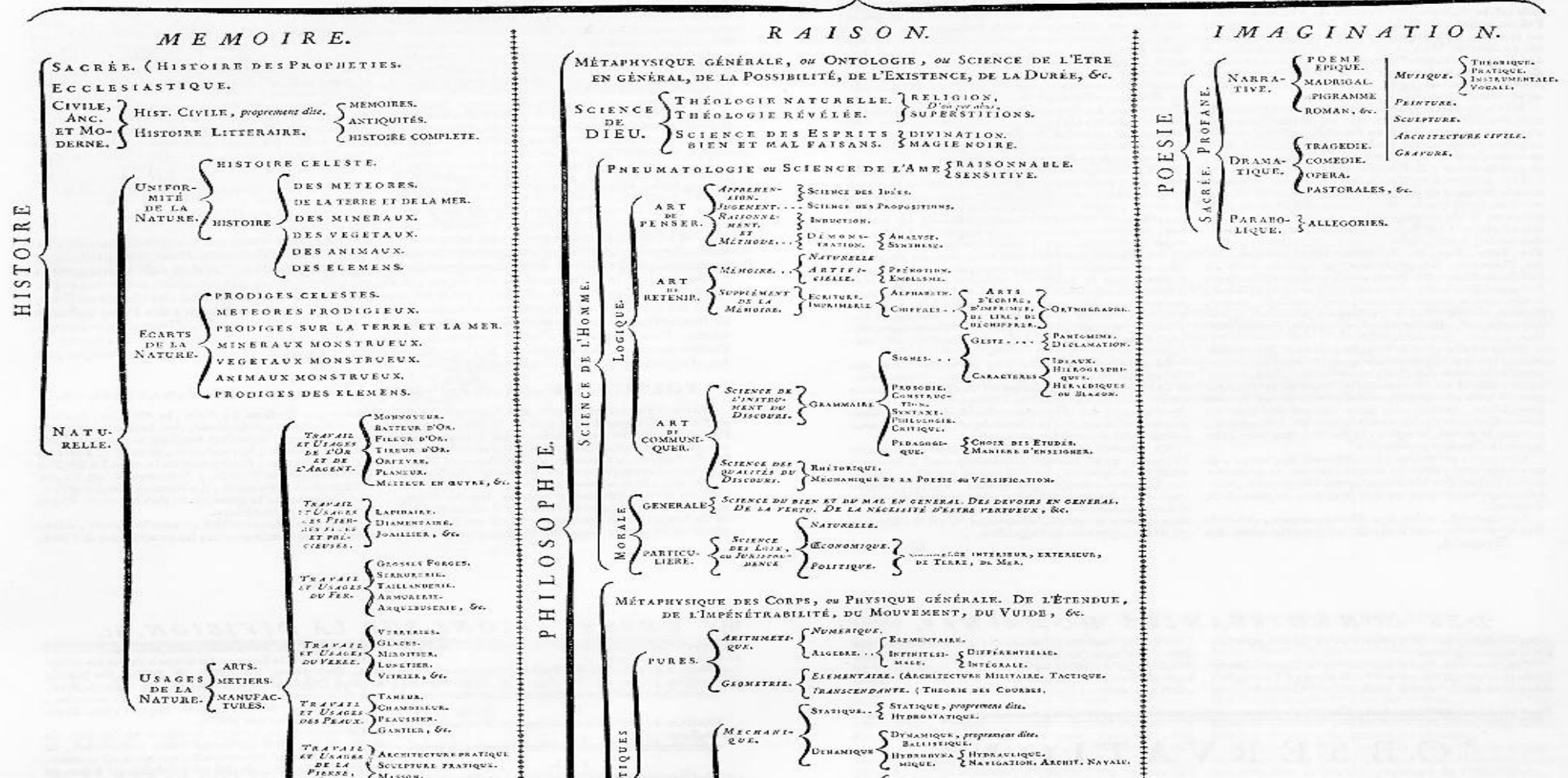


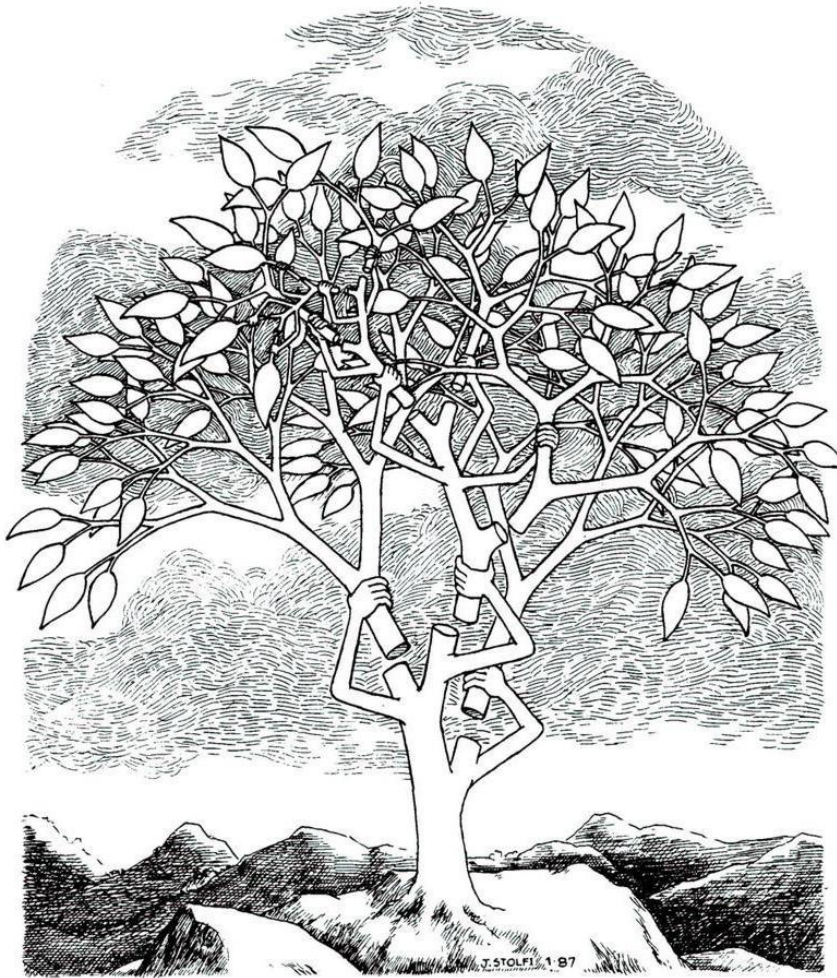
Yesterday's Technology for Tomorrow!

Data Trees for Nuclear Data

* *S Y S T È M E F I G U R É* *D E S C O N N O I S S A N C E S H U M A I N E S.* E N T E N D E M E N T.



Structure of a Tree



- A Tree has a trunk
- From the trunk comes a Branch
- That Branch can have one or more smaller Branches
- The smallest Branches end in Leaves

Definition of a Data Tree

- “In computer science, a tree is a widely-used data structure that emulates a hierarchical tree structure with a set of linked nodes” Wikipedia
- **IMPORTANT DISTINCTION:** ENSDF is already loosely structured as TREE but it isn't a DATA TREE
- A DATA TREE is a specific way of storing data in a linked list

Structure of a Data Tree

- A data tree has the familiar structure of an actual tree, hence the name
- Data Trees have large branches, smaller branches and terminate in leaves
- The leaves are the primitive data types, integers, doubles, strings, where the data resides
- The branches are the superstructure and don't themselves contain data, they're the (important) scaffolding

What Does a Data Tree Look Like?

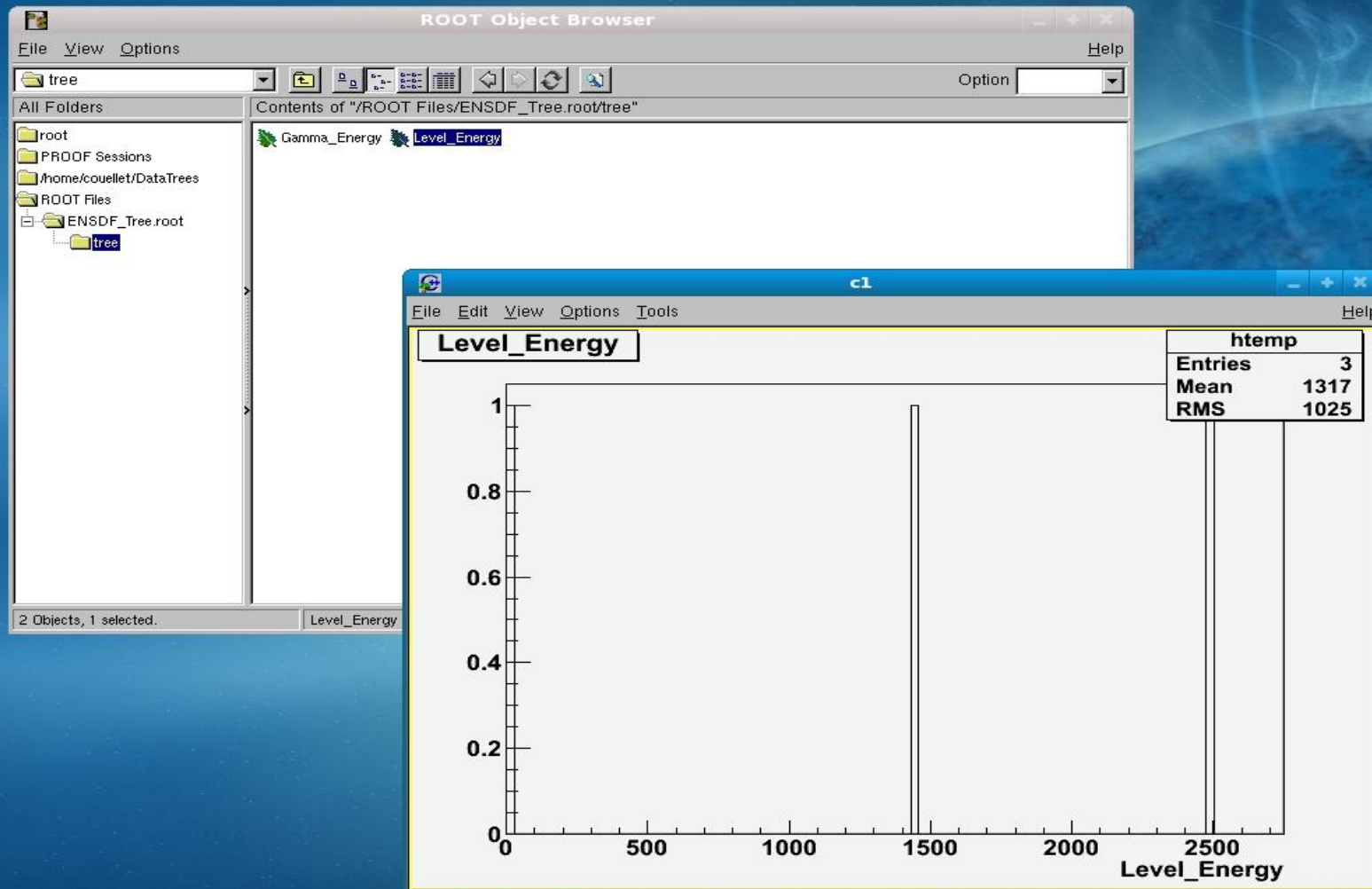
Role of the Interpreter Program

- The data is structured in a linked list, it is not a-priori intelligible if you simply open the tree file in a text editor
- You need a program to access and interpret the tree
- The interpreter can be a complex Graphical User Interface (GUI) or a simple command line access to the data

Data Tree Made in ROOT as Viewed in a Text Editor

[illegible]

Data Tree made in ROOT as viewed through the CINT interpreter



We Are Currently in a Land War with our Data: We Desperately Need TREE Power!



```

32AR Q -22620 SY 21560 SY 2420 50-8.70E3 16 2009AUZZ,2003AU03
32P L 16.64E3 74 KEV 14
32S CG RI(A)$ From 1972C013
32S CG RI(B)$ From 1976VE03
32S DG M,MR$ Rose and Brink: 1974VI02, 1973VE08, 1981HE09, 1976VE03, 1972C013
32S 2DG 1972C013; Ferguson and Rutledge: 1963TE01
32S CL E$ Values for levels up to 10756 KEV are from primary |g
32S 2CL transitions. Resonance energy levels are derived E(p)(cm)+
32S 3CL S(p), where S(p)= 8863.78 KEV 21 (2003Au03)
32S DL $ E(p)(cm)=31/32(E(p)(lab))
32S PN
32S L 0 0+
32S L 2230.5 3 2+ 152 FS 17
32S CL E$ from 1972C013
32S CL J$ from |g(|q) 1973VE08
32S DL E$ 2230.0 3 (1974VI02), 2230.5 3 (1972C013), 2231.7 10 (1971IN02),
32S 2DL 2231.1 10 (1973VE08)
32S CL T$ weighted average of: |t=252 FS 40 (1998KA31), 195 FS 70 (1974CH09),
32S 2CL 185 FS 75 (1972C012), 350 FS 60 (1971IN02), 260 PS 80 (1969TH03),
32S 3CL 175 FS 30 (1971RE15)
32S G 2231.7 10 100
32S CG E$ from (1971IN02)
32S DG RI$ 100 (1975B042), 100 (1972C013), 100 (1974VI02), 100 (1972LE29)
32S CG $ A2=+0.18 8, A2=+0.12 4, A4=+0.27, A2=+0.32 3, A4=+0.65,
32S 2CG A2=+0.26 (1969PI10), |g|g(|q): A2=-0.16 15, A4=+0.99 17, A2=+0.17 9,
32S 3CG A4=0.00 10 (1963SP03),
32S 3CG A2=+0.081 27, A4=-0.098 27 (1970F013)

```


Reasons to Use a Data Tree

DATA INPUT IS NOT CONSTRAINED BY FORMAT:

- The user inputs the leaves into the linked list and does not edit the tree file directly, format is the job of the interpreter program
- Examples: No 80 character limits, no repetition of nucleus identifier, etc...

```
32S CL T$ weighted average of: |t=252 FS 40 (1998KA31), 195 FS 70 (1974CH09),  
32S 2CL 185 FS 75 (1972C012), 350 FS 60 (1971IN02), 260 PS 80 (1969TH03),  
32S 3CL 175 FS 30 (1971RE15)
```

With an interpreted Tree it could be:

“Input T1/2 (value,error,units)”

245,40,FS;

195, 70, FS; ...

32S_tree->Fill

Reasons to Use a Data Tree

FORMAT DOES NOT CONSTRAIN THE DATA

- A data tree can always be expanded upon, branches and leaves can always be added as needed
- Examples: Data currently in comments (A2,A4) can be given their own branch, values can be quoted correctly in full precision (ex: Q-value)

```
32S CG $ A2=+0.18 8, A2=+0.12 4, A4=+0.27, A2=+0.32 3, A4=+0.65,  
32S 2CG A2=+0.26 (1969PI10), |g|g(|q): A2=-0.16 15, A4=+0.99 17, A2=+0.17 9,  
32S 3CG A4=0.00 10 (1963SP03),
```

```
32AR Q -22620 SY 21560 SY 2420 50-8.70E3 16 2009AUZZ,2003AU03  
32P L 16.64E3 74 KEV 14 A
```

With an interpreted Tree it could be:
32S_tree->AddBranch(A2,double);

Reasons to Use a Data Tree

DATA IN A DATA TREE ARE COMPUTATIONALLY ACCESSIBLE

- The elements of the tree can be accessed, added, subtracted, averaged, modified in any computational way with EASE
- Examples: No more scrolling - select the 32nd excited state multi-polarity with few keystrokes. Q-value changes, adjust all resonances trivially

With an interpreted Data Tree it could be:

```
32S_tree->GetMultipolarity(32S_tree->GetLevel(32));
```

```
32S_tree->SetQvalue(8853.64, KEV);
```

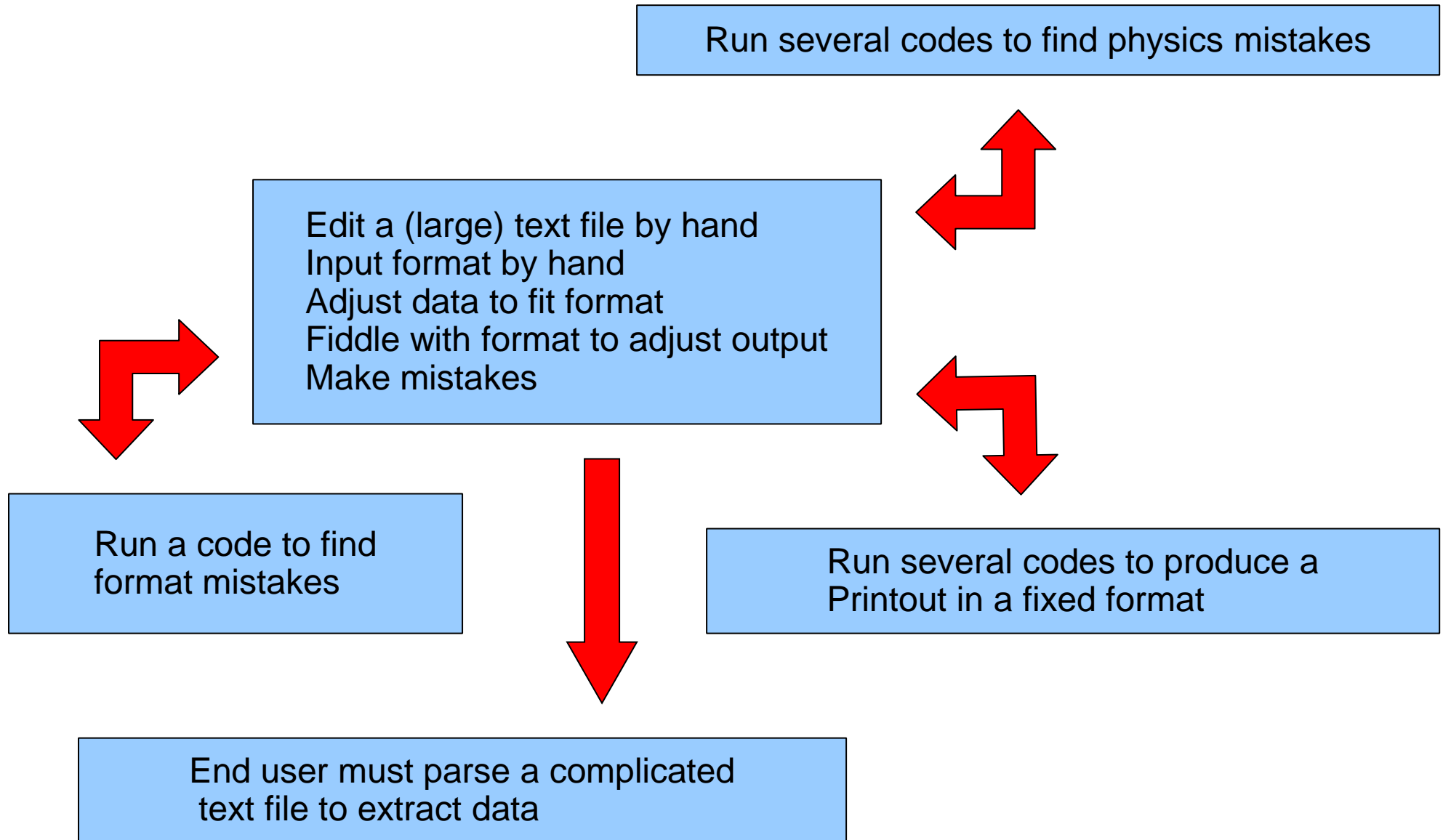
Or

```
32S_tree->SetQvalue(8853.64 + 2, KEV);
```

Interpreter Advantages

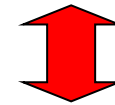
- The interpreter can be as complex or as simple as desired
- Examples of possible useful properties:
- Could perform physics calculations instantly, warning the user that his input doesn't make sense
- Could graphically display new input to the tree data (tables, energy level diagrams) in real time
- Built in compiler for easy data manipulation

Schematic of Current Practice

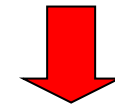


Schematic of Suggested Future Practice

Add Nuclear Data in a command line or GUI to a Tree



Evaluate effect of data entry on Physics, graphical output in real time



Produce an “Evaluated Data Tree”
And suggested graphical output

Centralized Program

Reads, Writes and Interprets Data Tree
Warns evaluator in real time of incorrect physics input
Graphically displays effect of input
In real time
Built in compiler for easy leaf data Manipulation and end user output