SCYTHE: AN EXTRACTOR LIBRARY YOU MIGHT LIKE

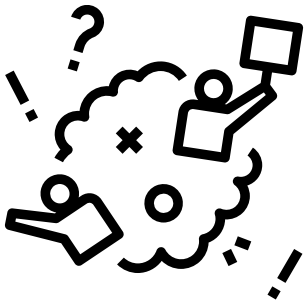
LOGAN WARD
Argonne National Laboratory
Data Science and Learning Division

WHY DID WE MAKE SCYTHE?

Well, we first called it “MaterialsIO”

Both Citrine Informatics and the Materials Data Facility had *messy data files.*

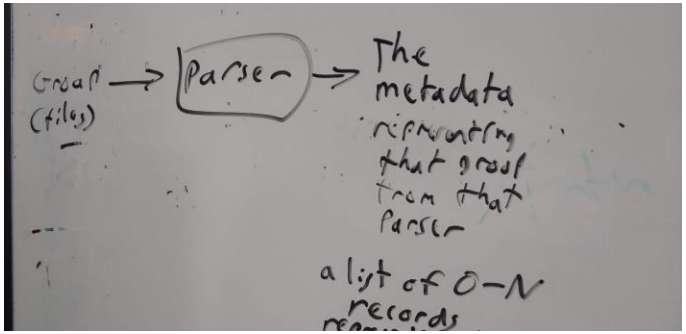
Each time the mess expanded
we created another tool
following whatever interface whomever felt like,



And did it *over and over again*

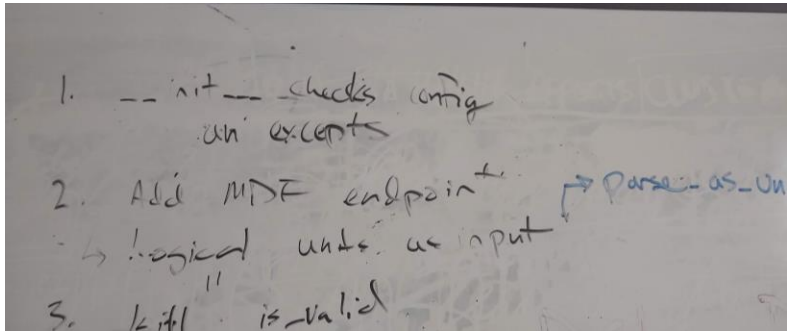
MERGING TOOK TALKING

We didn’t even have the same words for concepts, or goals



What is the operation we perform?

- Data to summary?
- Data to a new format (e.g., PIF)?

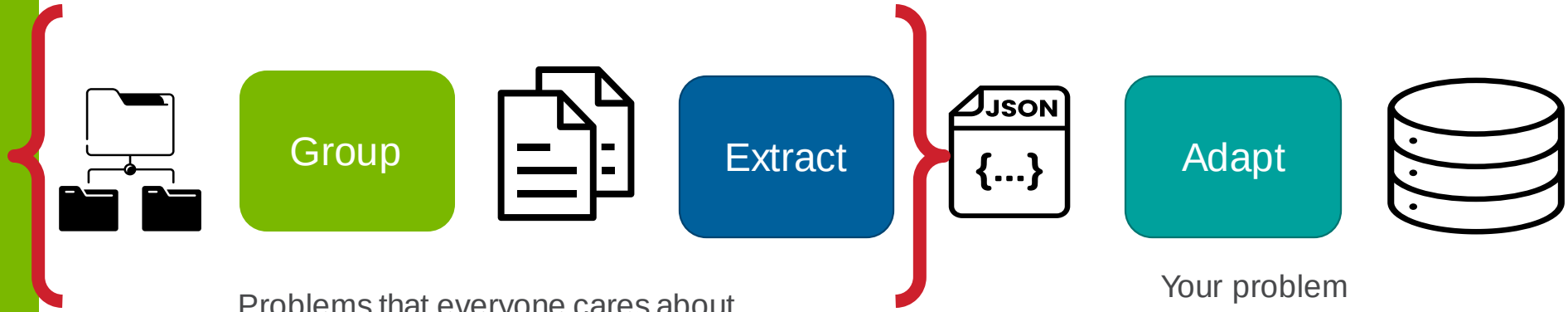


What is the interface?

- Where do file groups come from?
- Where do they go?
- What does the output look like?

SOME TIME LATER: A SYSTEM CHART

Between your files and your database



WHAT IS “GROUPING”

Scientists like many files, computers hate them

```
my_run\
   in.test
   out.test
   better.in
   better.out.3Oct94
   better.out.3Feb23
```

Group

```
[my_run\in.test
 my_run\out.test]
[my_run\better.in,
 my_run\better.out.3Oct94]
[my_run\better.in,
 my_run\better.out.3Feb23]
```

Grouping algorithms…

- use how files are arranged on filesystems
- to identify files that should be treated together
- for a <u>specific type of data extractor</u>

WHAT IS “EXTRACTING”

Turning trash into treasure

```
[my_run\better.in,
 my_run\better.out.3Feb23]
```

Extract

```
{
  ‘material’: ‘CuO’,
  ‘code_version’: asdf44,
  ‘temperature’: 1544,
   ...
```

Extracting algorithms…

- pull ugly scientific formats
- into a documentable format (JSON)
- that contains all data *anyone* cares about [note present tense!]

WHAT IS “ADAPTING”

Turning treasure back into *your preferred currancy*

```
{
   ‘material’: ‘CuO’,
   ‘code_version’: asdf44,
   ‘temperature’: 1544,
    ...
```

Adapt

```
<my_document>
   <material>CuO@1544</material>
   ...
</my_document>
```

Adapting functions…

- Take in the general-purpose, documented format
- and transform it into whatever you’d like
- so you can do whatever with it

THAT'S THE GENERAL PRINCIPLES

Now for the implementation

Our goals:

- Have the interface in Python
- Have a minimal interface that need be fulfilled
- Enable "file-system to database" in one line of code
- Make it work for >1 person

DOCUMENTATION TOUR

- Things to hit
  - Manifesto
  - Base classes
  - Utility operations
    - How Stevedore Makes it work
  - Adapter example for MDF

WHERE ARE WE AT?

It's still just a proof of concept

Who all is using it? No major users *yet*

- Not MDF (but its on the to-do list)
- Not Citrine (I'm not sure now that Citrination "open" is gone)

What are we actively doing?

- Revising the MDF extraction toolchain
- Explaining it to fine people (i.e., you) to see if it fits anywhere

What would I do next? [Given time]

- Finish up the documentation
- Strip out features to reduce dependencies (Py3.8+, stevedore is the goal)

WHAT'S NEXT

My question: Does this fit the model you need?

- If not, is there a small change we can try out?
- If so, how can we test to be sure?

My goals:

1. *Make it easier to use this groups' work in the MDF*
2. *Keep the interfaces for extractors as simple as possible, Pythonic*
3. *Save you all time*
4. *Puff my GitHub metrics 📈, get money 💰, retire early 🏖️*