

Open LCA Software Advances

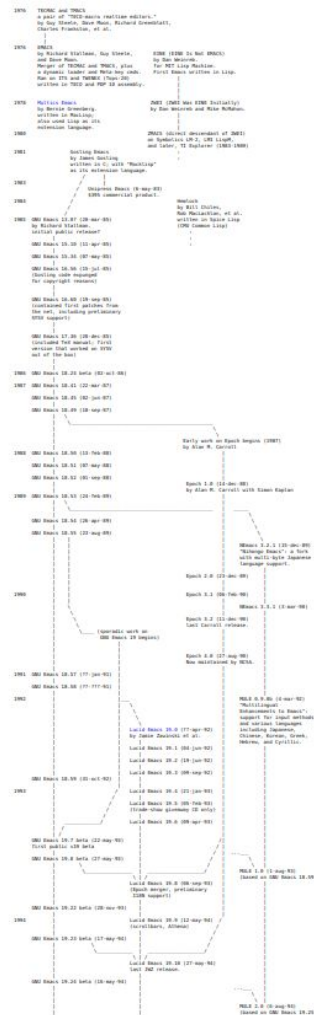
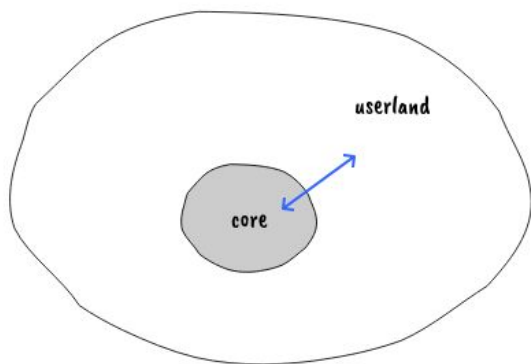
Michael Srocka, GreenDelta
Brightcon 2022

(openLCA software advances)

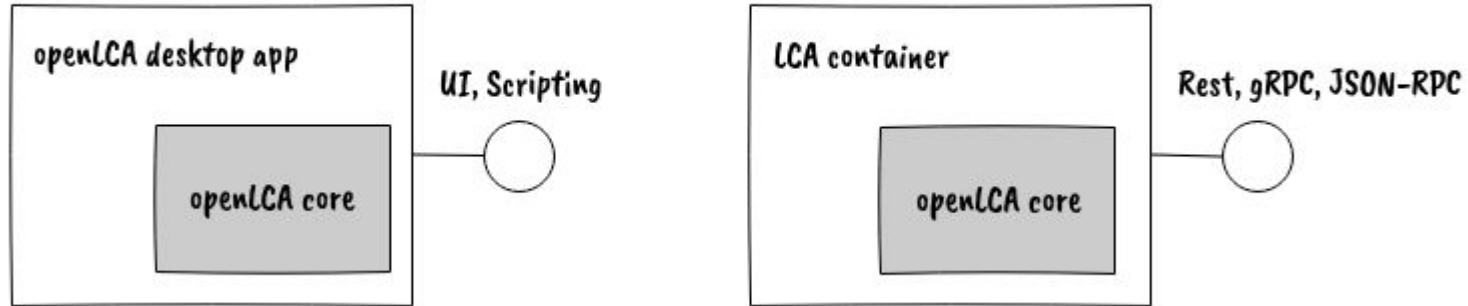
see [openLCA.org](https://openlca.org)

Vision for LCA tooling, maybe ...

- ... a bit like Emacs:
 - provide an open **core** that just works
 - with a **communication concept**, the "buffer"
 - build creative things on top of it, in **userland**
 - move best things back into the core
 - iterate



openLCA: desktop & web-services



Data model & communication protocol

Data model and communication protocol

- Java API: access the internals
 - Java, Kotlin, Scala, ...
 - Jython interpreter in openLCA
- external interface: openLCA schema

```
mass = db.getForName(FlowProperty, 'Mass')
brick = db.getForName(Flow, 'brick, at plant')
wall = db.insert(Flow.product('Wall element', mass))
process = Process.of('Wall something', wall)
process.input(brick, 1000) # kg
db.insert(process)

App.open(process)
```

openLCA schema

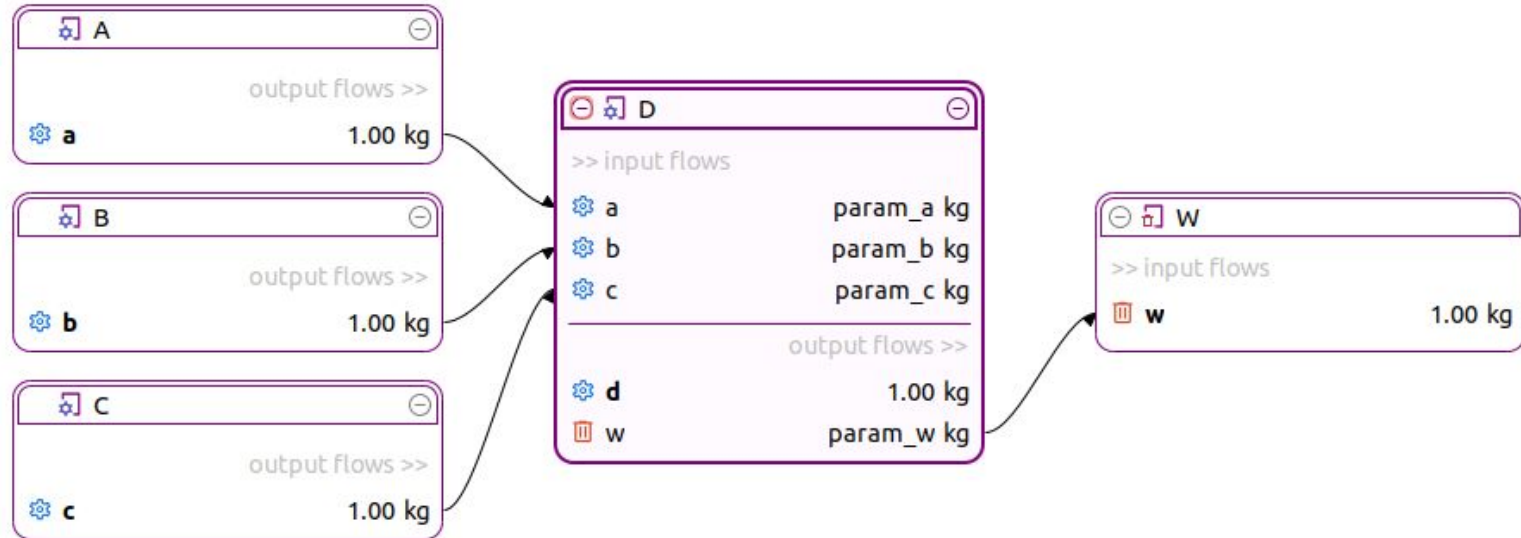
- Version 2: github.com/GreenDelta/olca-schema
- a meta-format spec for:
 - Json, Protocol Buffers, Python classes ...
- used for:
 - file based data exchange
 - JSON-RPC, gRPC, Rest APIs
 - Git based data exchange
 - ...

```
cd olca-schema/osch && \  
| go build && \  
| ./osch all
```

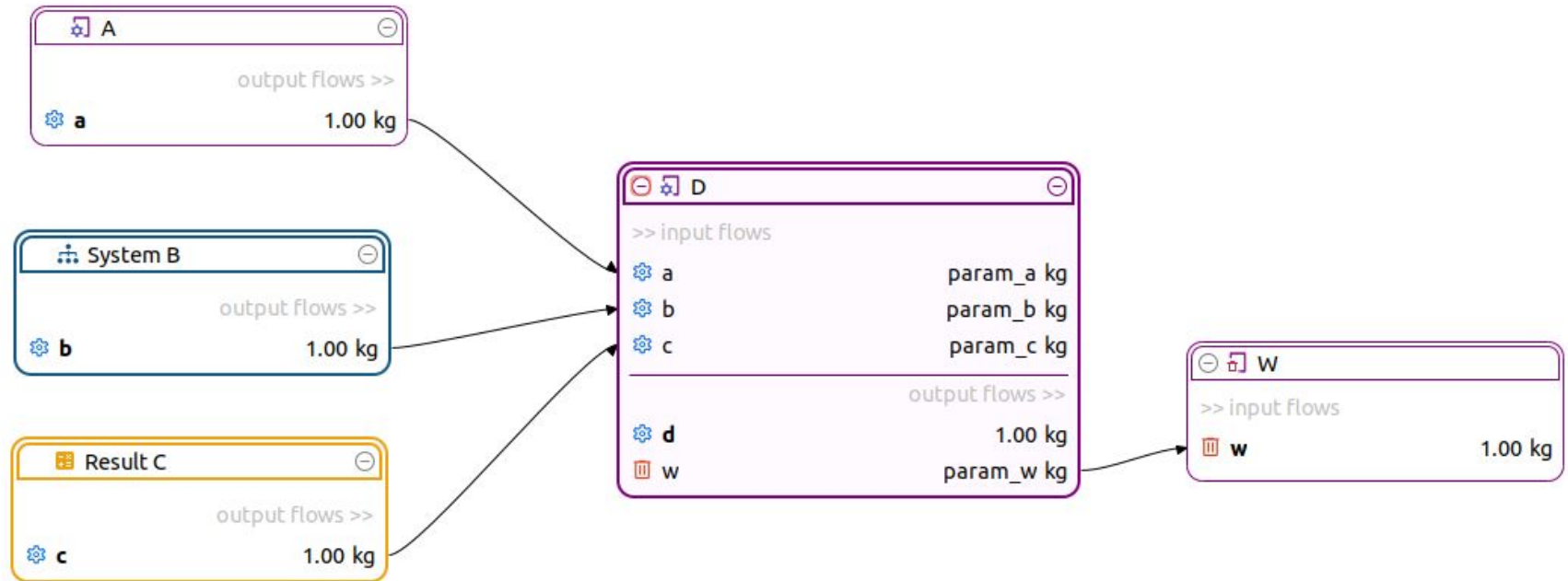
or

```
pip install olca-schema
```

Example: product systems



Product systems



Results

- stored LCI results, LCIA results only; or both
- e.g. imported from EPDs, or used to build EPDs
- ILCD+EPD & OpenEPD support

The screenshot displays the software's EPD management interface. On the left, the 'Find EPDs' section shows a search for 'CEM I 52.5 N'. Below this is a table of search results. A context menu is open over the 'CEM I 52.5 N' entry, with options 'Import EPD results' and 'Save as file'. To the right, two process flow diagrams are shown. The top diagram, titled 'CEM I 52.5 N - A1A2A3 - TRACI...', shows an output flow of 'CEM I 52.5 N' with a value of '1.00E3 kg'. The bottom diagram, titled 'my mortar - CH', shows input flows including 'CEM I 52.5 N' (0.34 kg), 'chemicals organic, at pl...' (8.00E-4 kg), and 'conveyor belt, at plant -...' (3.33E-8 m). Arrows indicate the flow of data from the search results into these process diagrams.

EPD	Manufacturer	Category	Declared unit
CEM I 52.5 N (vs) (Velox)	Cementa AB	Manufacturing Inputs >> Cementitiou...	1 t
Portland cement - CEM I 52.5 N	Nesher Israel Cement Enterprise	Manufacturing Inputs >> Cementitiou...	1E+03 kg
CEM I 52.5 N	Titan Cement Group	Manufacturing Inputs >> Cementitiou...	1000 kg
ADOCIM CEM I 52.5 N	Adoçim	Manufacturing Inputs >> Cementitiou...	1 t
FutureCEM® cemen	Aalborg Portland A/S	Manufacturing Inputs >> Cementitiou...	1000 kg

Parameter redefinitions

- redefinitions applied per calculations ...
 - scopes: global, processes, LCIA categories

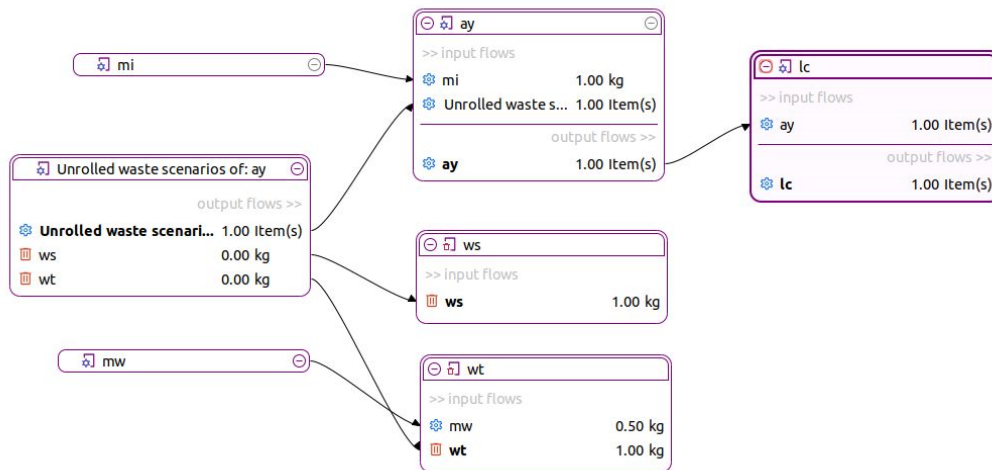
Parameters		
Context	Parameter	Amount
global	param_a	0.1
global	param_b	0.2
D	param_c	0.3
D	param_w	0.4

Contribution	Process	Required amount	Result
▼ 100.00%	D	1.00000 kg	2.00000 kg
20.00%	W	0.40000 kg	0.40000 kg
15.00%	Result C	0.30000 kg	0.30000 kg
10.00%	System B	0.20000 kg	0.20000 kg
05.00%	A	0.10000 kg	0.10000 kg

Parameter redefinitions: stateless services

- each parameter can be redefined in a calculation request
- application in memory only
- can be stored in product systems
- sub-systems:
 - redef. of outer system overwrites redef. of sub-system
 - but not when redef. in sub-system is protected

Protected parameters redefs.



SimaPro CSV Import

Selected SimaPro CSV files:

/home/ms/Downloads/WasteCycle.csv

Flow mapping: SimaPro_Import.csv From file

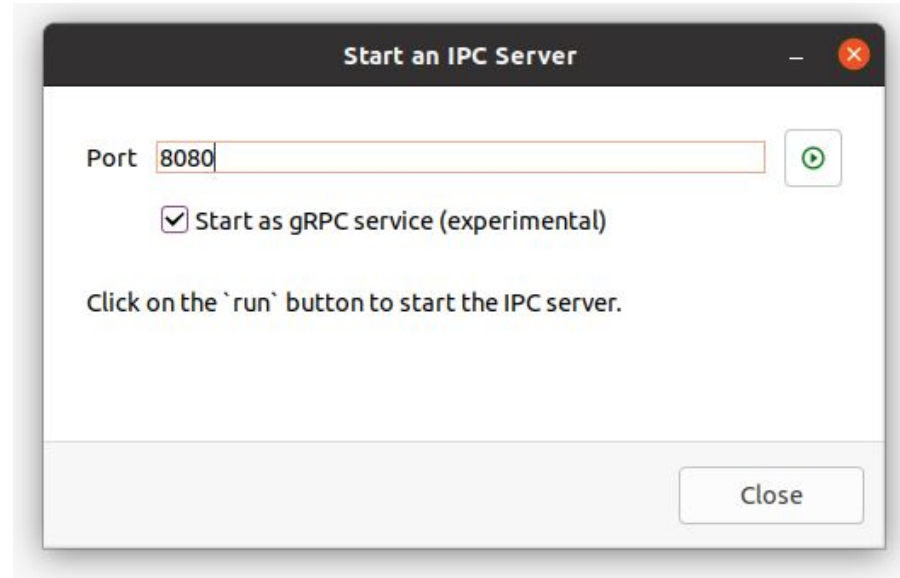
Generate:

- ☒ Product systems for life cycles
- ☒ Parameters for waste scenarios
- ☒ Characterization factors for sub-compartments

< Back Next > Cancel Finish

openLCA schema - next

- spec for calculation setups and results
 - simple results (LCI, LCIA, LCC, SLCIA,)
 - contribution results
 - upstream trees
 - graphs for Sankey diagrams
 - ...
- extensions
- for service integrations

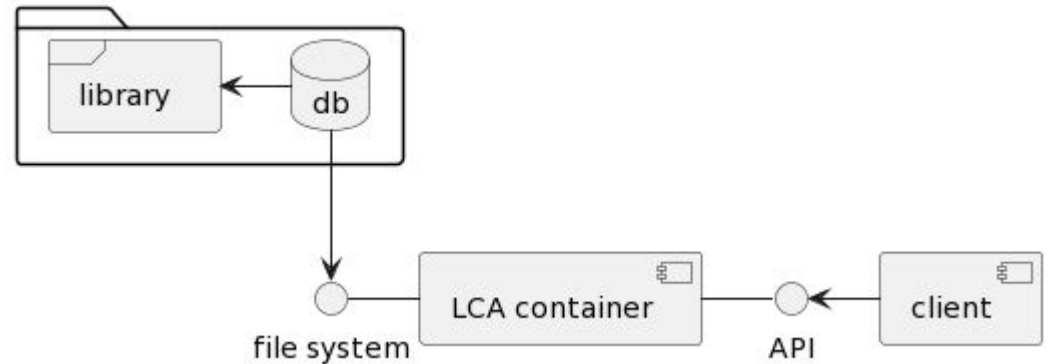


Service integrations

LCA containers

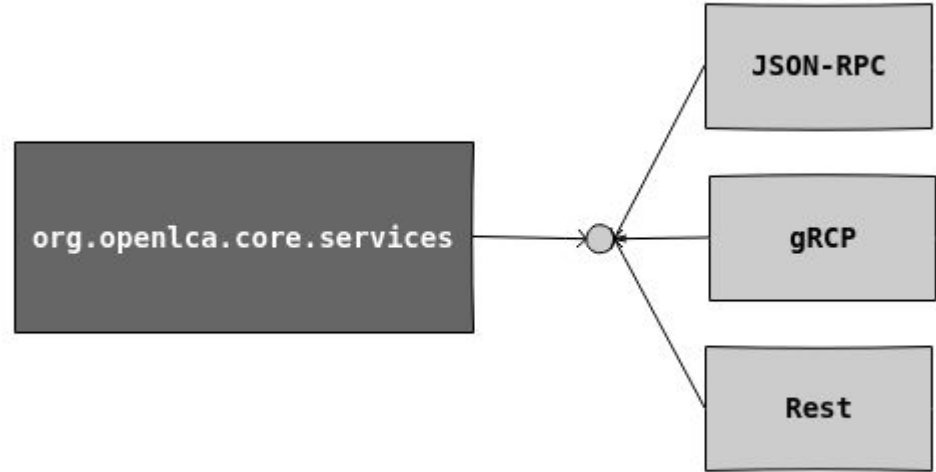
- container: server + service backend
- mount file system: database + possible libraries
- expose API

- works with every openLCA database and multiple service frontends
- model pipelines

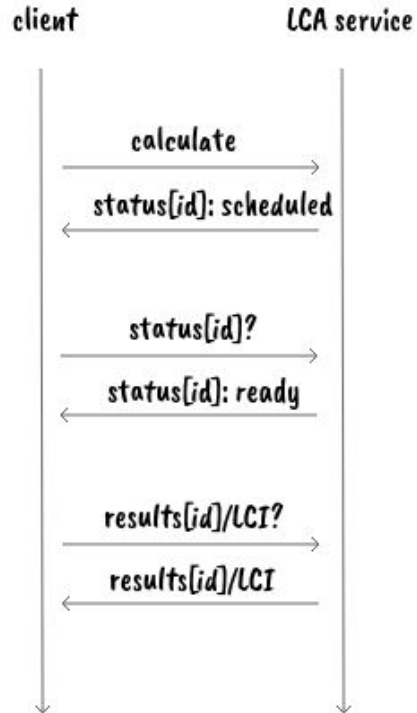


Service backend

- in development: a backend for multiple service frontends
 - calculation queue, result polling, JSON de/serialization, ...



Calculation queue



Client APIs

```
system = client.get_descriptor(ProductSystem, uid)
setup = ipc.CalculationSetup(
    product_system=system,
    amount=500,
    unit=units.unit_ref('g'),
    parameters=[
        # ...
    ])
result = client.calculate(setup)
# ...
client.dispose(result)
```











Libraries

Libraries

- a vertical core extension
 - data packages with pre-processed matrices
- improvements
 - calculation speed & memory usage
 - graphical editing
 - data sharing
 - ...

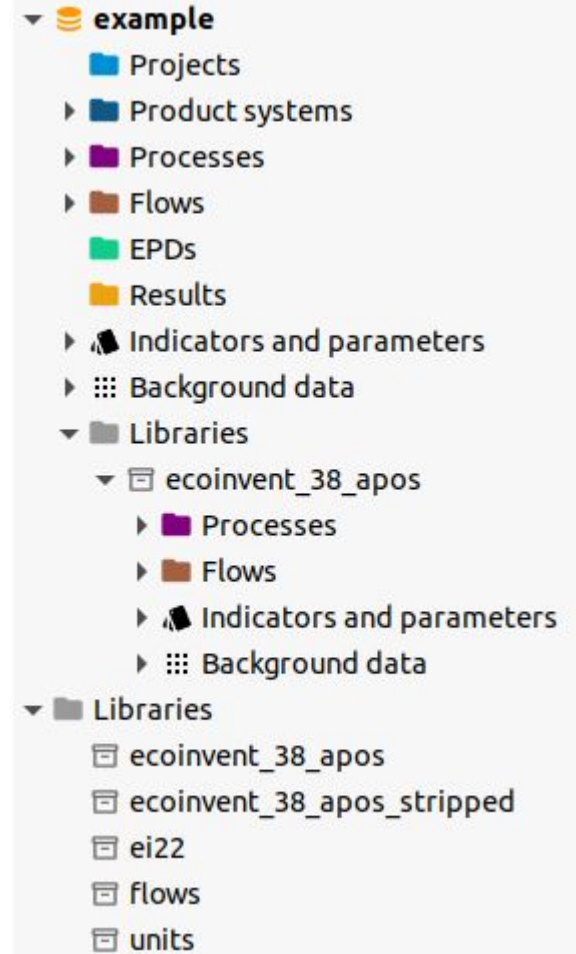
Libraries

- read-only package format
- based on open standards:
 - matrices: NPY, NPZ
 - indices: CSV, Protocol Buffers
 - meta-data: openLCA schema

Home openLCA-data-1.4 libraries ecoinvent_38_apos		
Name	Size	
 A.npz	4,6 MB	
 B.npz	3,7 MB	
 C.npz	752,6 kB	
 index_A.bin	12,4 MB	
 index_B.bin	294,9 kB	
 index_C.bin	137,2 kB	
 INV.npy	3,1 GB	
 library.json	51 bytes	
 M.npy	342,0 MB	
 meta.zip	135,0 MB	

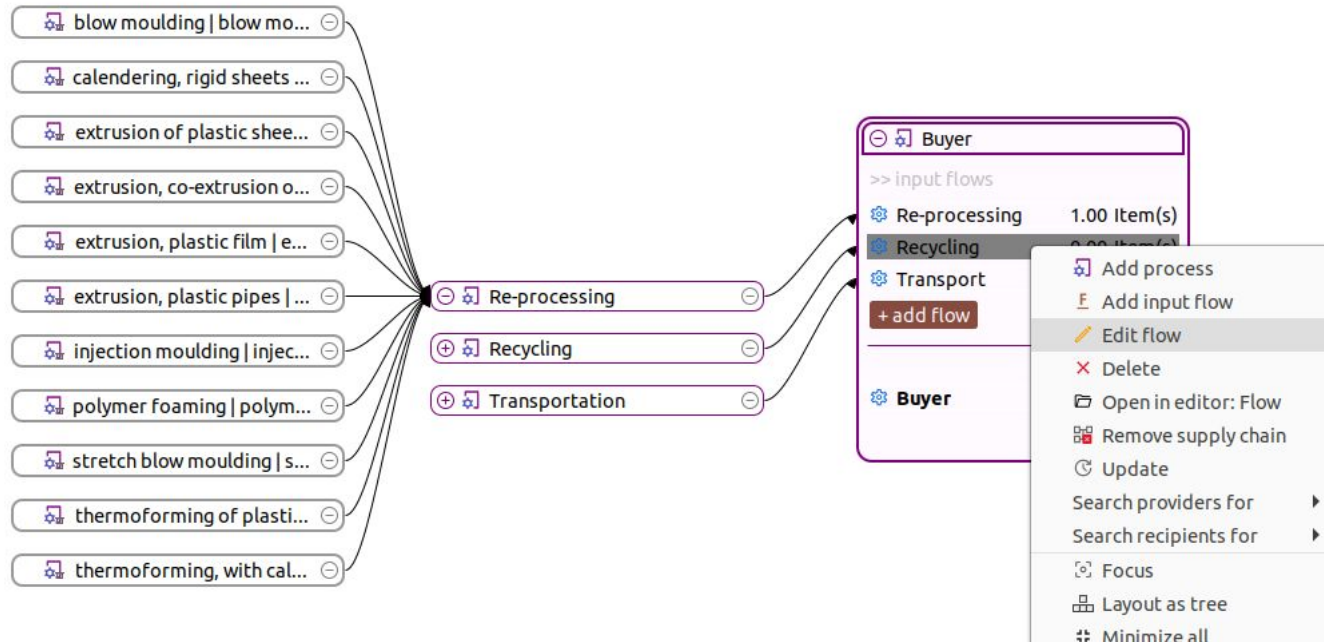
openLCA integration

- works with sparse and dense systems
- library objects usable like any other object
- support recursive dependencies for reference data (e.g. units, flow lists)



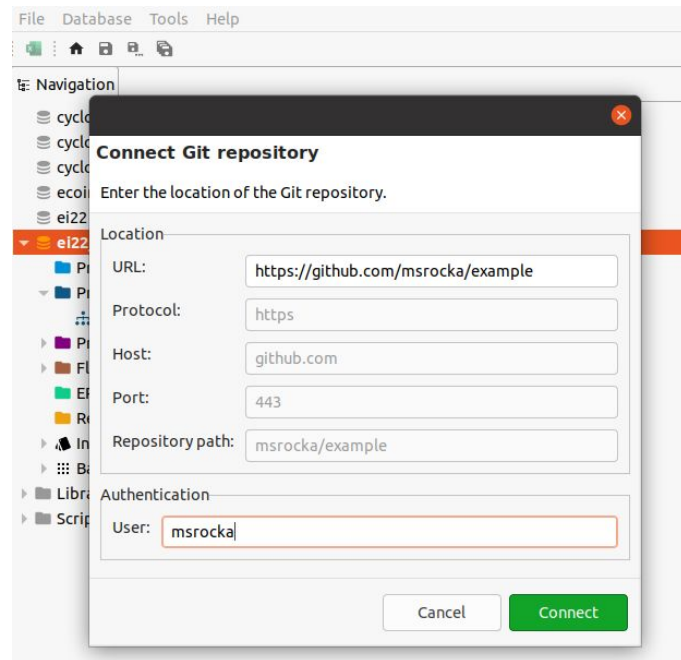
Graphical editing

- draw your LCA model; libraries hide details
- but you get all details in the results



Data sharing

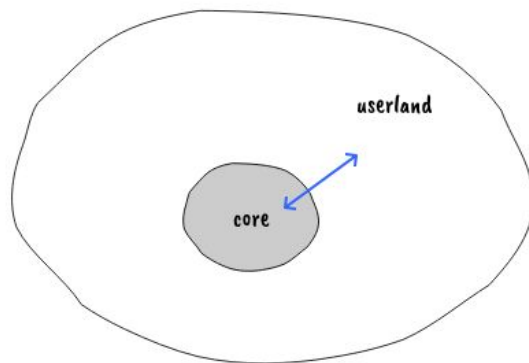
- supported in: openLCA Collaboration Server 2.0
- builds on top of openLCA schema & Git
- openLCA db <-> sync with local repo <-> sync with remote
- with libraries: only foreground system is shared



Conclusions

Conclusions

- "dogfooding": building new things in userland first
- use open standards: Json, Npy, ... SQLite?
- make the data model extendible
(like key-value-pairs in Brightway)



Thank you!

srocka@greendelta.com