Karlsruher Institut für Technologie

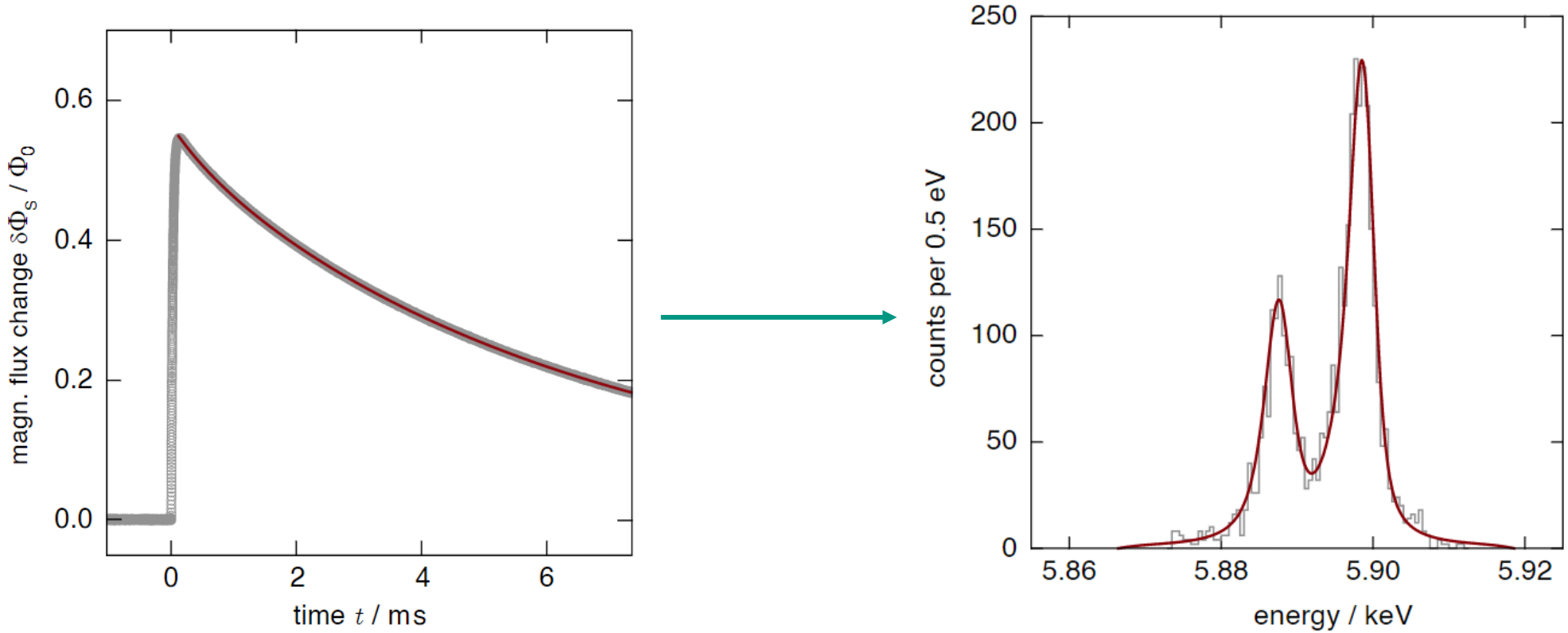
Summary of the first DELight hackathon

Francesco Toschi

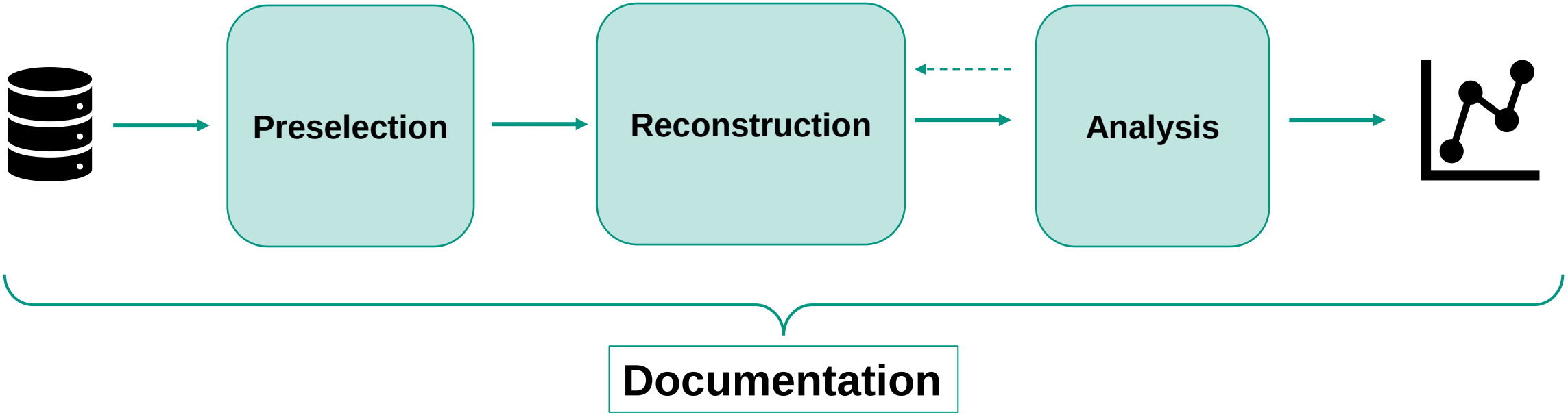
DELight collaboration meeting
Heidelberg, 17.05.2023

What was the aim?

- Developing a tool for loading, processing and storing the data from MMCs



What was the aim?



The team

- Greta Heine
- Benedikt Maier
- Francesco Toschi

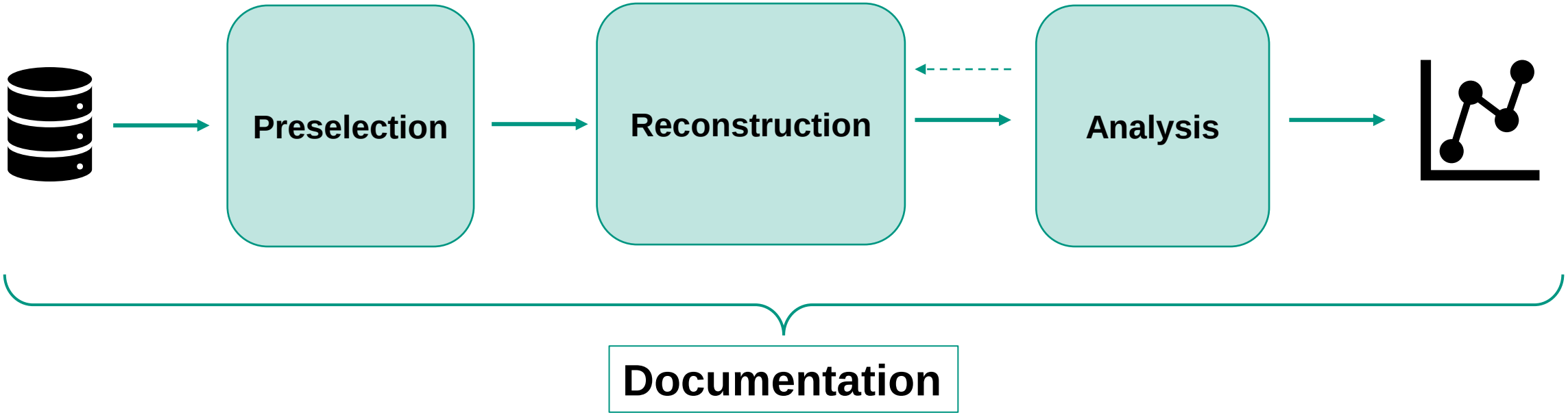
DELight

- Alexander Zaytsev
- Matthew Wilson

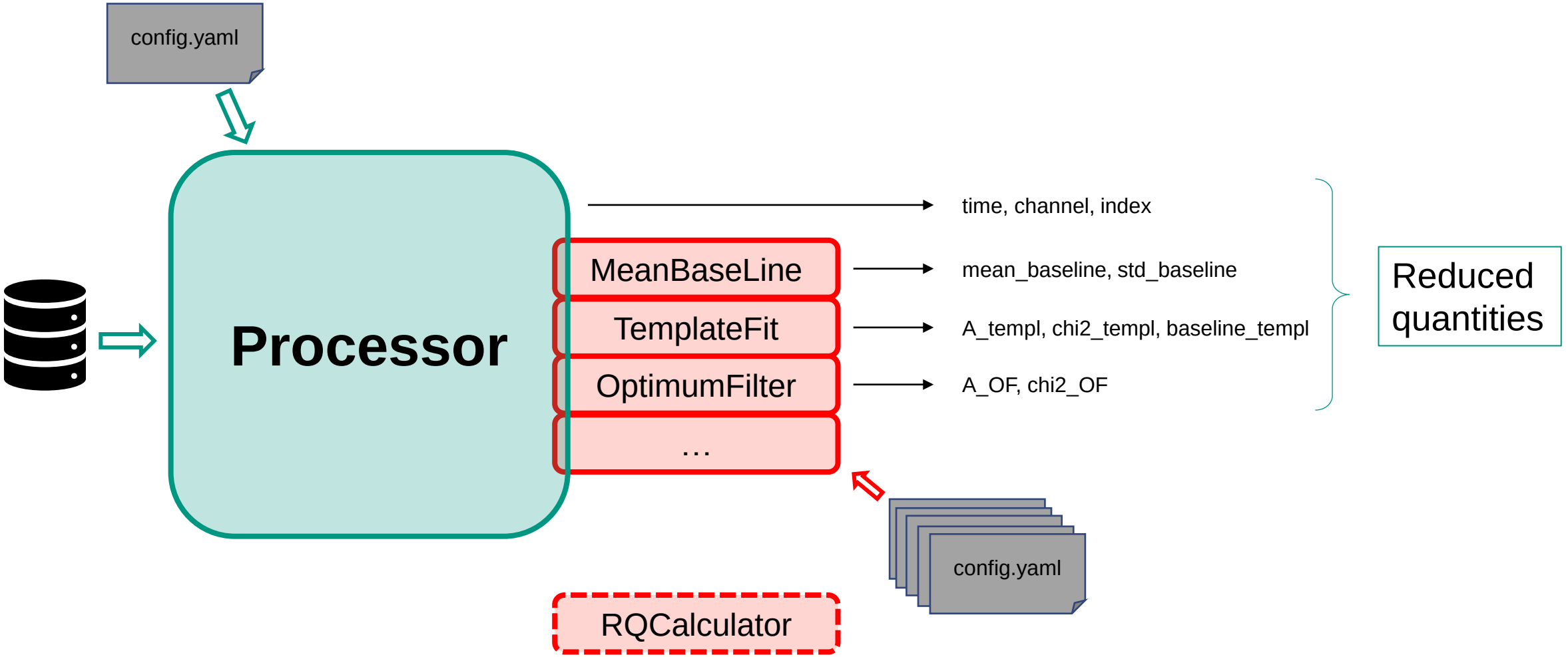
SUPER CDMS

And great help from Belina von Krosigk and Sebastian Kempf!

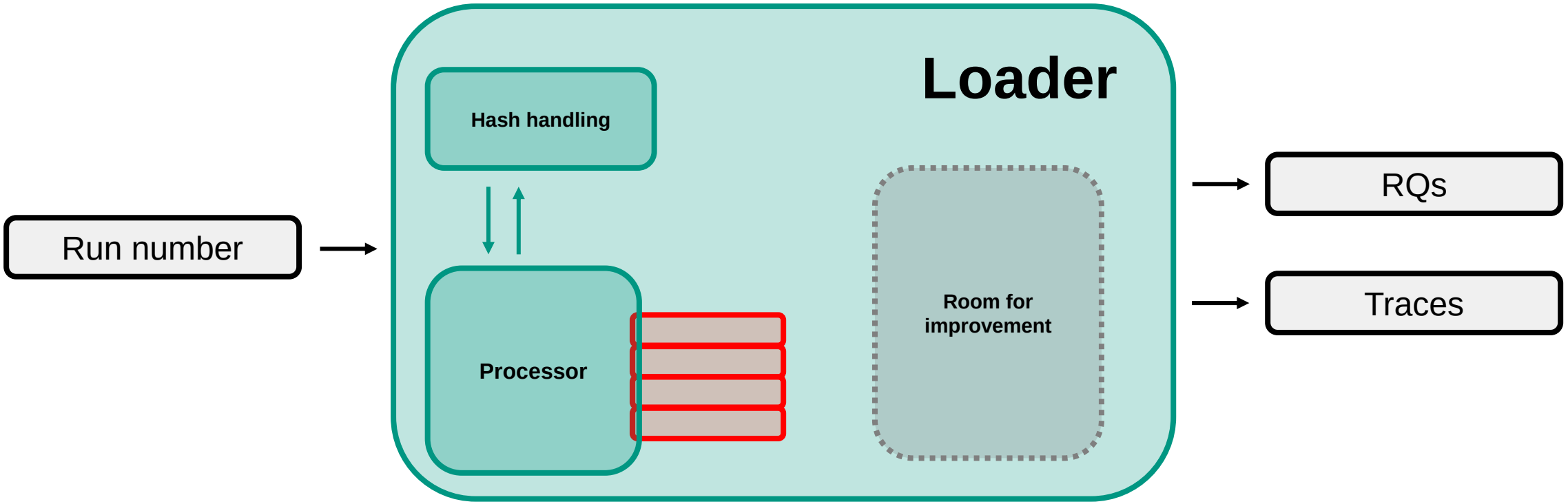
Initial idea



Processor


config.yaml
Processor
MeanBaseLine
TemplateFit
OptimumFilter
…
time, channel, index
mean_baseline, std_baseline
A_templ, chi2_templ, baseline_templ
A_OF, chi2_OF
Reduced quantities
config.yaml
RQCalculator

Loader


Loader
Hash handling
Processor
Room for improvement
Run number
RQs
Traces

How to use it

```
python3.8 Processor.py spectrum_for_1p3eV.pkl
```

Before

Can be also *.raw*

How to use it

Karlsruher Institut für Technologie

[1]:
```
from pprint import pprint
import RQ
from Processor import *
from Loader import *
from utils.plotter import *

run_number = 0
loader = Loader()

# if not processed
rqs, traces = loader.processor.process('spectrum_for_1p3eV.raw')
loader.processor.save(rqs, traces, run_number)

# load
rqs, config_rq = loader(run_number)

# plotter
plot = WaveformPlotter(*loader.input_plotter(run_number))
```

Now

rqs shape
(N_{events}, $N_{channels}$)

Output (rqs)

rqs shape
(N_{events}, $N_{channels}$)

	time	channel	trace_index	mean_baseline	std_baseline	A	rise_time	TF_ampl	TF_chi2	TF_baseline	baseline_slope
0	295986	1	0	128.96450	5.505519	10407.03550	0.000026	10396.672351	1.414104	2.982654	0.000008
1	401300	1	1	131.35750	5.545737	10422.64250	0.000026	10414.062287	1.112502	1.192284	0.000382
2	866004	1	2	132.40300	5.674733	10494.59700	0.000026	10485.467997	1.059241	1.313043	0.000066
3	1039604	1	3	128.61825	5.414196	10404.38175	0.000026	10396.163293	1.231764	0.335026	0.000172
4	1539099	1	4	128.21525	5.568341	11405.78475	0.000026	11393.542359	1.307578	-0.546275	0.000156
...	...	...	...	...	...	...	...	...	...	...	...
95	32416708	1	95	128.19275	5.361352	11401.80725	0.000026	11392.103459	1.283110	-1.162503	-0.000293
96	32452643	1	96	125.33550	5.663165	10418.66450	0.000026	10405.360324	1.183365	2.324553	0.000021
97	32526382	1	97	129.34100	5.589071	10430.65900	0.000026	10414.397523	1.078284	0.934023	-0.000550
98	32812247	1	98	130.05075	5.596890	10404.94925	0.000027	10393.783913	1.074091	1.127158	-0.000130
99	24777869	1	99	132.16700	5.657483	10411.83300	0.000026	10394.637057	1.519386	-0.646263	0.000696

100 rows × 15 columns

Fixed channel - shape (N_{events},)

Hash handling

```
ftoschi@LAPTOP-OB5EDF1U:~/DELight/Hackathon_May_2023/processor/output$ ls
000000_rqs_q7qsk26gif.npz  000000_traces_kz5i7bvz4f.npz
```

Unique deterministic hash based on config of each RQ and their versions

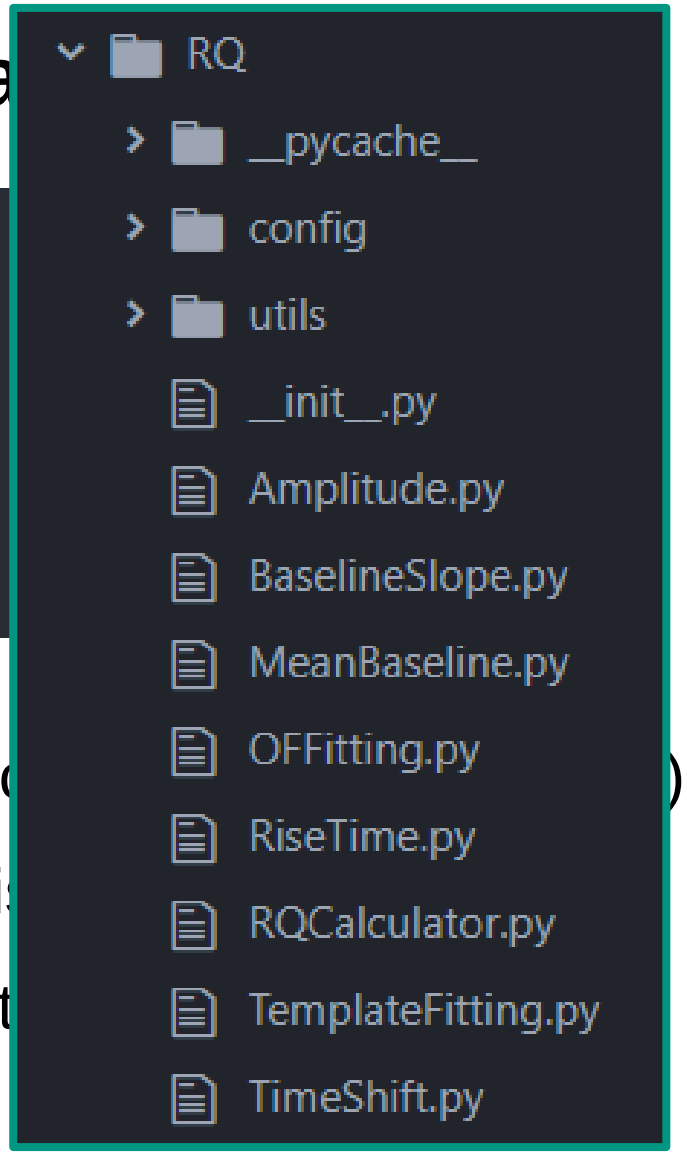
Reduced quantities (RQs) calculator

```
class RiseTime(RQCalculator):

    __version__ = '0.0.1'
    data_type = [('rise_time', np.float64)]
    dependencies = ['Amplitude', 'MeanBaseline']
```

- **version**, when changed the hash changes (change in the algorithm);
- **data_type**, it states what the RQCalculator is calculating;
- **dependencies**, it states what is needed for the calculation.

Reduced quantities (RQs) calcula

```python
class RiseTime(RQCalculator):

    __version__ = '0.0.1'
    data_type = [('rise_time', np.float64)]
    dependencies = ['Amplitude', 'MeanBaseline']
```

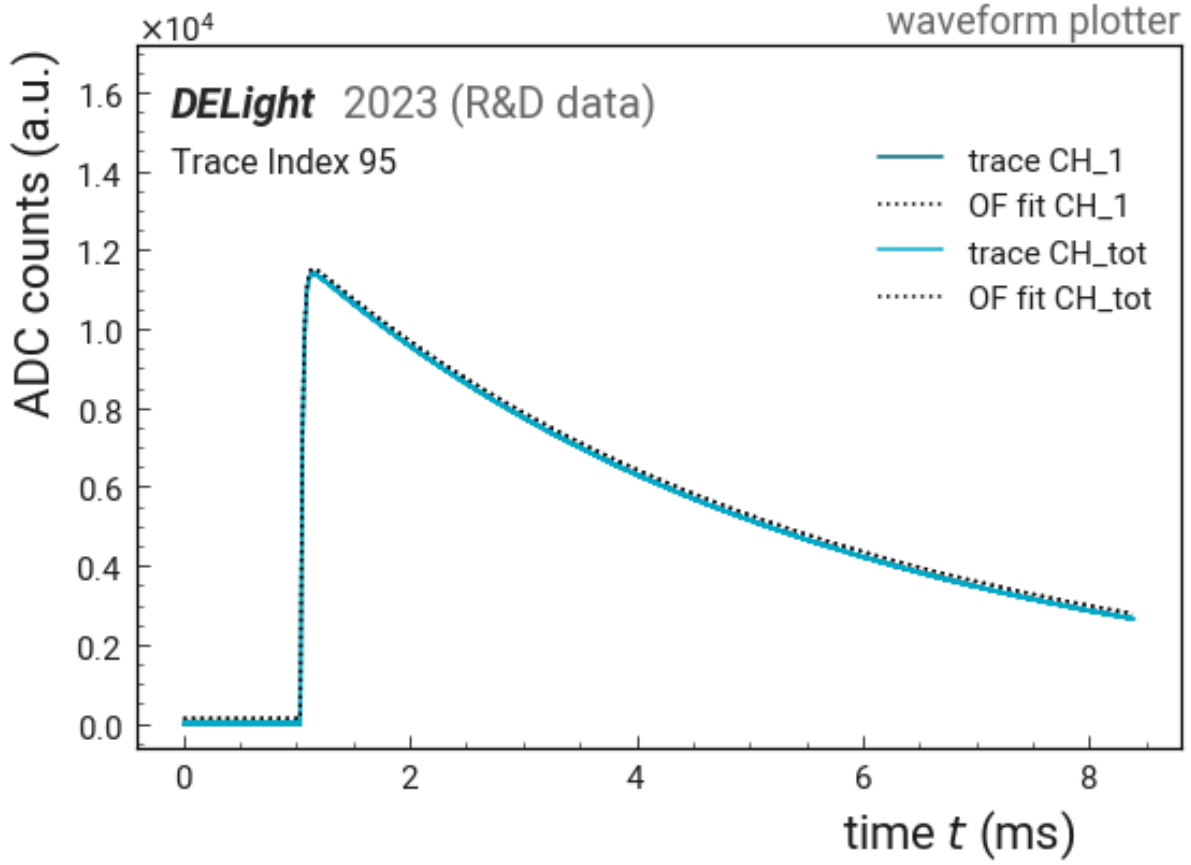


- **version**, when changed the hash changes (
- **data_type**, it states what the RQCalculator i
- **dependencies**, it states what is needed for t

);

Trace/waveform plotter

```
•[18]: # plotter
plot = WaveformPlotter(*loader.input_plotter(0))
plot.plot_waveforms([95], channels='all', template_fit_OF=True, xunits='time')
```



We have only one channel in this case, but the plotter can handle also more channels

Future of the software

- Most urgent: **find a name!**

- Integration of the waveform plotter in the loader;
- plotter loading directly raw data (no need to save *traces* output);
- *process_loop* implementaiton in *Processor* for raw data (large size);
- saving file in chunks and loading accordingly;
- interactive waveform visualization (brokeh?).