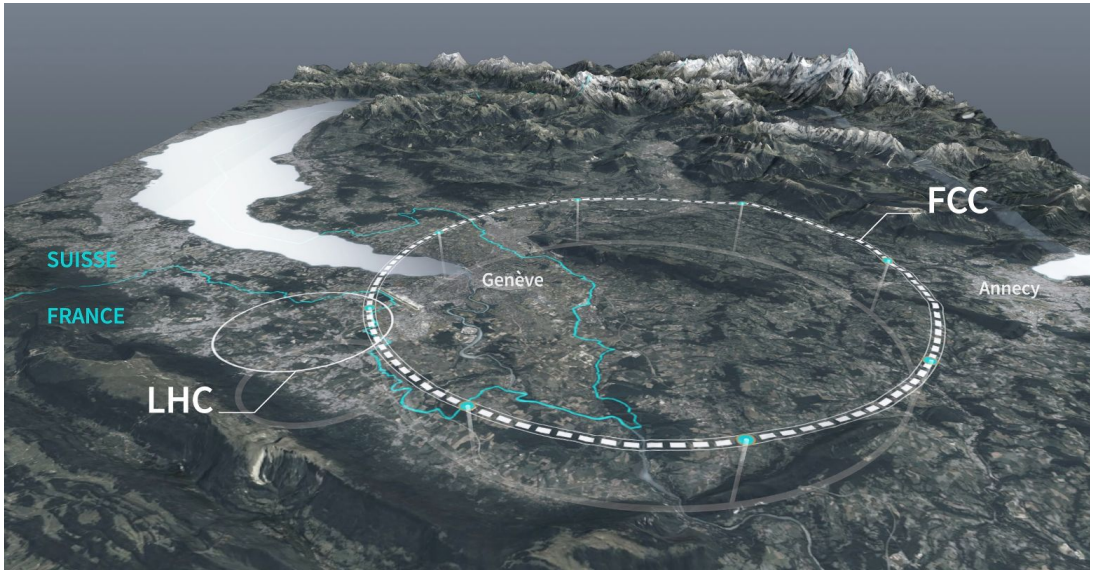
FCCAnalyses: Python-Driven Analysis at Scale for the Future Circular Collider

Birgit Stapf, Juraj Smieško, Pantelis Kontaxakis

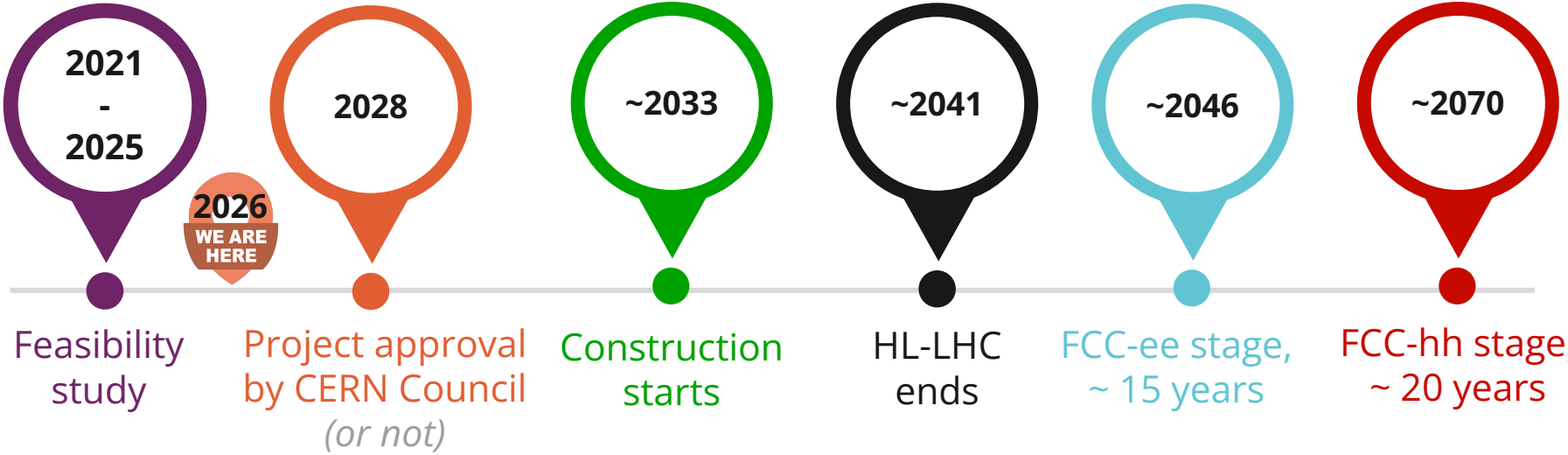
07.09.2026 | PyHEP.dev 2026 | Nikhef, Amsterdam

The FCC integrated program



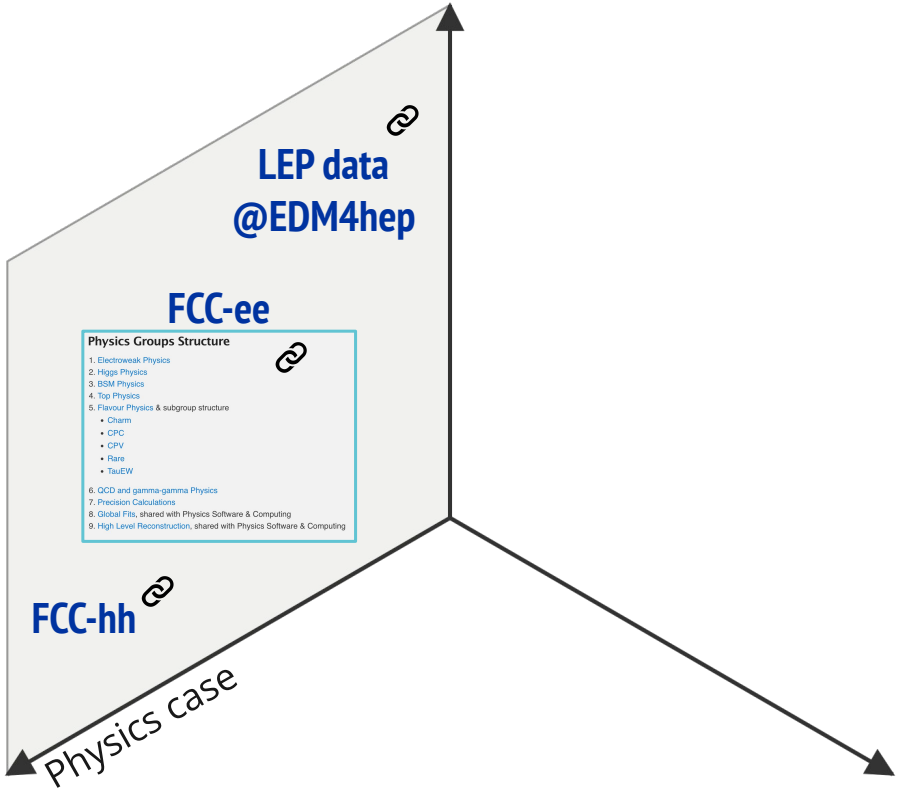
- Proposed Future Circular Collider: 91 km tunnel to be built at CERN and operated in two stages: Electron-positron (FCC-ee) and proton-proton collisions (FCC-hh)
  - A tried & tested strategy!

The FCC integrated program: Timeline

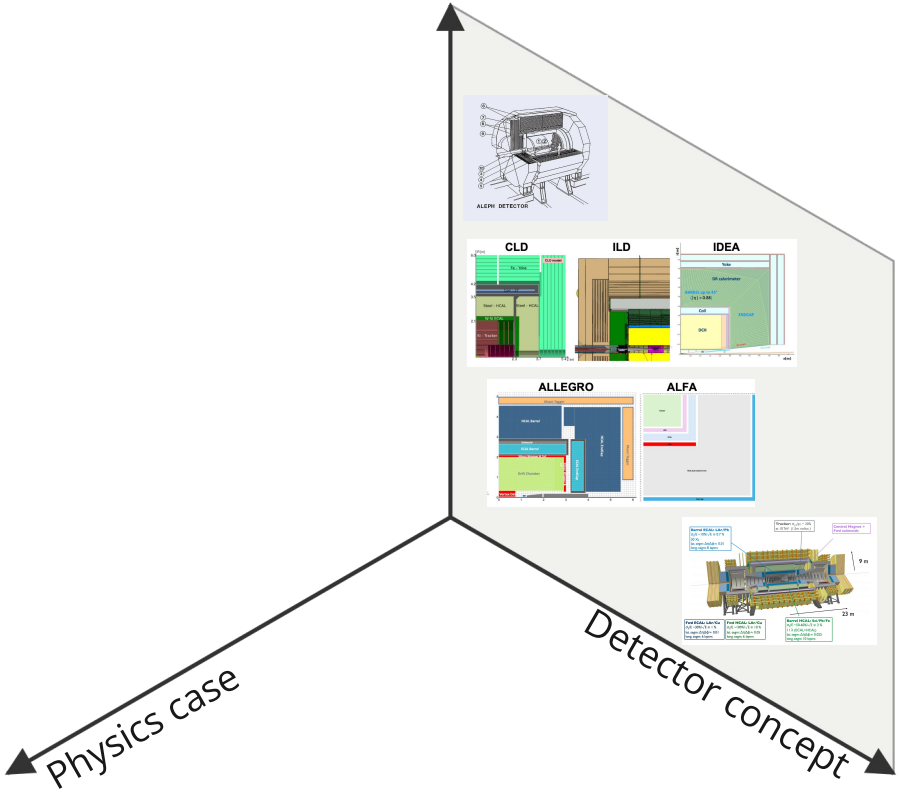


- Recent milestone: ESPP recommended FCC-ee as the preferred next flagship in May this year
- Reference design phase: full simulation, larger datasets, stricter reproducibility, and a growing user base, raising the bar for common software
- No detector, no data yet - every axis wide open

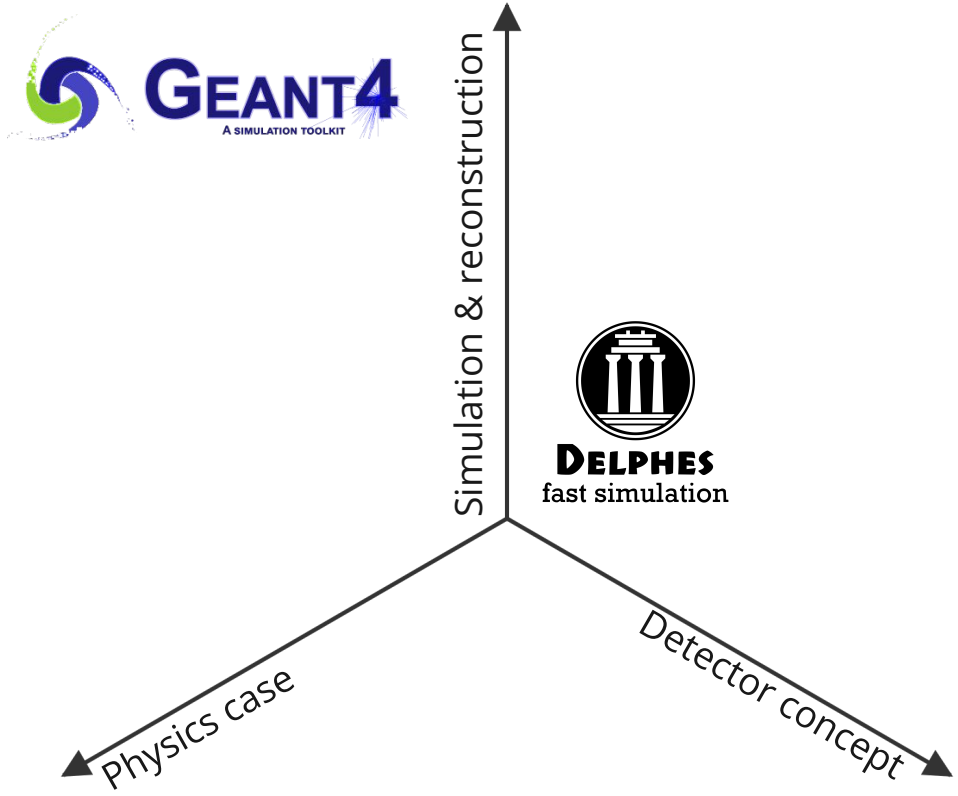
The FCC software phase space & kinematics



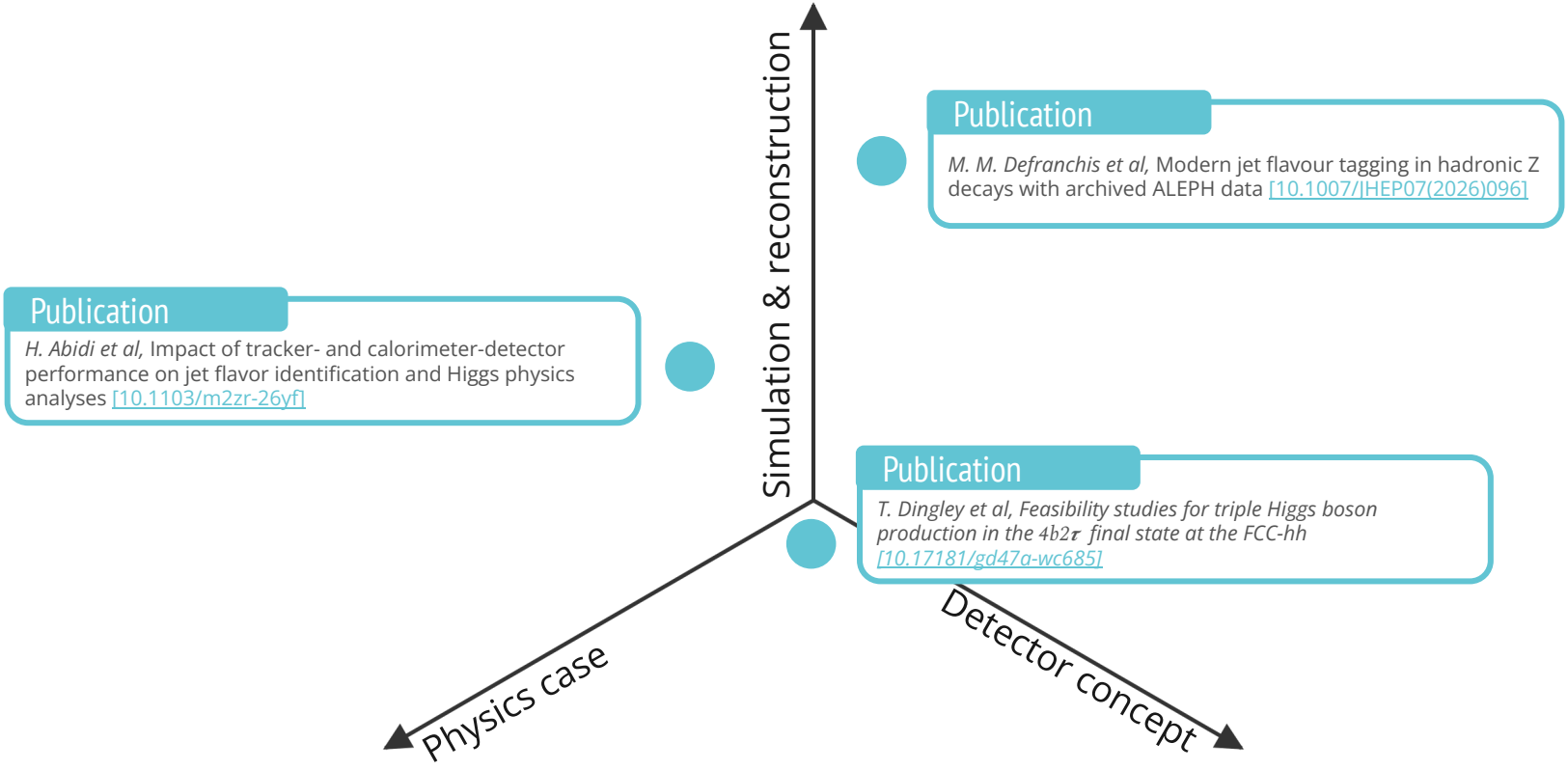
The FCC software phase space & kinematics



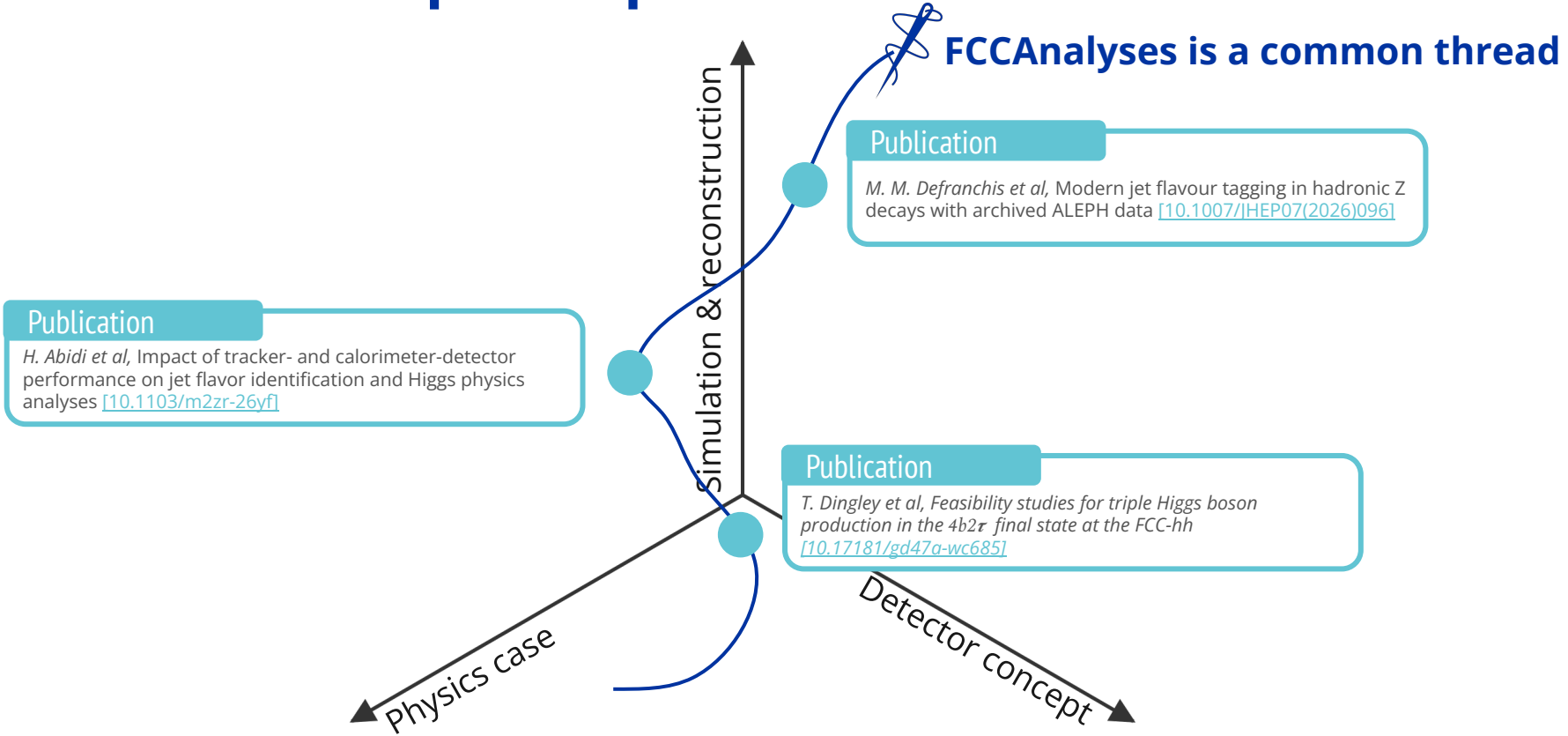
The FCC software phase space & kinematics



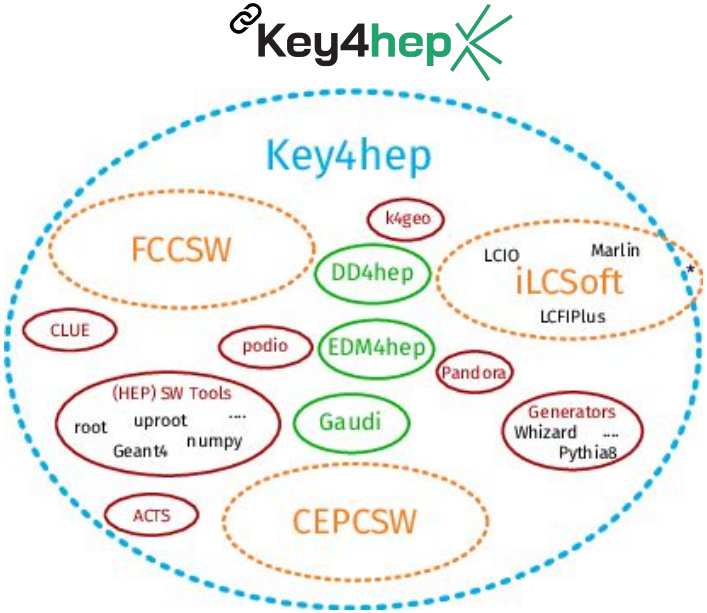
The FCC software phase space & kinematics



The FCC software phase space & kinematics



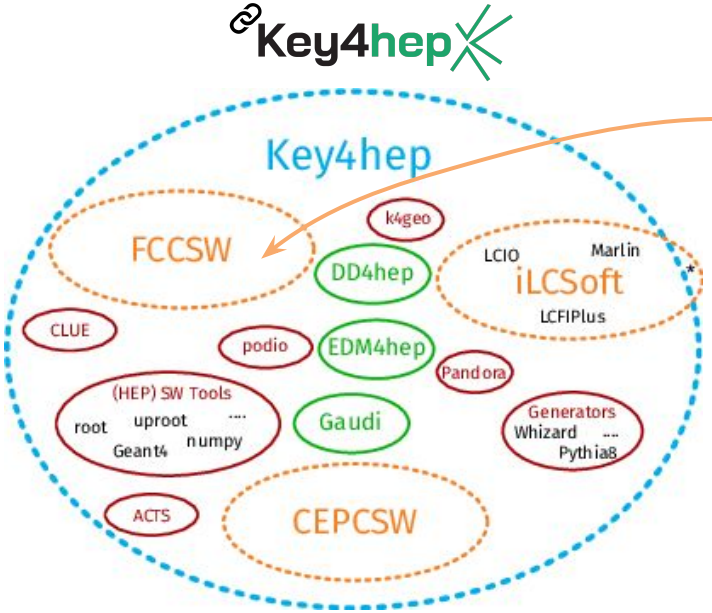
FCCAnalyses architecture at a glance



Sasikumar et al., PoS(ICHEP2024)1029

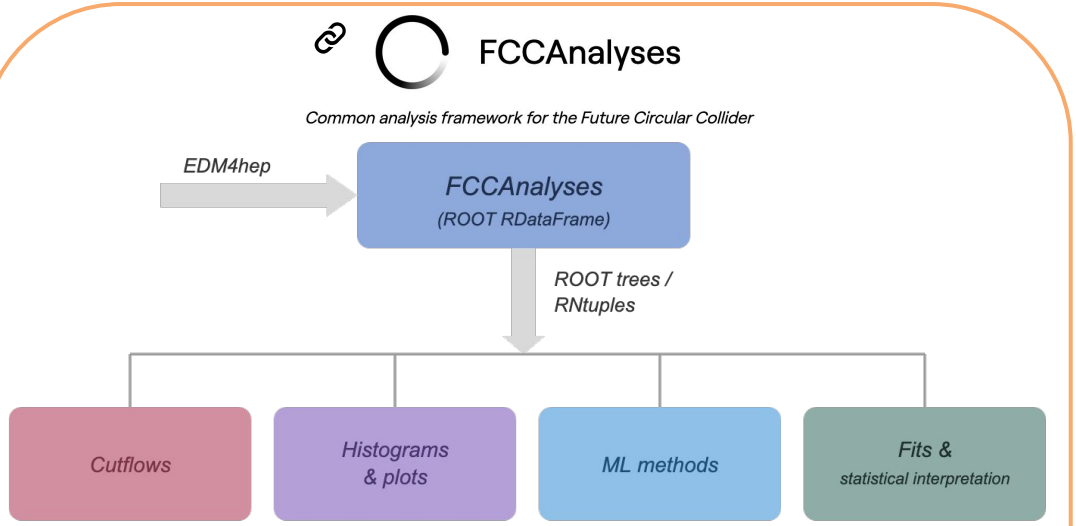
- Turnkey software stack for future collider studies

FCCAnalyses architecture at a glance



Sasikumar et al., PoS(ICHEP2024)1029

- Turnkey software stack for future collider studies



- Python builds the analysis graph over C++, with shipped functions or custom code written by user
- Auto-normalization: cross-sections & k-factors from the sample metadata service
- Precompiled version or build your own, need cvmfs

New features: A job to do the job

Code

PR #514 (merged), J. Smieško

Guiding principle

Support different workflows

- Backend refactor: replace scattered local-execution logic in `run_fccanalysis.py` with one clean class
- One `Job` = one unit of local work, one output ROOT file
- Adds option for more flexibility: per-sample `stride` & `n-events-max`, to e.g. run a test slice only

Framework

```
dframe_job = Job(job['input-file-list'], config['analysis-chain'],
                 config['use-data-source'])

dframe_job.restrict_events(job['n-events-max'], job['stride'])
dframe_job.run()
dframe_job.finalize()
n_events, elapsed_time = dframe_job.get_benchmark_info()
```

User interface

```
class Analysis():
    '''Higgs mass recoil analysis in Z(mumu)H.'''

    def __init__(self, cmdline_args):
        self.samples = {
            'p8_ee_ZH_ecm240': {
                'stride': 2,            # process every 2nd event
                'n-events-max': 300,  # cap at 50k events
            },
        }
        self.input_dir = '/eos/experiment/fcc/hh/tutorials/edm4hep_tutorial_data/'
        self.output_dir = 'outputs/FCCee/higgs/mH-recoil/mumu/stage1'

    def analyzers(self, dframe):
        # Define/Filter chain: build zed_leptonic, compute recoil mass, etc.
```

`fccanalysis run example.py`

Output

```
----> INFO: ================================ SUMMARY ================================
          Elapsed time (HH:MM:SS):                 00:00:03
          Number of events processed:              300
          Events processed per second:             87.66
          Number of events available locally:      1,000
          =========================================================================
```

New features: Dirac notation for job submission

Code

PR #538 (in review), B. Wach

Guiding principle

Support different infrastructures

- One command shape, two backends: `fccanalysis submit ht-condor / grid`
- Same input handling either way use *EOS/XRootD* samples, or an explicit *DIRAC* file list
- *DIRAC* runs in its own isolated process, so it can't collide with the *Key4hep* frontend environment

Framework

```python
def submit_analysis(parser: argparse.ArgumentParser) -> None:
    args = _parse_submit_arguments(parser)

    if args.where == 'ht-condor':
        if shutil.which('condor_q') is None:
            LOGGER.error("HTCondor tools can't be found!")
            sys.exit(3)
        LOGGER.info('Submitting analysis to HTCondor...')

    elif args.where == 'grid':
        try:
            submit_grid_submission(args)
        except GridSubmissionError as error:
            LOGGER.error('%s', error)
            sys.exit(3)
```

UI

```python
class Analysis():
    '''Higgs mass recoil analysis in Z(mumu)H.'''
```

unchanged!

`fccanalysis submit ht-condor example.py`

Output

```
----> INFO: Job submission command:
            condor_submit batch-submission-files/2026-09-06_17-49-33/p8_ee_ZH_ecm240/
job_desc_p8_ee_ZH_ecm240.cfg
----> INFO: Submitting job(s).
            1 job(s) submitted to cluster 12866985.

----> INFO: Submission successful.
```

New features: Branching out with RNTuple

Code

PR #531 (merged), D. Juknevičiūtė

Guiding principle

Keep pace with the underlying ecosystem

- One flag, either format, eg.: `--output-format rntuple` (*TTree* stays default)
- Can be set once in the analysis class too, with same pattern as `output_dir`
- Planned as baseline for the next round of studies, physics groups surveyed, no issues raised

Framework

```python
options = ROOT.RDF.RSnapshotOptions()
if self._output_format == 'rntuple':
    options.fOutputFormat = ROOT.RDF.ESnapshotOutputFormat.kRNTuple

final_dframe.Snapshot("events", self._output_file_path, self._output_variables, options)
```

User interface

```python
    def __init__(self, cmdline_args):
        self.samples = {
            'p8_ee_ZH_ecm240': {
                'stride': 2,            # process every 2nd event
                'n-events-max': 300,  # cap at 50k events
            },
        }
        self.input_dir = '/eos/experiment/fcc/hh/tutorials/edm4hep_tutorial_data/'
        self.output_dir = 'outputs/FCCee/higgs/mH-recoil/mumu/stage1'

        # Optional: output ROOT file format, default is 'ttree'
        self.output_format = 'rntuple'

    def analyzers(self, dframe):
        # Define/Filter chain: build zed_leptonic, compute recoil mass, etc.
```

Output

`fccanalysis run example.py`

```
KEY: ROOT::RNTuple     events;1
KEY: TParameter<int>   eventsProcessed;1
KEY: TParameter<float>         SumOfWeights;1
```

New features: To fit or not to fit

Code

PR #520 (in review), S. Niyogi

Guiding principle

As much of the common analysis chain as is sensible

- Define statistical model — channels, processes, systematics as a native Python class `examples/fcc_ee_zh_mumu_bb.py`
- `fccanalysis fit` generates *Combine* datacard, option -e to run
- `--backend` is pluggable by design; *Combine* is implemented, other backends under consideration

Framework

```python
def _get_metadata(self):
    imax = len(self.channels)
    jmax = max((sum(1 for p in c.get('processes', {}).values()
                    if p.get('type') == 'background')
                for c in self.channels.values()), default=0)
    kmax = len(self.systematics)
    return {'imax': imax, 'jmax': jmax, 'kmax': kmax}
```

User interface

```python
class Fit:
    def __init__(self):
        self.shapes = {
            "*": {
                "mumu_bjets_channel": "fcc_ee_zh_shapes.root $CHANNEL/$PROCESS $CHANNEL/$PROCESS_$SYSTEMATIC",
            }
        }

        self.channels = {
            "mumu_bjets_channel": {
                "observation": -1, # -1 instructs Combine to use asymptotic or toy datasets
                "processes": {
                    "ZH_signal":   {"type": "signal",     "rate": -1},
                    "ZZ_bkg":      {"type": "background", "rate": -1},
                }
```

Output

```
------------------------------------------------
bin                     mumu_bjets_channel    mumu_inter_channel
observation             -1                    -1
------------------------------------------------
bin                     mumu_bjets_channel    mumu_bjets_channel    mumu_bjets_channel
 mumu_inter_channel      mumu_inter_channel    mumu_inter_channel
process                 ZH_signal             ZZ_bkg                Zjets_bkg
 ZH_signal               ZZ_bkg                Zjets_bkg
process                 0                     1                     2
 0                       1                     2
rate                    -1                    -1                    -1
 -1                      -1                    -1
------------------------------------------------
lumi_FCC        lnN     1.005   1.005   1.005   1.005   1.005   1.005
```

New features: Weight, there's more

Code

PR #523 (draft), Z. Çabukoğlu

Guiding principle

Support for different workflows

- Adds option to use parameter reweighting, e.g. κ_3/κ_4 coupling scans from one production, or systematic variations
- Implements `GetAllWeights()` / `GetWeightByName("label")` C++ functors, called from Python
- Needs a companion *PODIO* change exposing file metadata

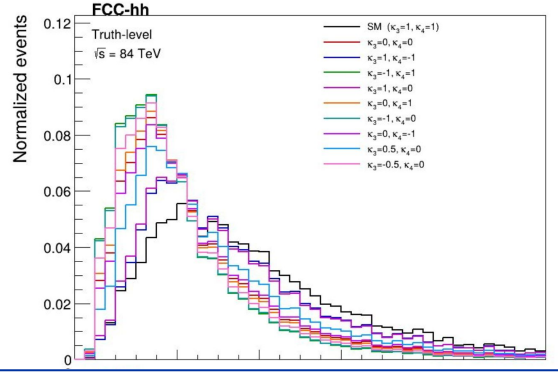
Framework

```
double operator()(const edm4hep::EventHeaderCollection& eventHeaders,…
) const {
    if (eventHeaders.empty()) {return -1.0;}
    int weightIndex{-1};

    // Loop through all weight labels supplied by the DataSource.
    for (std::size_t i = 0; i < weightNames.size(); ++i) {

        if (weightNames[i] == requestedWeightName) {
            weightIndex = static_cast<int>(i);
            break;
```

UI

```
def analyzers(self, dframe):
    dframe2 = dframe.Define("event_weights","GetAllWeights{}(EventHeader)")
    selected_weight_functor = ROOT.GetWeightByName("rwgt_4")

    dframe3 = dframe2.Define("selected_weight",selected_weight_functor,["EventHeader", "_EventWeightNames"])
```

Output



Conclusion

It's all circular

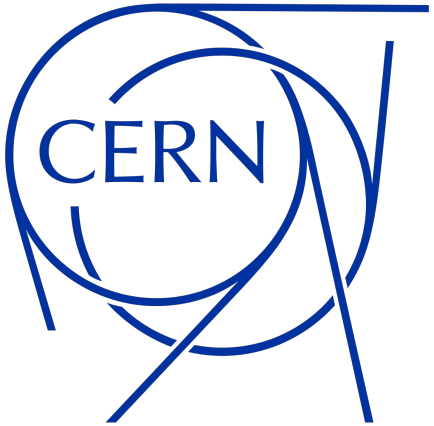
FUTURE CIRCULAR COLLIDER
Expanding our Horizons

Conclusion & outlook

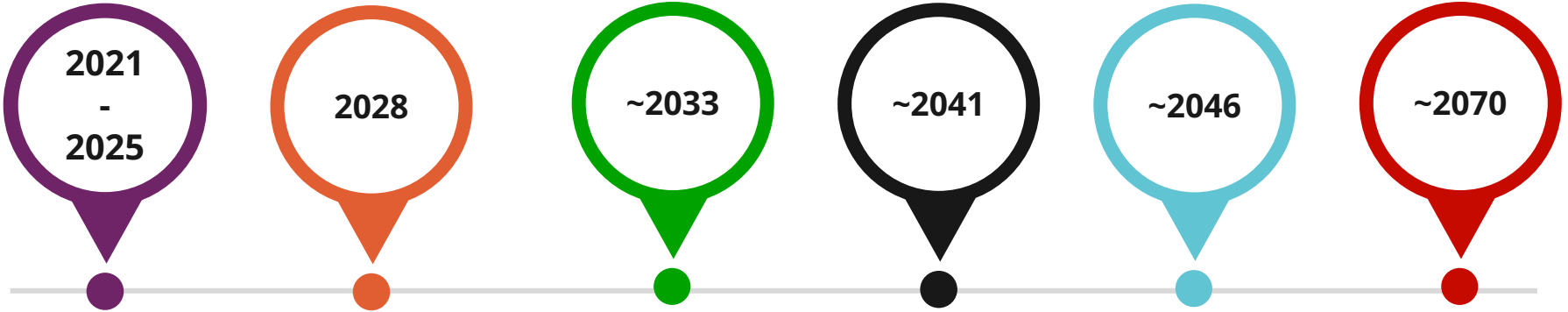
FUTURE CIRCULAR COLLIDER
Expanding our Horizons

- FCCAnalyses is a common thread, safely embedded in key4hep, that successfully supports studies across a vast phase space, with a user-friendly python interface and following guiding principles to
  - Support different workflows
  - Support different infrastructures
  - Keep pace with the underlying ecosystem
  - Provide as much of the common analysis chain as is sensible

- And will be put up to further trials & challenges in the next phase of the project!
- Built by the community that uses it, come help shape what's next
  - FCCAnalyses repo | FCCAnalyses forum | Analysis tools meetings
- Let's compare notes - where do we overlap? Do you have ideas for us?


CERN

The FCC integrated program: Timeline

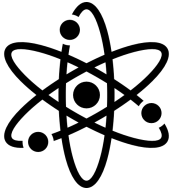


Master's student

PhD graduation

2nd post-doc

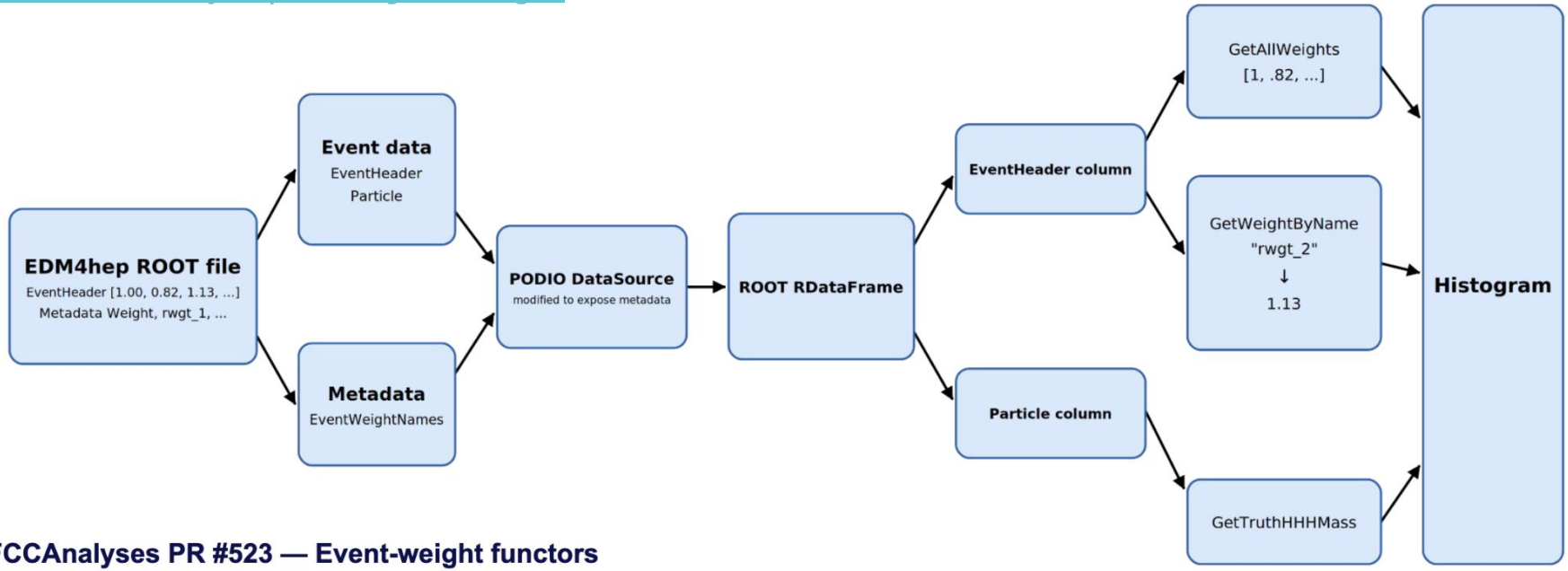
Achieve tenure



Retired

Event weights implementation

Slide from Zeynep Eda Çabukoğlu



FCCAnalyses PR #523 — Event-weight functors
PODIO PR #996 — DataSource metadata access

https://github.com/HEP-FCC/FCCAnalyses/pull/523
https://github.com/AIDASoft/podio/pull/996