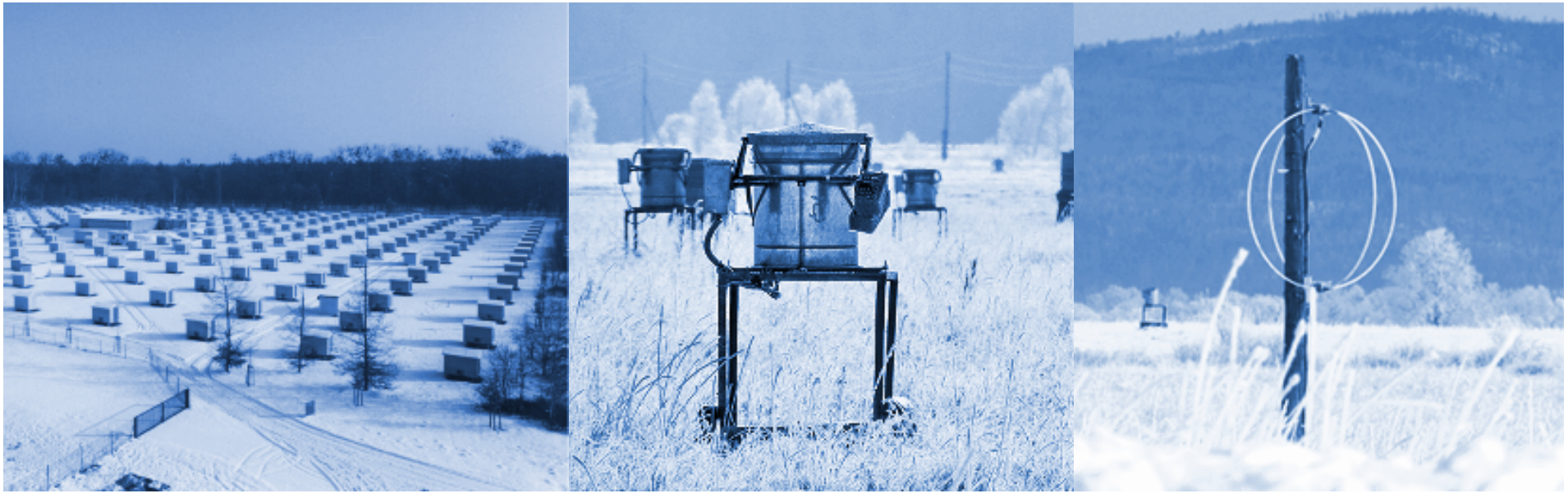
Handling the metadata for CORSIKA simulations (SiMM).



D. Kostunin, V. Lenok

CORSIKA Cosmic Ray Simulation Workshop, Karlsruhe 2019

HELMHOLTZ GEMEINSCHAFT

DESY.

Disclaimer

- The talk is focused on the problem of the coherent handling of large number of simulations ($> O(10^6)$) and unrelated to physics
- The approach and software described in the talk are:
  - developed five years ago (with few later bugfixes)
  - maintained by 2 people

  $\Rightarrow$ partly consisting of outdated tools & partly documented

  ... used for:

  ... TAIGA ($\gamma_{Ch.}$ + radio + e/μ)

  ... AERA (radio)

  ... Baikal-GVD (μ_{CR}) ?
- The initial goal was to handle CoREAS simulation for Tunka-Rex

  $\Rightarrow$ inclination towards radio simulations/measurements

`https://bitbucket.org/tunka/simm/` (GNU GPL v3+)
First implementation was presented in 2013

Motivation

- Metadata for steering with version control
  - CORSIKA configuration (hadronic model, flags, etc)
  - Environment (atmosphere, magnetic field, etc)
  - Detector configuration (layout, altitude, etc)
- Sophisticated simulation strategies
  - Computing time reduction (CONEX→CORSIKA→CoREAS fits)
  - Template fits (radio)
  - Run-wise simulations (IACTs)
- Metadata for CORSIKA output
  - Indexing the major parameters ($X_{\max}$, N_e, N_μ, etc.)
- Avoid direct writing steering files
- Fine tuning, scaling, freedom from large overhead
- Interface for data analysis software

\+ Metadata & index tools for Data Life Cycle / Data Centers

The concept of SiMM (SiMulation Manager)

- All metadata stored in the central database
- User defines simulation/evaluation strategy ("sequence")
- The simulations run on grid, the order is defined by the sequence

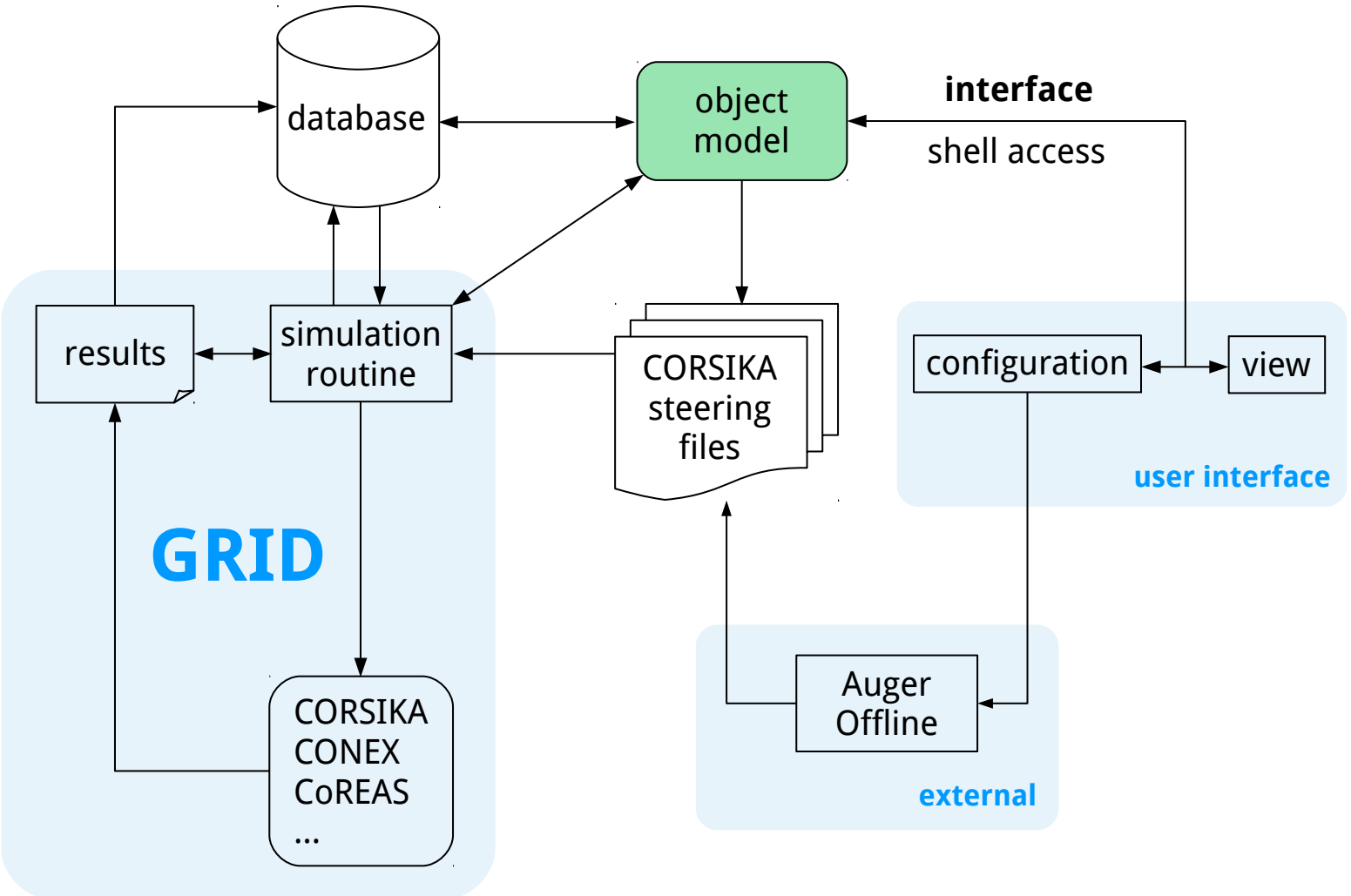
Two major modes:

- `step`: $T_{\text{exec}} \approx 0$ – configuration and evaluation of simulations
- `run`: $T_{\text{exec}} \gg 0$ – running simulation on distributed HPC

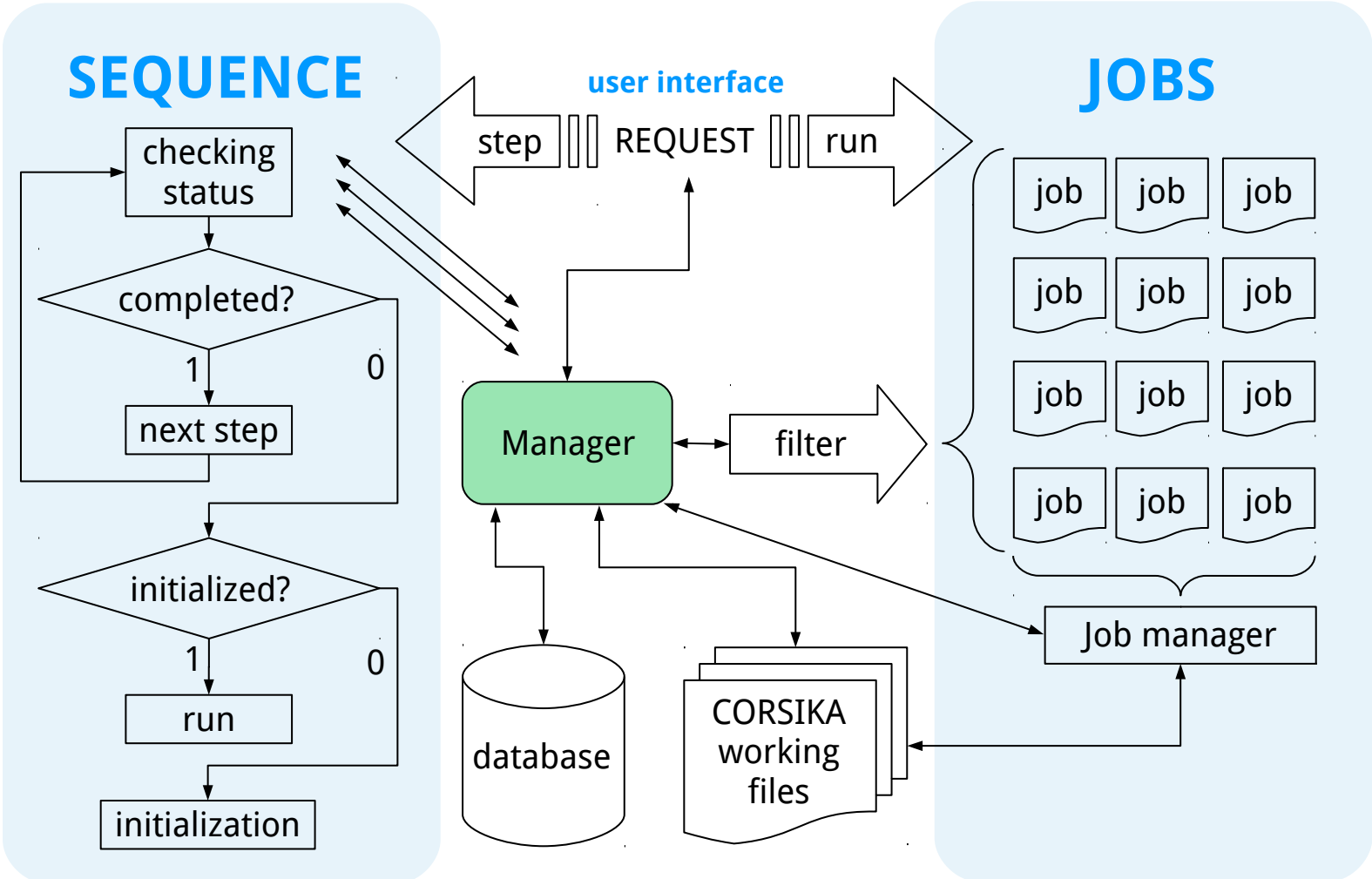
`SiMM` is deployed and used at IKP KIT, SCC KIT, ISDCT

Few examples:
Tomographic radio simulations: `10.1016/j.astropartphys.2015.10.004`
Radio template fitting: `10.1103/PhysRevD.97.122004`

Principal scheme



Object model



The structure of the database

Configuration of CORSIKA

- `corsika`

Configuration of sequences

- `sequence`
- `relations`

Seeding information

- `corsika_random_seed`
- `magnetic_field`
- `simulation`

Output data

- `hillas_fit`

Third-party tables

- `cherenkov_reconstruction` (storage for Tunka-133 events)
- `reconstruction` (storage for Tunka-Rex events)
- `geomag_angle` (not a table, but geometry routine)

Simulation metadata

- id
- run_id
- seq_id
- corsika_id
- type
- corsika_seed
- reas_seed
- primary_particle
- energy
- theta
- phi
- magnetic_field
- fit
- status

The features of the SiMM engine

- `Python 2(?)` + `sqlalchemy`
- `MySQL` database
- Plug-in system for sequences (interface, configuration, examples)
- Sequences for radio, particles and Cherenkov light
- Writers for: `CORSIKA`, `CoREAS`, `Offline`
- `RdCoREASSimulationCreator` module for Radio Offline
- Bash utilities: backup, repair, plotting, etc.

Example sequence

```
import core.Schema
from core.Schema import Step, Sequence, Simulation
from sqlalchemy.orm.session import make_transient
from Engine import DummyConexExample, \
                   DummyCoreasExample, \
                   DummyConfiguration, \
                   DummyInfoExample
import os

dirname, filename = os.path.split(os.path.abspath(__file__))
config = DummyConfiguration(dirname + "/config.rc")

Sequence = Sequence()
Sequence.name = "dummy"
Sequence.description = "This is dummy sequence"

def init(sequence) :
  sequence.add(DummyInfoExample)
  sequence.add(DummyConexExample)
  sequence.add(DummyCoreasExample)
  sequence.add_relation("conex")
  sequence.add_relation("coreas")
  sequence.config = config
```

Interface

```
[SIMM.core] Initialization
[SIMM.core] Loading config
[SIMM.Configuration] Reading
                       /path/to/simm/config.rc
                     Loading database
                     Loading simulation
                     Loading detector
                     Loading atmosphere
                     Loading offline
                     Loading backup
                     Loading misc
[SIMM.core] Connecting to database
[SIMM.core] Reading sequences
            Found 2 sequences
            Checking tunkaRex
            Checking dummy
[SIMM.MF(tunka)] Initialized with
                   T = 60.3181 uT; I = 71.7571; D = -2.7625
                   North = 18.8604038444 uT; East = -0.910054399618 uT;
                   Vertical = 57.2863871738 uT;
                   Horizontal = 18.8823471047 uT
[SIMM.core] Welcome to SIMulation Manager v1.0.0
[SIMM.core] Nothing to do
[SIMM.core] Bye!
```

Usage (step mode)

```
simm.py --sequence dummy --mode step
```

```
[SIMM.core] Using sequence dummy
[SIMM.dummy.DummyInfoExample] Sequence information
[SIMM.dummy.DummyInfoExample] Registered steps:
[SIMM.dummy.DummyInfoExample]   DummyInfoExample
[SIMM.dummy.DummyInfoExample]   DummyConexExample
[SIMM.dummy.DummyInfoExample]   DummyCoreasExample
[SIMM.dummy.DummyInfoExample] Registered relations:
[SIMM.dummy.DummyInfoExample]   conex
[SIMM.dummy.DummyInfoExample]   coreas
[SIMM.dummy.DummyInfoExample] Completed
[SIMM.dummy.DummyConexExample] Running
                               Making CONEX simulation
[SIMM.writer] File : /path/to/sim/data/123/RUN000123.inp
[SIMM.core] Bye!
```

Usage (run mode)

```
simm.py --sequence tunkaRex --mode run --id 123 -o dry
```

```
[SIMM.core] Found 1 simulations to run
[SIMM.core] Taking first. ID = 123
[SIMM.sim(554)] Starting run
[SIMM.sim(554)] CMD :
  export FLUPRO=/path/to/flupro/fluka-cluster ;
  cd /path/to/corsika/corsika-74000-cluster/run ;
  ./corsika74000Linux_QGSII_fluka_thin_curved_coreas < \
  /path/to/sim/data/123/RUN000123.inp 1> \
  /path/to/sim/data/123/RUN000123.log 2>> \
  /path/to/sim/data/123/RUN000123.err
[SIMM.sim(554)] Dry run! Exiting...
[SIMM.core] Bye!
```

Contribution to CORSIKA 8 ?

... after discussion at this workshop

- At the moment, there is no steering "user-friendly" utilities in C8
- C8 (sub)sequences configuration can be simply implemented/integrated in SiMM

we can contribute to

- Python binding/wrapper/interface for C8
- Metadata plugin (based on SiMM)

future actions

- Move SiMM to IKP gitlab (hg $\rightarrow$ git)?
- Using common steering/reading library, connect metadata as plugin?