The Open Standard for Particle-Mesh Data

Franz Poeschel (CASUS/HZDR)
DMA ST1 Synergy Workshop
DESY Hamburg

On behalf of the openPMD Community incl. content from
Axel Huebl (LBNL), Lipeng Wan (GSU), Remi Lehe (LBNL)
Norbert Podhorszki (ORNL), Junmin Gu (LBNL),
Maxence Thévenet (DESY), Erik Schnetter (PITP),

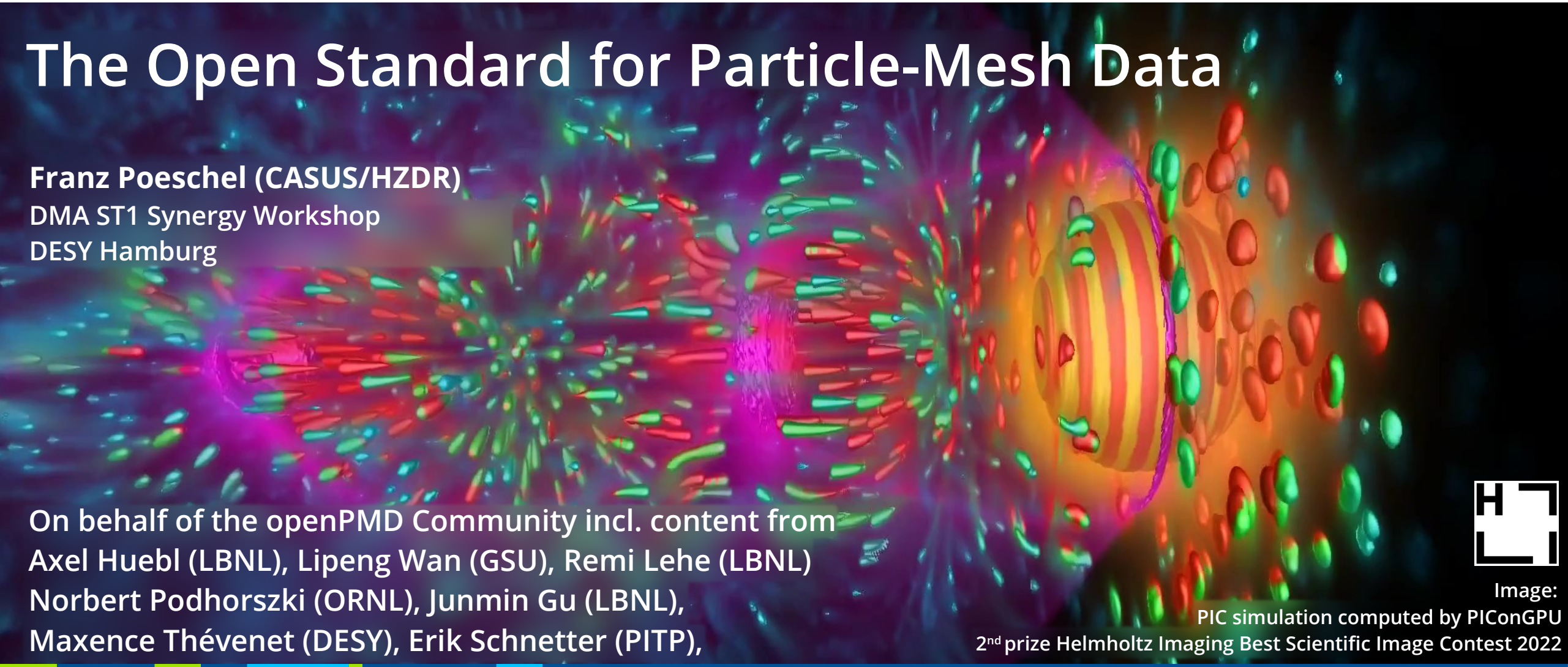
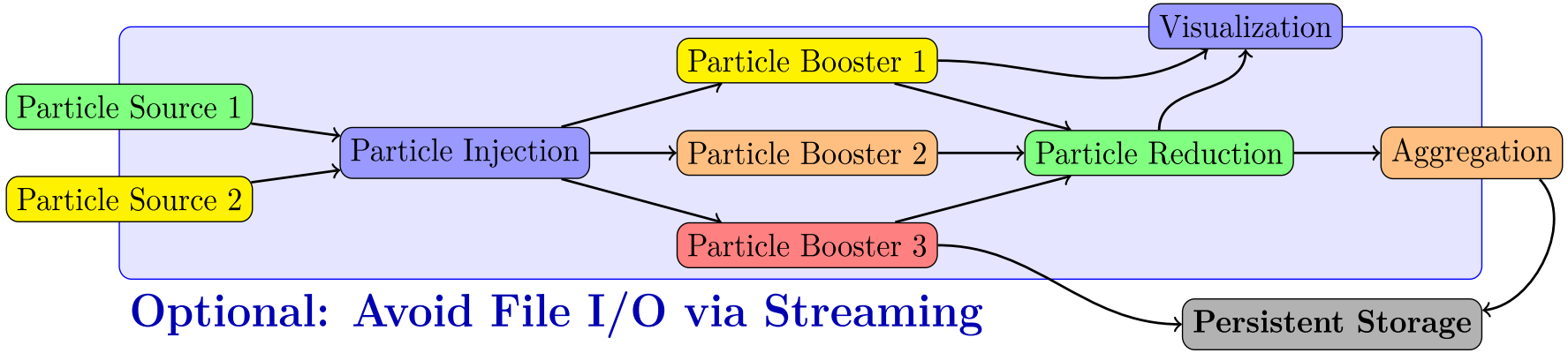


Image:
PIC simulation computed by PIConGPU
2nd prize Helmholtz Imaging Best Scientific Image Contest 2022

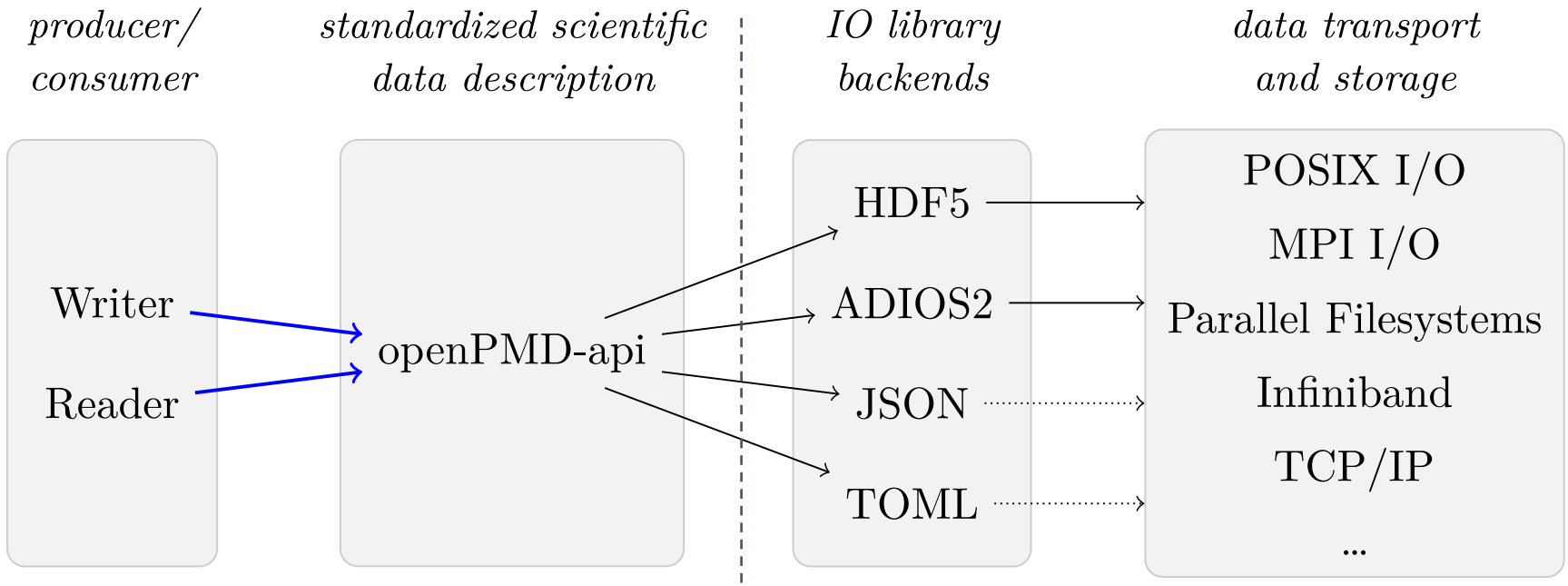
FAIR data meets Exascale I/O

Complex workflows
→ need F.A.I.R. data



	Titan	Summit	Frontier	*Growth Factor*
Peak Performance:	27 Pflop/s	200 Pflop/s	1.6 Eflop/s	~60
FS Throughput:	1 TiByte/s	2.5 TiByte/s	5.5 TiByte/s	~5
FS Capacity:	27 PiByte	250 PiByte	679 PiByte	25

Bringing F.A.I.R scalable I/O to software



C++17

Python

alpha: Julia

dev: C

I/O workflow expressed
in scientific domain language

Runtime parameters:

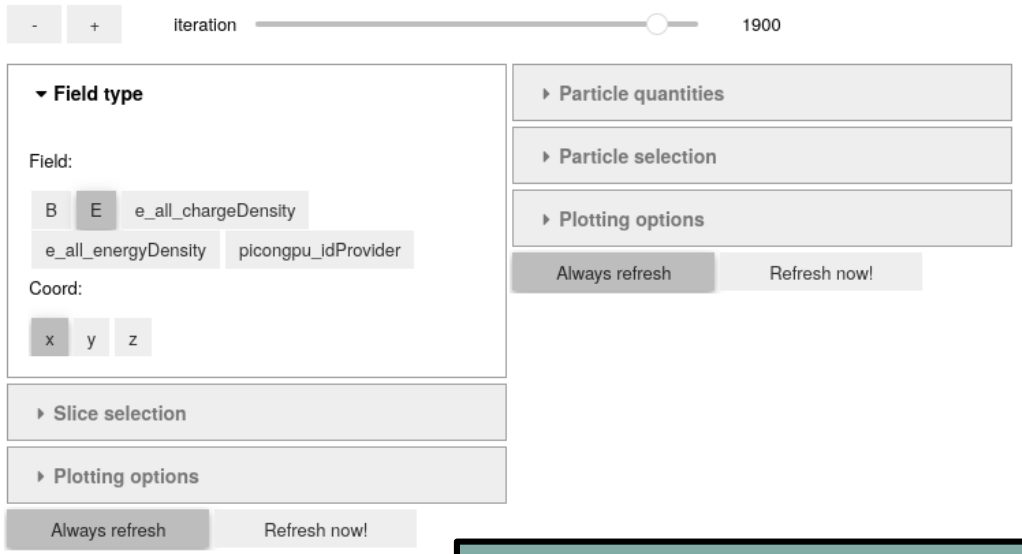
backend • data transport • streaming/file I/O
node-level aggregation strategies • compression
parallel FS configuration (e.g. OST usage)
buffering and serialization (e.g. chunking)

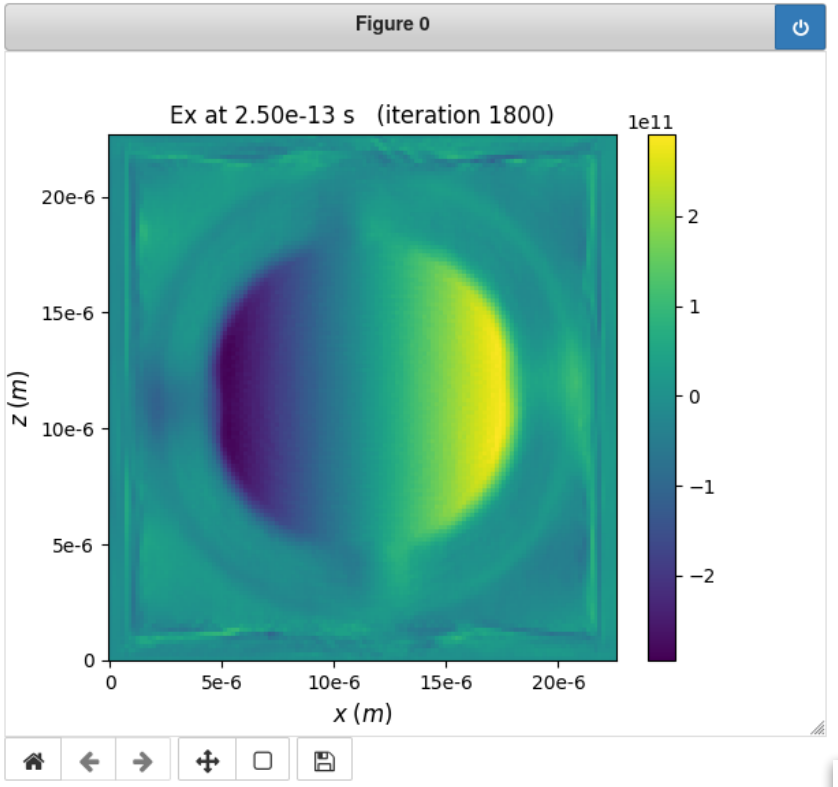
Analysis and Visualization

```
In [1]: import numpy as np
        %matplotlib notebook
        # or `%matplotlib inline` for non-interactive plots
        # or `%matplotlib widget` when using JupyterLab (github.com/matplotlib/jupyter-matplotlib)
        import matplotlib.pyplot as plt
        from openpmd_viewer import OpenPMDTimeSeries

In [2]: # Replace the string below, to point to your data
        ts = OpenPMDTimeSeries('/home/franzpoeschel/singularity_build/pic_run/openPMD')

In [3]: # Interactive GUI
        ts.slider()
```





openPMD/openPMD-viewer

Standardization of data

→ integration into modern scientific compute workflows