LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

ERUM DATA HUB

Bundesministerium für Forschung, Technologie und Raumfahrt

DEMOS Expert Interviews

Lorenz Gärtner, Caspar Schmitt

July 17, 2026

Who we asked

5 expert interviews + 10+ Belle II user-side interviews

Peter Fackeldey	junior	experiment	developer	evermore, scikit-HEP
Sabine Kraml	senior	theory	dev + user	SModelS, spey
Stefano Moneta	junior	experiment	user	pyhf practitioner
Danny van Dyk	senior	theory	developer	EOS
Lukas Heinrich	senior	experiment	developer	pyhf lead

Caveat: all interviewees are from the (particle-)HEP sphere — HEP bias.

Usability first

consensus

Minimize time-to-success: copy-paste examples; ∼3 lines to a first result [Heinrich]

- Usability over reproducibility: easy installation on any system [Fackeldey]
- Interactive execution layers — swap distributions easily, cf. LoRA fine-tuning [Fackeldey]
- Make assumptions, defaults & optimizer settings explicit — in docs or in the model JSON [Moneta]
- Compact models: current HS3 files too long for LLM parsing/exploration [Fackeldey]

Hackability: ideas as code

consensus

- "We communicate ideas with code"
- **Simplicity is key:** minimum code, clear structure ⇒ code-level understanding
- Minimal dependencies; keep backends replaceable
- Users replace, modify, or extend functionality with minimal friction [Moneta]
- Extension classes (à la `torch.nn.Module`) for new distributions
- Fast merge path into the main repo; CI-/LLM-assisted duplicate review [Fackeldey]
- Library of custom amplitudes, distributions, and model components [Moneta]

FlatPPL as an anchor

contested

- **FlatPPL as anchor:** use the advanced ongoing work
- Compare before committing: Pyro, NumPyro, Stan [Heinrich]
- Outcome stays open: adopt, adapt, or switch
- **Surface/core:** FlatPPL front end → portable JSON/ProtoBuf-style IR
- Learn from LLVM IR, Keras, ONNX, PPX / PyProb

Anchor, compare, decide

Anchor: FlatPPL work

Compare: existing PPLs

Modify: DEMOS primitives

Decide: evidence-based fit

Backend lessons: pyhf vs. evermore

Model-as-a-Service

contested

Moneta — cautious: containers hide complexity, add overhead, hinder reproducibility ⇒ optional side feature at most

Fackeldey — translators first: containers are the path of least resistance, prone to clutter; invest in HS3 ↔ Python translators

van Dyk — "model as an *open* service": versioned containers; immutable data (new DOI on change), mutable metadata

Heinrich — non-issue: the container effort barrier is self-enforcing; fine as-is

Combining models across analyses

open

A naming scheme for systematics is the blocker [Kraml]

- Persistent identifiers for correlatable systematics — cross-experiment correlations are usually missing today [van Dyk] (Jung)
- May need to be community-specific; document it formally [Kraml]
- Simplification schemes for fast, stable inference on large combined models — key for EFT fits & theory [Kraml] (Barlow "effective sigma", spey)
- Closure tests across ROOT/HistFactory, pyhf, HS3: needed, but expensive and not systematically tracked [Kraml]

Scope & ecosystem

consensus

- DEMOS is ambitious — break it into smaller, well-defined components [van Dyk]
- Connect to existing infrastructure: HEPData, INSPIRE, and other repositories [van Dyk]
- Theory predictions are hard to standardize ⇒ prioritize **reproducibility** over one universal standard [van Dyk]
- Document the default theoretical model per measurement, reproducibly [van Dyk]
- Structured-prediction reference: POPxf — Polynomial Observable Prediction Exchange Format [van Dyk]

Decision points

for the consortium

1. Evaluate FlatPPL vs. external PPLs: what evidence changes course?
2. If FlatPPL remains the anchor, what must be adapted?
3. Commit to surface language + serialized core?
4. Containers: core feature, optional, or out of scope?
5. Who drives the systematics naming scheme?
6. How to accommodate ML serialization & big-data parsing?
   [Fackeldey]

Present at the reinterpretation workshop (Nov) and at pyhep.dev (Sept)

Main takeaways

Usability wins

minimize time-to-success

~3 lines to a first result

Simplicity is key

code-level understanding

users trust what they can inspect

Hackability

ideas as code

extensions; minimal dependencies

Naming scheme

blocker for combination

correlatable systematics