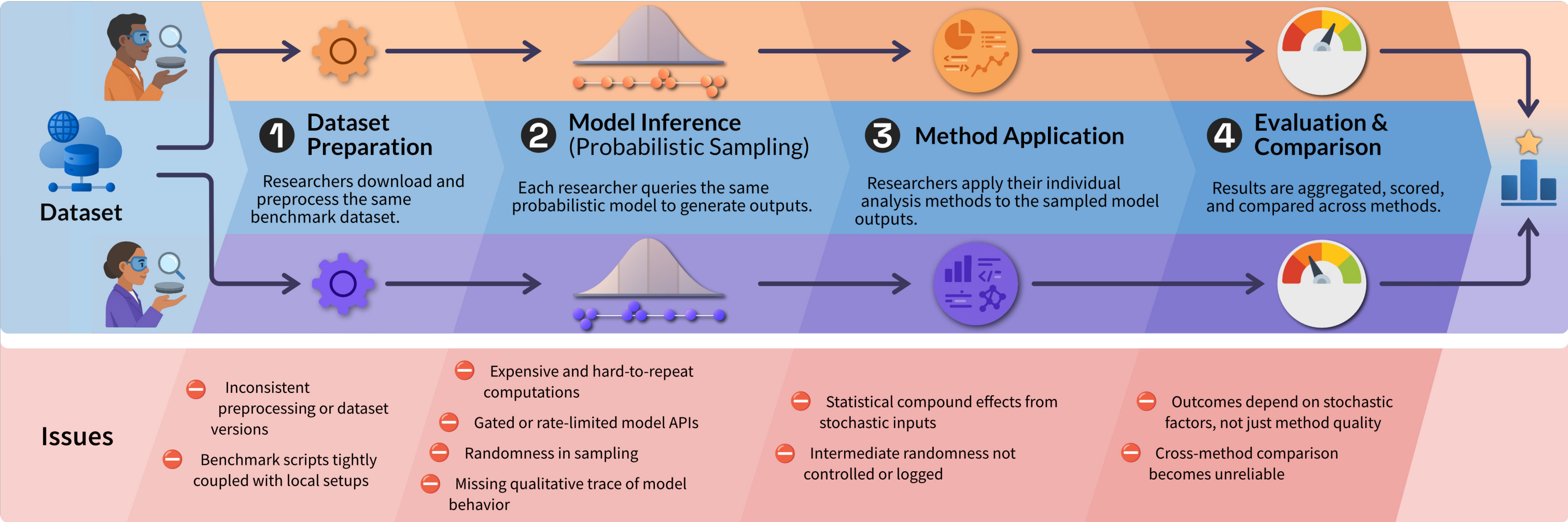# A Modular Framework for Benchmarking Evolving Probabilistic AI Systems

Philip Müller[1], Peter Steinbach[1]   [1] Helmholtz-Zentrum Dresden-Rossendorf
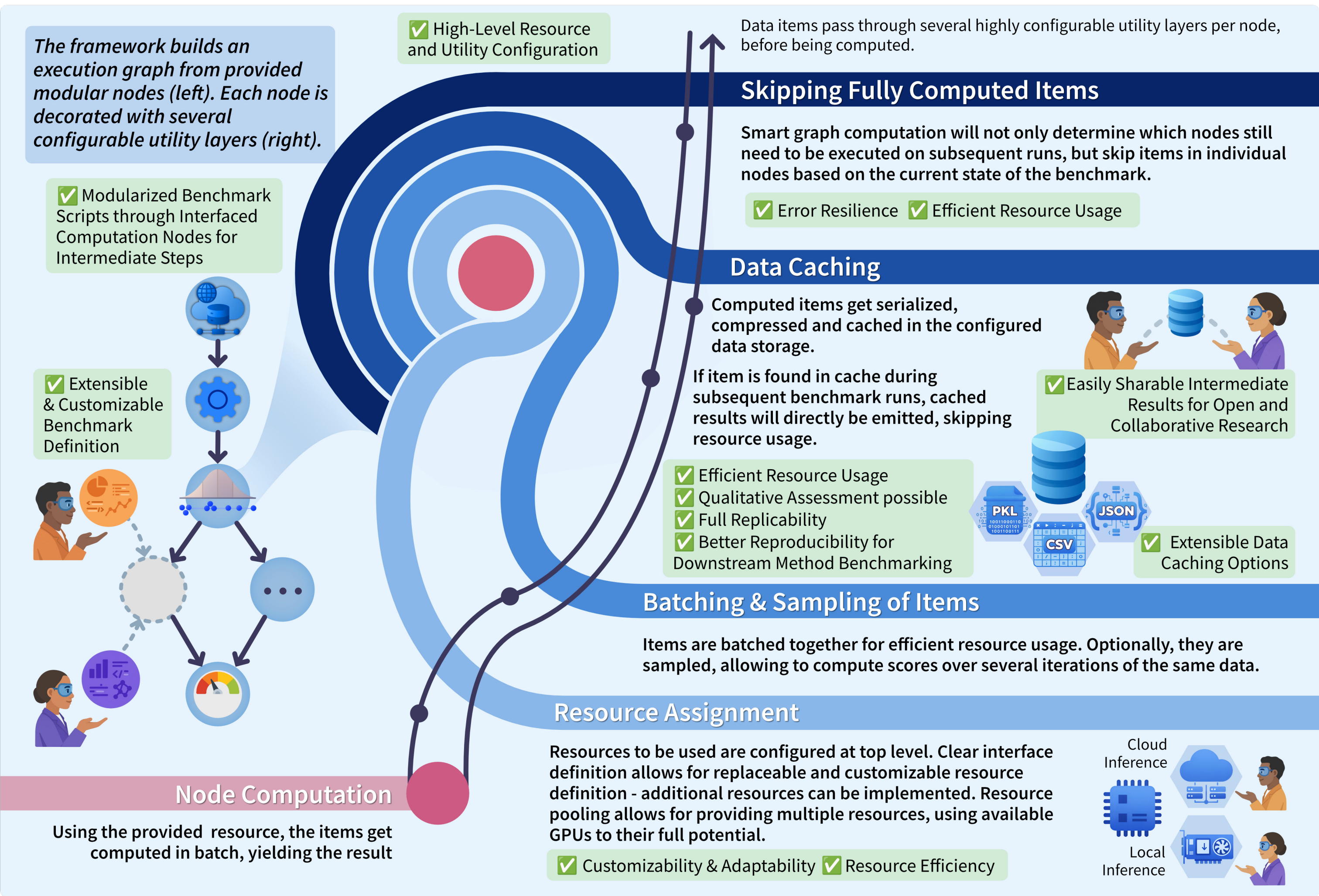
PROBLEM STATEMENT
Benchmarking Pipelines Using Probabilistic Models

Dataset

1 Dataset Preparation
Researchers download and preprocess the same benchmark dataset.

2 Model Inference (Probabilistic Sampling)
Each researcher queries the same probabilistic model to generate outputs.

3 Method Application
Researchers apply their individual analysis methods to the sampled model outputs.

4 Evaluation & Comparison
Results are aggregated, scored, and compared across methods.

Issues
- ⊖ Inconsistent preprocessing or dataset versions
- ⊖ Benchmark scripts tightly coupled with local setups
- ⊖ Expensive and hard-to-repeat computations
- ⊖ Gated or rate-limited model APIs
- ⊖ Randomness in sampling
- ⊖ Missing qualitative trace of model behavior
- ⊖ Statistical compound effects from stochastic inputs
- ⊖ Intermediate randomness not controlled or logged
- ⊖ Outcomes depend on stochastic factors, not just method quality
- ⊖ Cross-method comparison becomes unreliable

SOLUTION
Reproducibility-Centered Framework for Probabilistic Model Benchmarking

The framework builds an execution graph from provided modular nodes (left). Each node is decorated with several configurable utility layers (right).

✅ Modularized Benchmark Scripts through Interfaced Computation Nodes for Intermediate Steps

✅ Extensible & Customizable Benchmark Definition

Node Computation
Using the provided resource, the items get computed in batch, yielding the result

✅ High-Level Resource and Utility Configuration

Data items pass through several highly configurable utility layers per node, before being computed.

Skipping Fully Computed Items
Smart graph computation will not only determine which nodes still need to be executed on subsequent runs, but skip items in individual nodes based on the current state of the benchmark.

✅ Error Resilience   ✅ Efficient Resource Usage

Data Caching
Computed items get serialized, compressed and cached in the configured data storage.

If item is found in cache during subsequent benchmark runs, cached results will directly be emitted, skipping resource usage.

✅ Efficient Resource Usage
✅ Qualitative Assessment possible
✅ Full Replicability
✅ Better Reproducibility for Downstream Method Benchmarking

✅ Easily Sharable Intermediate Results for Open and Collaborative Research

PKL   JSON   CSV

✅ Extensible Data Caching Options

Batching & Sampling of Items
Items are batched together for efficient resource usage. Optionally, they are sampled, allowing to compute scores over several iterations of the same data.

Resource Assignment
Resources to be used are configured at top level. Clear interface definition allows for replaceable and customizable resource definition - additional resources can be implemented. Resource pooling allows for providing multiple resources, using available GPUs to their full potential.

✅ Customizability & Adaptability   ✅ Resource Efficiency

Cloud Inference

Local Inference

IMPACT STATEMENT
Advancing Reproducible and Transparent Benchmarking

This work strengthens trustworthy AI evaluation by enabling reproducible benchmarking for probabilistic and resource-intensive models. By decoupling benchmark logic from model execution and caching stochastic outputs, it ensures consistent comparisons across evolving systems while reducing computational cost. The framework supports longitudinal studies, shared intermediate results, and open collaboration, offering a scalable and transparent foundation for benchmarking in rapidly changing AI environments.

The Framework will be released at the end of January — star to get notified!

HZDR
HELMHOLTZ ZENTRUM DRESDEN ROSSENDORF