

SCIENTIFIC MULTI-AGENTIC SYSTEMS FOR HEP



Nesar Ramachandra



U.S. DEPARTMENT
of ENERGY

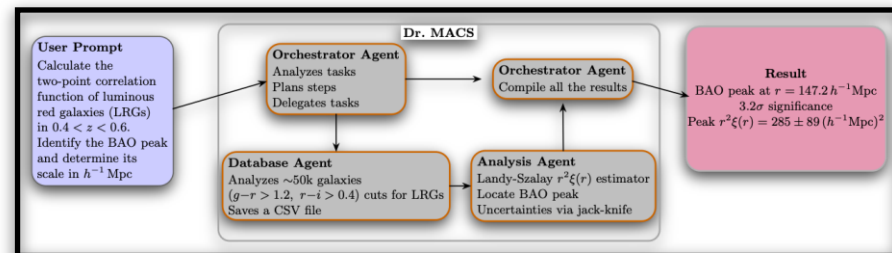
Argonne National Laboratory is a
U.S. Department of Energy laboratory
managed by UChicago Argonne, LLC.

Argonne
NATIONAL LABORATORY

Agentic systems on laptops (first generation)



InferA: A Smart Assistant for Cosmological Ensemble Data, Tam et al. (incl. Ramachandra, <https://arxiv.org/abs/2510.12920>)

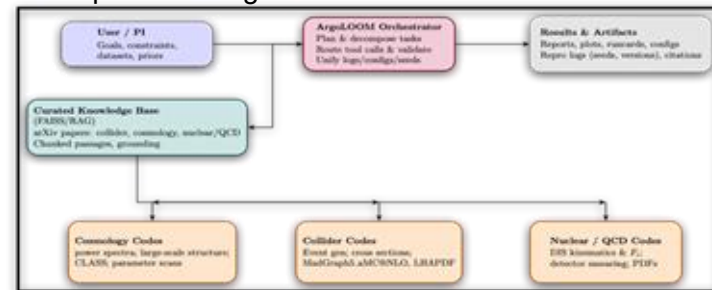


Dr MACS: agentic AI for astronomy, Jennifer Coburn, Nesar Ramachandra, Azton Wells et al.

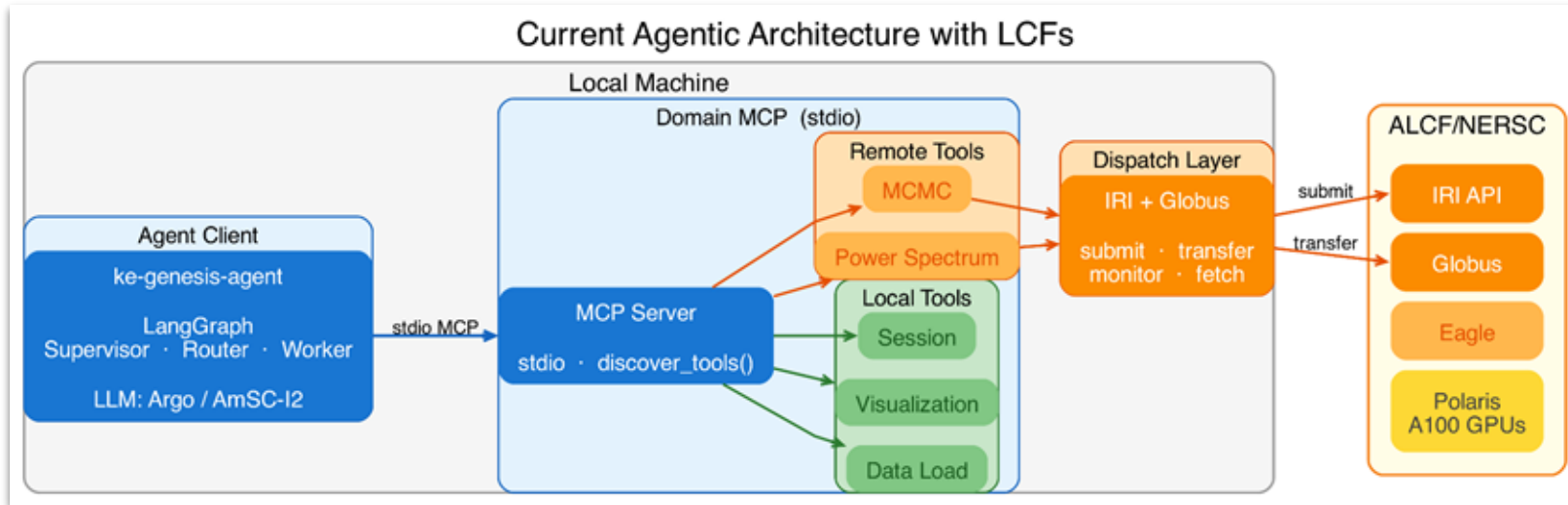
Ingredients of Multi-Agent systems – Tools, Servers, and databases:

- Large Reasoning models with generic tool access (web search, data base reading)
- Multiple expert interactions (Roles: Orchestrator, domain-expert, coding expert, critic)
- Local datasets (texts and numerical data) access, codes, tool interfaces.
- Python server to run codes, feed results back to LLMs

ArgoLOOM: agentic AI for fundamental physics from quarks to cosmos
<https://arxiv.org/abs/2510.02426>



Agentic systems + supercomputing (second generation)



Multi-Agent systems with HPC scaling: Harnesses on local systems, Servers on Supercomputers, APIs for control and transfer.

- Multiple persistently hosted servers for each sub-domain, linked to data on large storages.
- Repeated re-vibe-code some of the scripts is not needed – coding only when really needed, and within constraints.
- Domain experts to write, and benchmark the tools, and results
- Interfaces to large scale codes, all accessible via local agentic systems.

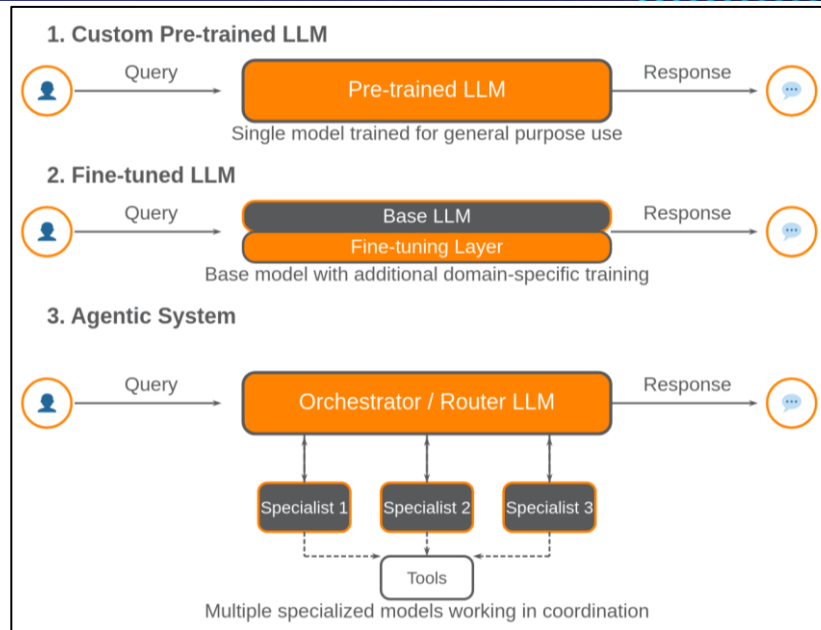
Why agentic systems for HEP?

Limitations in LLMs can be overcome with agentic systems

- Knowledge cut-off vs RAG/Knowledge databases
- Fabrication of synthetic data/codes vs custom tools
- Big-data handling

Custom Agentic systems vs commercial LLM harnesses

- Reasoning abilities of LLMs + coding expertise to be combined with scientific datasets
- Custom tools and custom harnesses provide a higher level of trust in the results
- Ability to work with HPC clusters and exascale codes
- Privacy and control concerns
- Scaling across large scientific user base
- Reusability vs token usage concerns

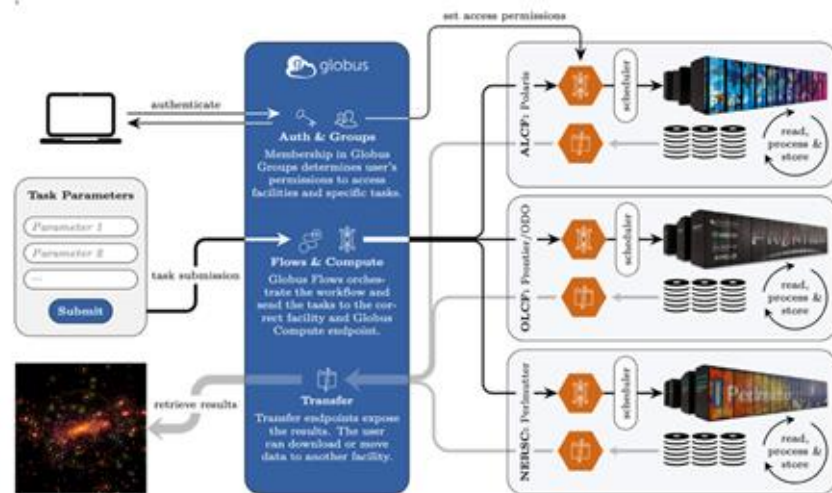


Infrastructure for HPC-scale Agentic systems

- Data currently resides on large computing facilities
- HPC resources are needed for data analysis tasks and to support agentic systems
- Longterm vision: Static and on-demand integration of data from major experiments and observatories (EIC, LHC, Rubin LSST, ...)
- Data and analysis and simulation integration points are currently at ALCF, NERSC, and OLCF
- Local models hosted on 1000s of GPU nodes.

Infrastructure products:

- ❑ API for LLM interfaces
- ❑ API for MCP servers
- ❑ Federated Compute
- ❑ Data Lakehouse
- ❑ Agent Skills
- ❑ Model training & tracking



Source: [OpenCosmo](#)

Agentic systems + Supercomputing systems

```

TERMINAL
python
README.md
compute.py 1 X

(cosmodev) nramachandra@roomba: /data/a/cpac/nramachandra/Projects/AmS
C/ke_genesis_agent (master)$ python examples/05_agent_domain_test.py
- arxiv_agent
- download_arxiv_paper
- download_full_arxiv_paper
- list_files
- read_text_file
- search_arxiv

Query: Using the observational data from eBOSS, compare the linear mat
ter power spectrum P(k) for LCDM and LCDM with massive neutrinos (sum
_mnu=0.10 eV). Create a visualization showing both power spectra with
the data.

Running agent...
[dispatch] Dispatching compute_power_spectrum to Polaris...
[dispatch] Local staging: /data/a/cpac/nramachandra/Projects/AmS/ke_
genesis_agent/.staging/compute_power_spectr_260409_090750_8qbjbhyz
[dispatch] Uploading 9 files to Eagle via Globus Transfer...
[dispatch] Globus upload task: 7ee0c6a4-341d-11f1-8e4c-02535127e3d7
[dispatch] Job submitted: 7066622.polaris-pbs-01.hsn.cm.polaris.alcf.
anl.gov
[dispatch] Output ready: results.json (2058)

=====
HARD FAIL: DISPATCH_TO_POLARIS=true but Polaris dispatch failed.
Error: Error: CosmoSevereError:

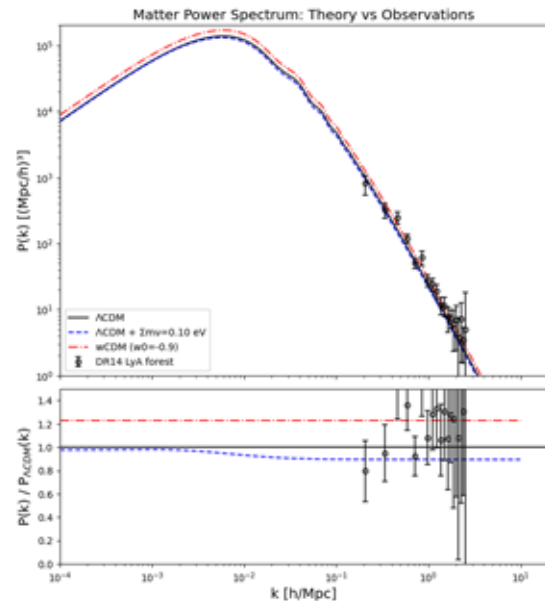
Error in Class: Class did not read input parameter(s): m_ncdm
. Ensure params is dict and k_values is list of floats.

Fix: resolve the Globus Transfer permission issue, or set DISPATCH_TO
_POLARIS=false for local execution.
See ISSUES.md for details.

=====
[dispatch] Processing request of type CallToolRequest
[dispatch] Dispatching compute_power_spectrum to Polaris...
[dispatch] Local staging: /data/a/cpac/nramachandra/Projects/AmS/ke_
genesis_agent/.staging/compute_power_spectr_260409_090844_8dv2b830
[dispatch] Uploading 9 files to Eagle via Globus Transfer...
[dispatch] Globus upload task: 9ee33307-341d-11f1-8144-02535127e3d7
[dispatch] Job submitted: 7066632.polaris-pbs-01.hsn.cm.polaris.alcf.
anl.gov

Update is ready, click to restart.
Ln 24, Col 5 Spaces: 4 UTF-8 LF Python Finish Setup 3.10.12 Python 3.10.12

```



Plot generated by the power-spectrum agent. The power spectra lines are generated by CLASS tool with parameters decided by the agentic system.

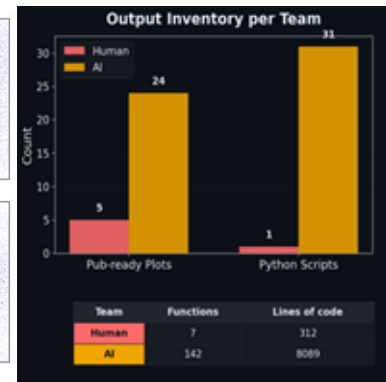
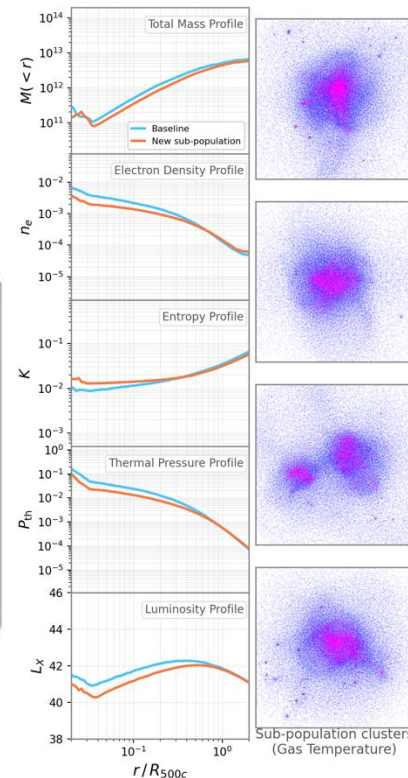
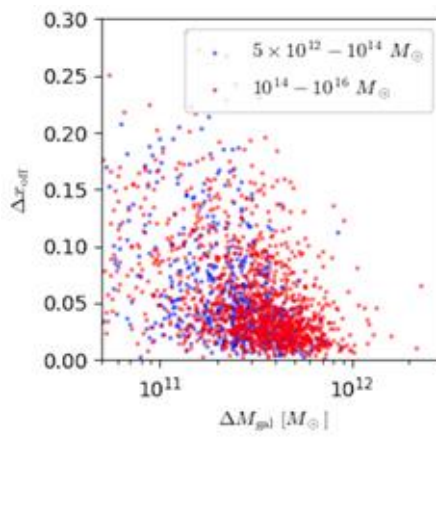


Validation and verification: Physics checks

One of the hardest things in Agentic-AI is confirming the systems provide useful science results.

Comparison of physicists with and without AI-companion tests (A/B Tests):

- Not scalable but insightful
- Comparisons of time taken in development vs verification, reasoning decisions, artifacts, final results.
- Compliments standard QA validation sets and coding benchmarks.
- Human domain expertise is crucial with AI usage too, particularly when we are dealing with novel steps and results.



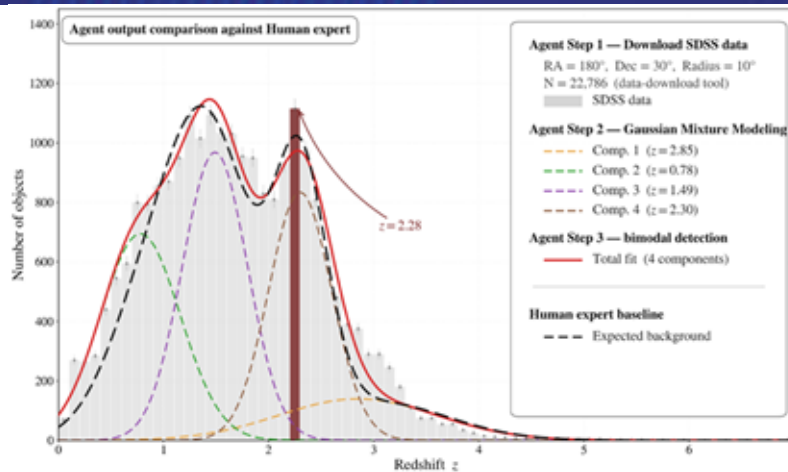
Comparison of analysis of 2 research teams with and without AI use

Validation and verification: Physics checks

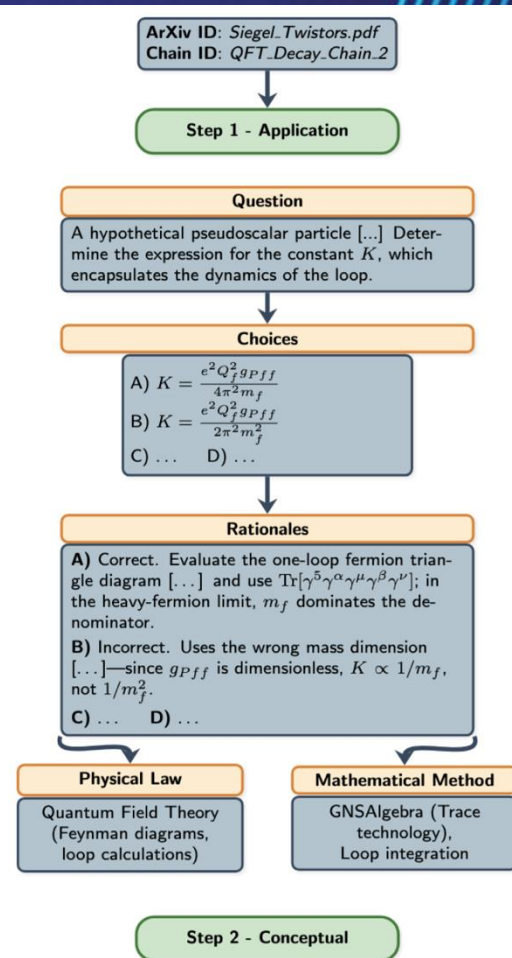
Automated benchmarks sets with multiple-turns, testing on complex tests:

Benchmarking datasets close to real-world tests – quantitative metrics, ground truth steps, progressive hints with penalty.

- Scalable with caveats.
- Manageable with short queries

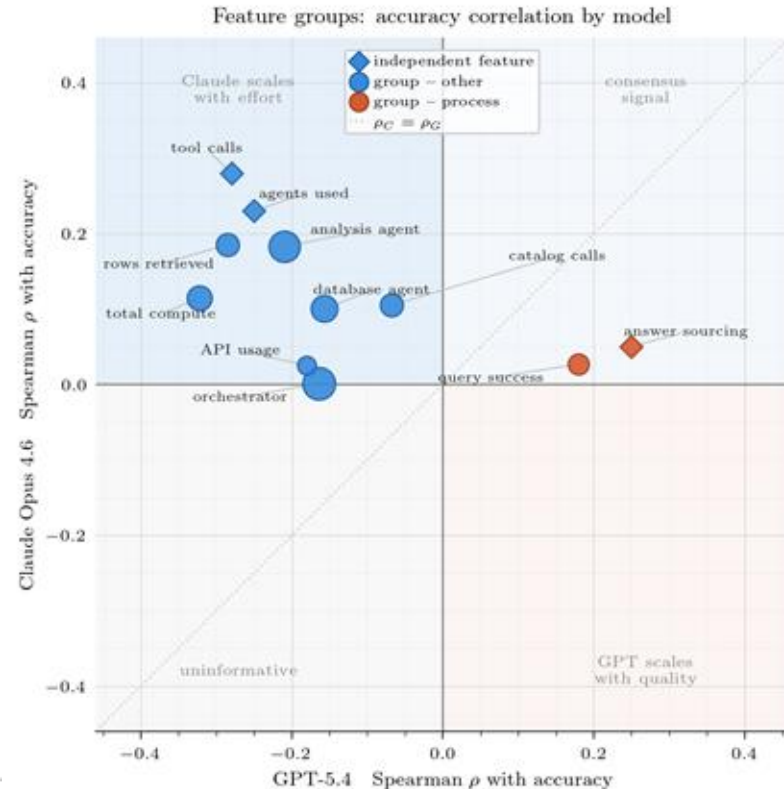
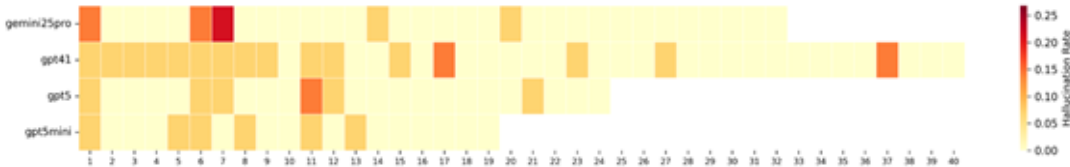


Right: Reasoning steps tested in automated multi-step benchmark systems (Gerou, Ramachandra & Wells, 2026)

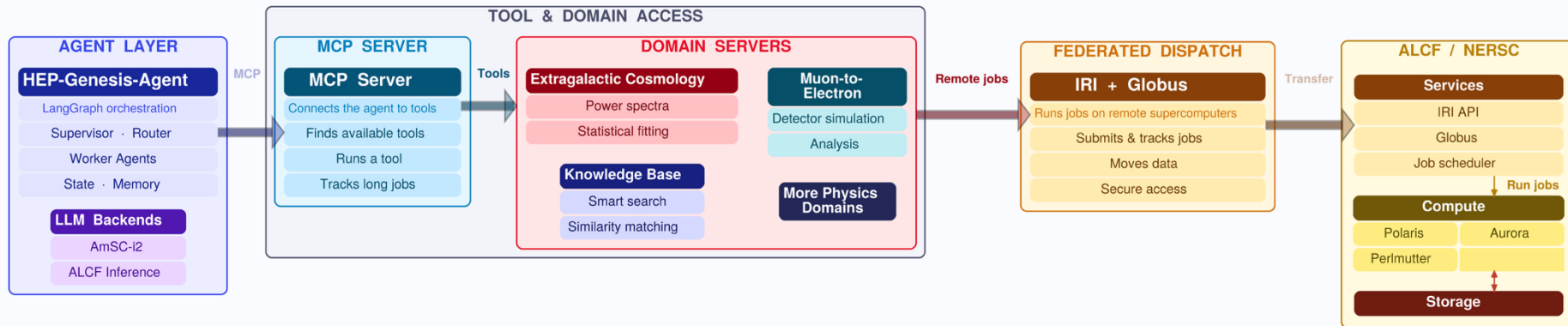


Validation and verification: Architecture checks

- Testing design choices – ablation studies.
 - Tests entire agentic system, with different LLM backends.
 - At the component level – i.e., correlating the features of the agentic systems with accuracies.
 - Initial results show the advantage of tool usage and multi-agent architectures.
- Tracking hallucination occurrences at different steps.
 - Mitigation strategies: working with files, provenance checks.



Agentic systems: Client + Server model



Multi-Agent systems with HPC scaling: Harnesses on local systems, hosted servers for each sub-field, APIs for control and transfer.

- Integration with Genesis Mission projects (HEP-Knowledge extraction and Quarks2Cosmos): customized agentic systems for Physics
- Tutorials/slack channels/public github repos are available outside the collaborations.
 - Significant interest from science collaborations, universities and industry partners.
- Bridge between DOE-Genesis platform and Science teams.
 - Topics: LSST-DESC data exploration, Mu2e studies, EIC simulation-based inference, DESI clustering pipelines, beamline optimization, Foundation models for HEP and NP,

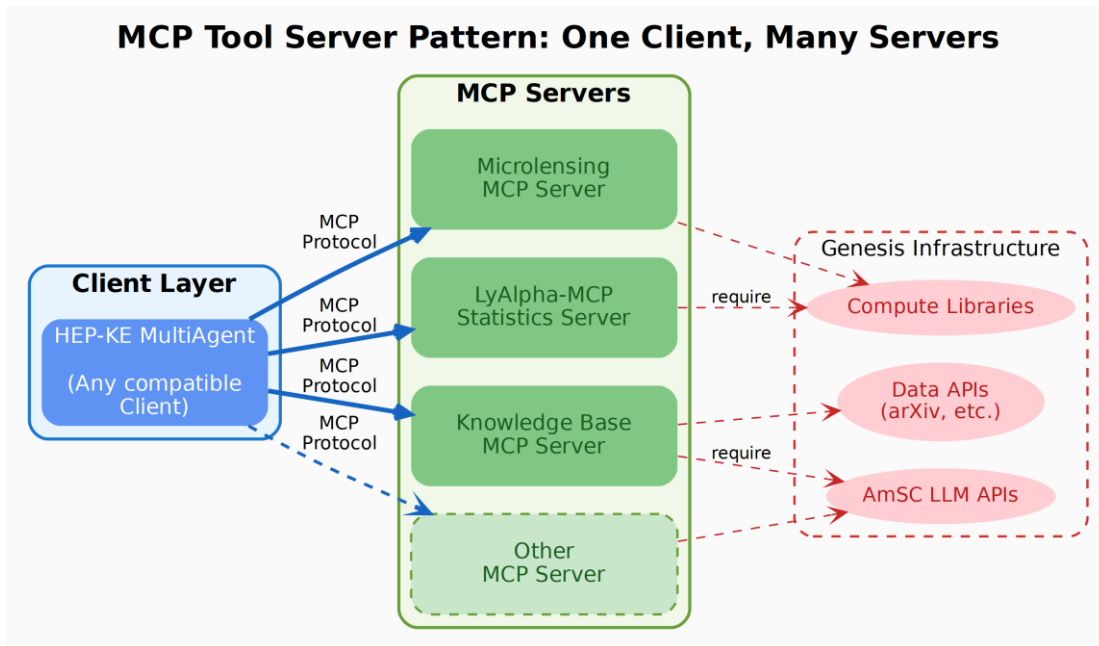
Interfacing between LLMs and tools

MCP: a standard interface between agents and tools

- Write the tool once and host on a server — any MCP-speaking client (our LangGraph client, Claude Code, CODEX) can call it.
- The server owns the science: the data files, and plotting all live server-side.
- The client only sees tool schemas and small results — fully isolated from the servers.

Multiple ways to manage MCP:

- stdio: client launches the server as a subprocess (same machine).
- streamable-http: server runs anywhere, clients connect over the network — covered today.



Looking forward

- Leveraging emerging infrastructure (from DOE-Genesis Mission and industry)
 - Self-improving models, autonomous learning loop systems.
 - Beyond-language systems, using world models)
- Focus on physics discovery results, not just experimenting with latest technology.
 - Faster deployment - feedback - improvement cycles.
 - Reliability checks in real-world scientific use cases.



Tutorials for building your own servers + clients: <https://hep-ke.github.io/multiagent-client-demo>

HEP-KE Agentic system (alpha-version): <https://hep-ke.github.io/hep-genesis-releases>