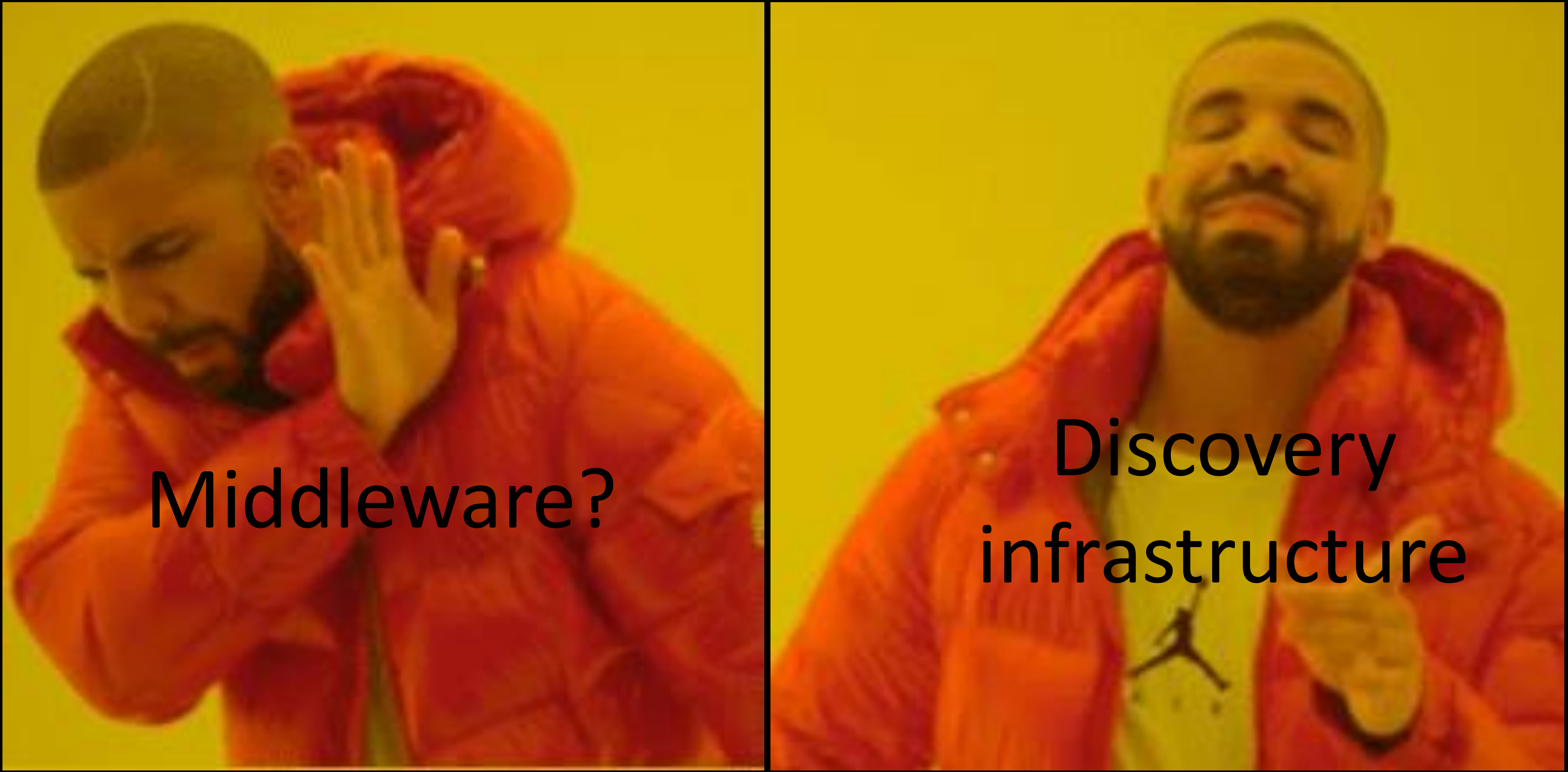
Mind the Middleware

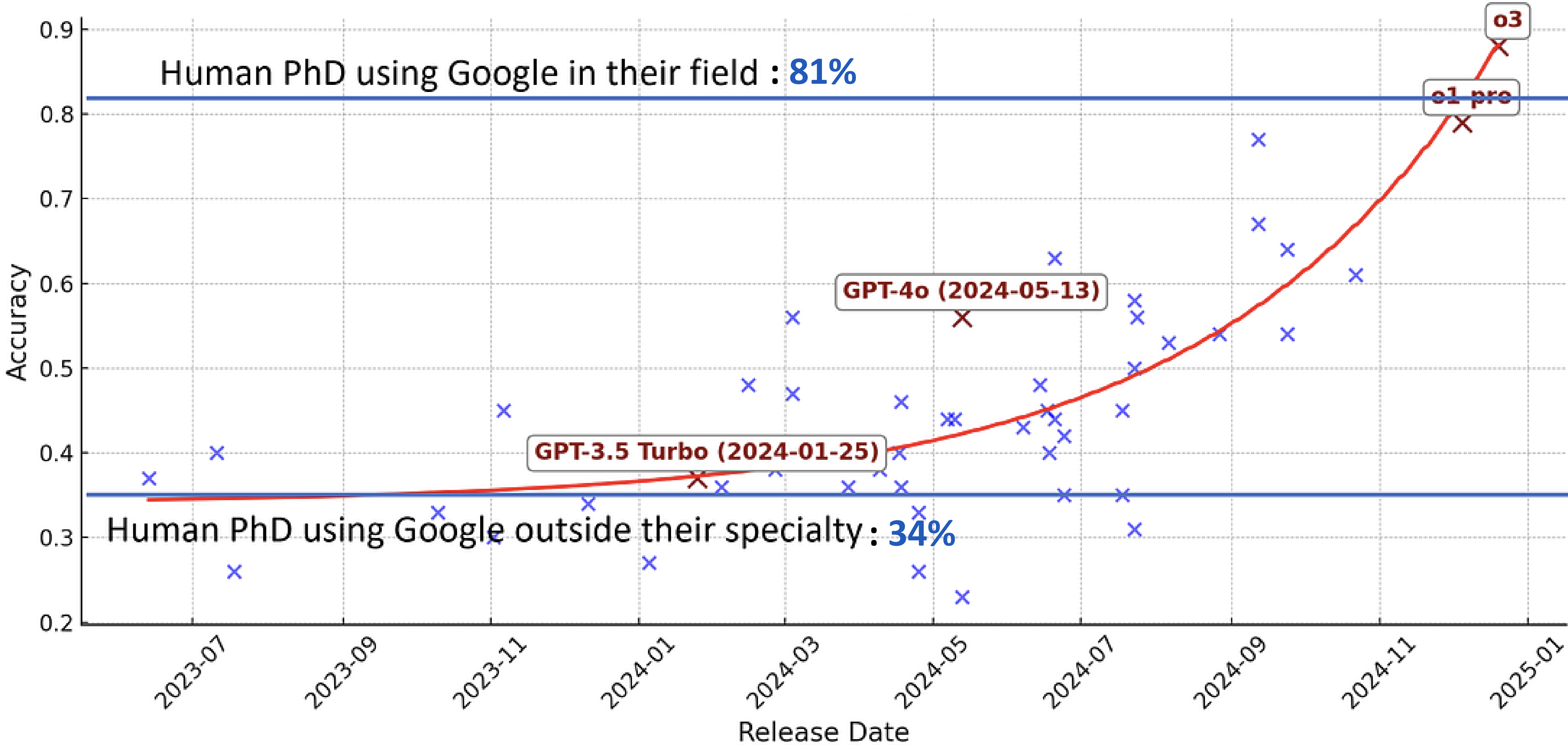
From Data to Discovery in Brain-Scale Agentic AI

Ian Foster
foster@anl.gov
foster@uchicago.edu

THE UNIVERSITY OF CHICAGO
globus labs
Argonne NATIONAL LABORATORY


Middleware?
Discovery infrastructure

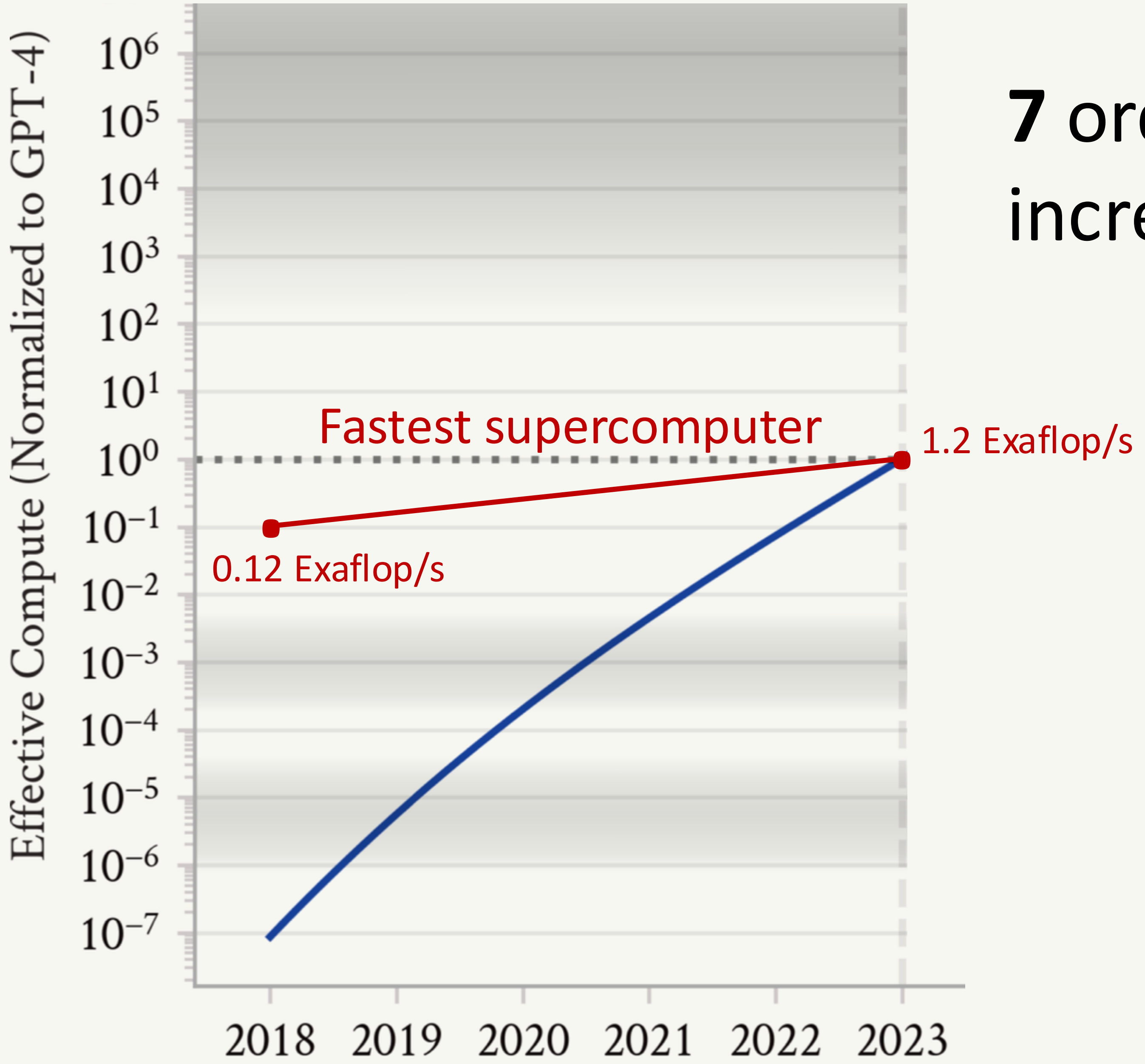
Graduate-Level Google-Proof Q&A test (GPQA), Diamond problems



https://arxiv.org/pdf/2311.12022

https://epoch.ai/data/ai-benchmarking-dashboard

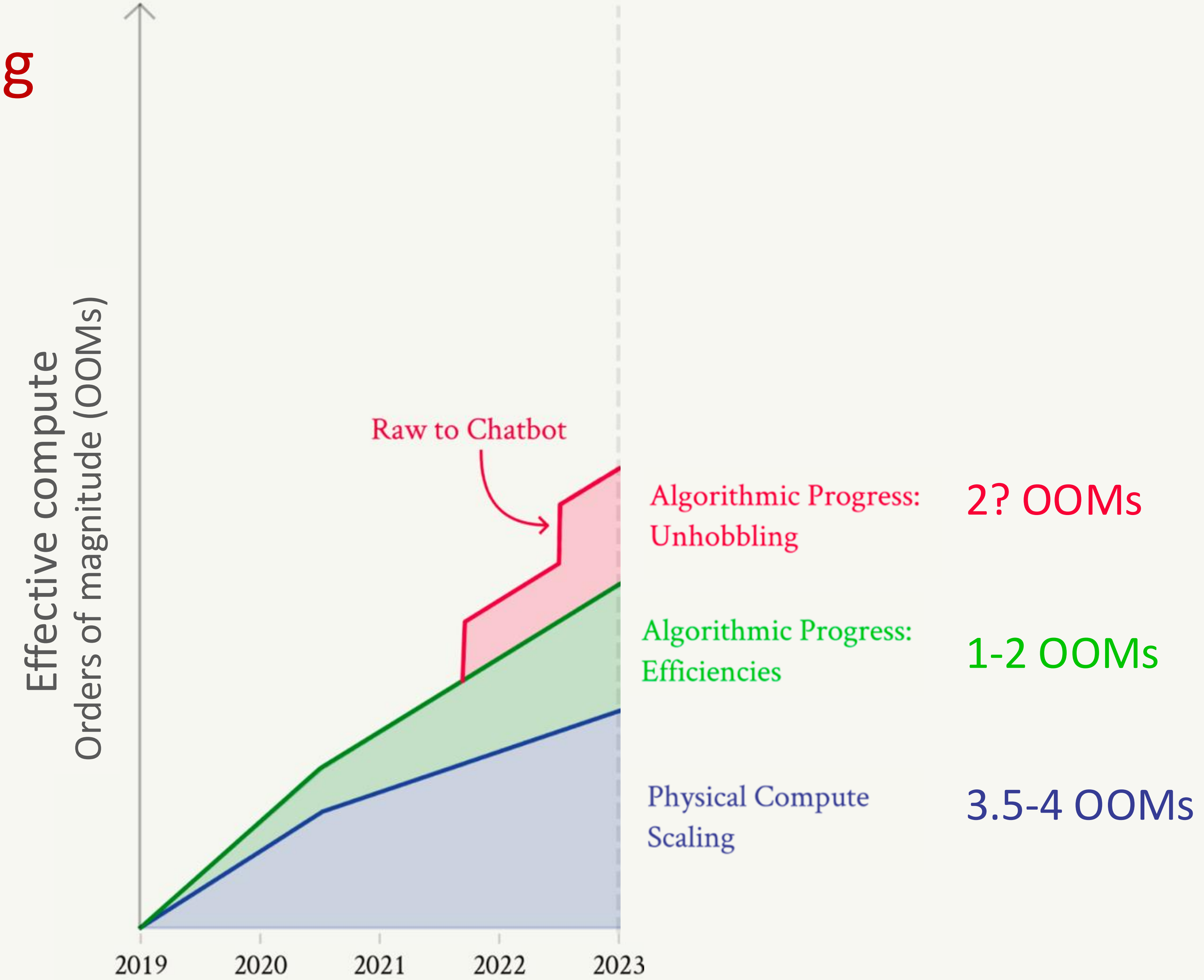
Aschenbrenner’s analysis: Scale up of effective compute



7 orders-of-magnitude increase in **5** years

Leopold Aschenbrenner https://situational-awareness.ai

Decomposing drivers of progress **2019-2023**

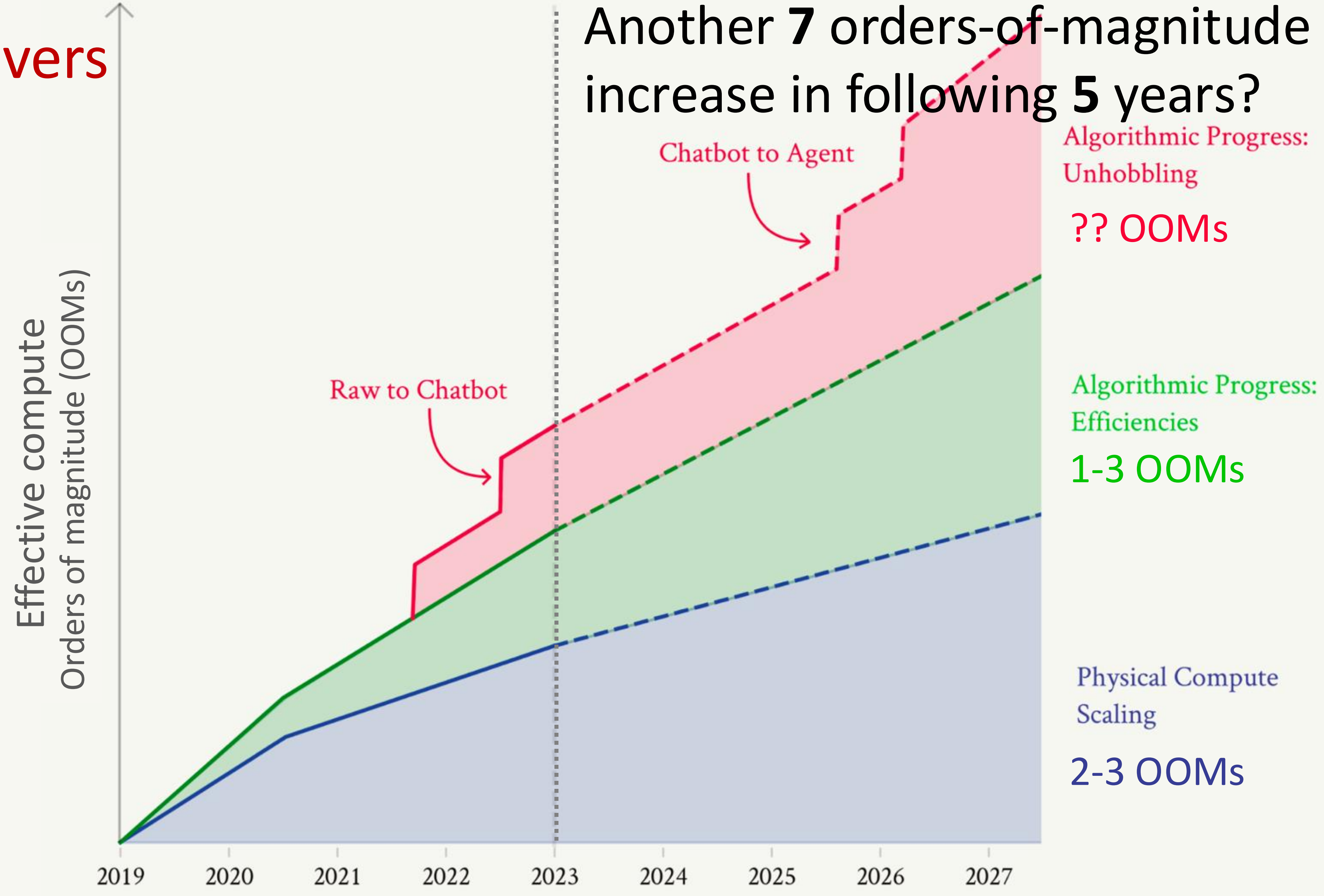


Rough illustration

https://situational-awareness.ai

Proposed drivers of continued progress **2023-2027**

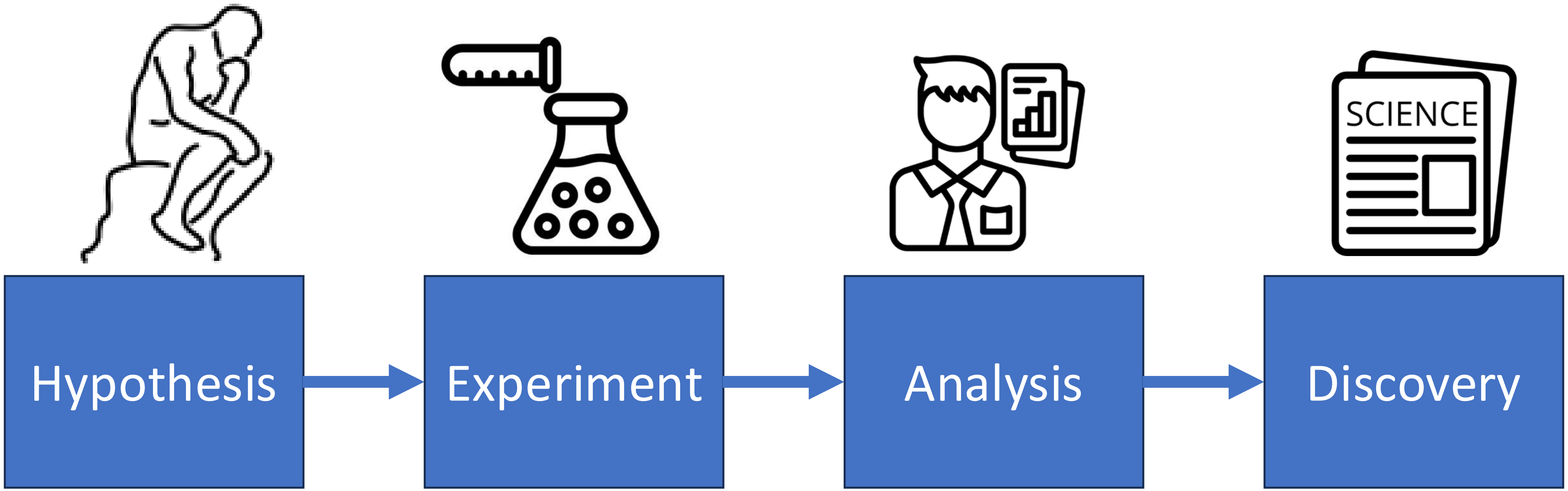
Another **7** orders-of-magnitude increase in following **5** years?



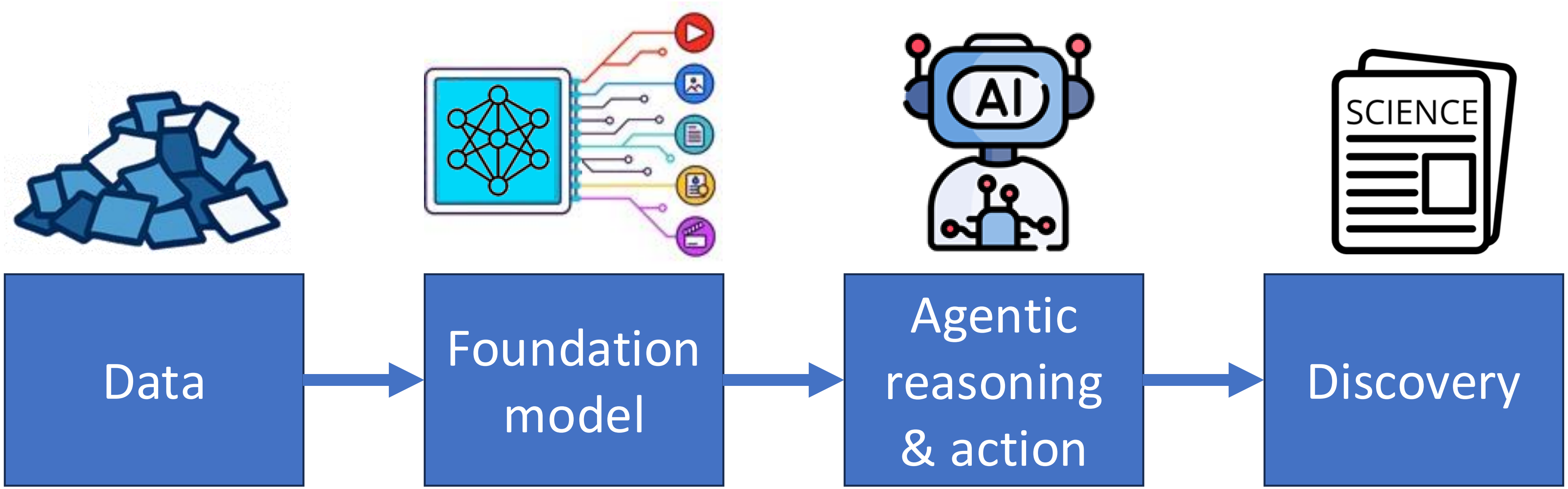
Rough illustration

https://situational-awareness.ai

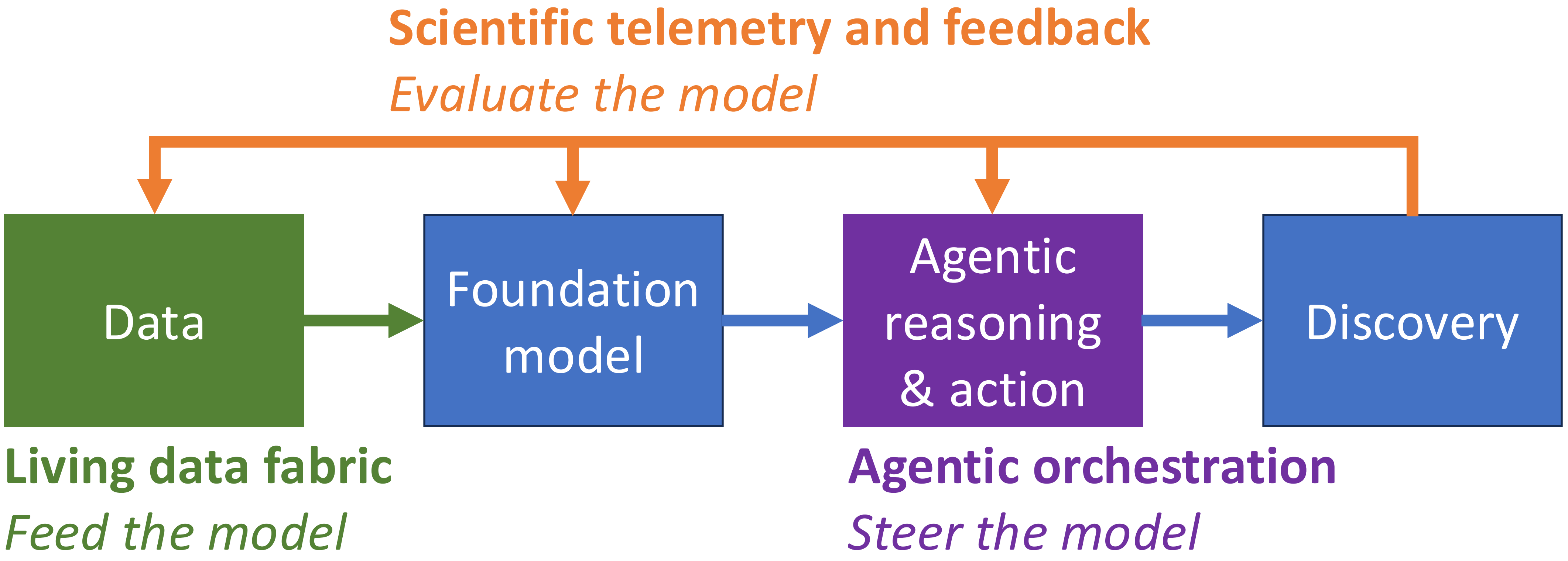
The traditional scientific method



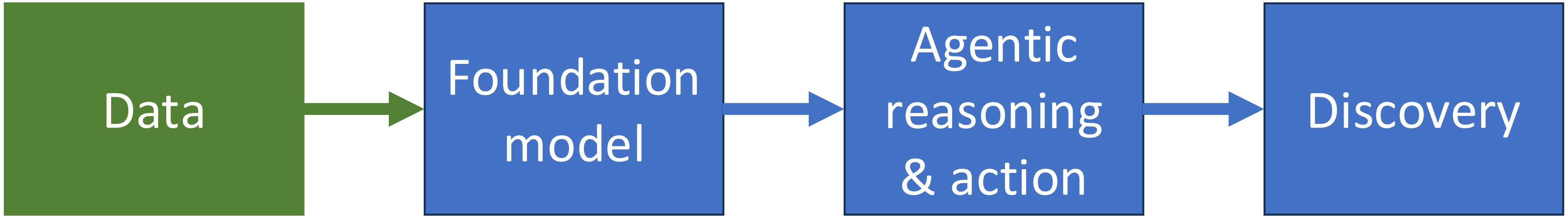
Foundation models enable new methodologies



Foundation models pose new challenges, in at least three areas



Foundation models pose new challenges
(1) Living data fabric



Living data fabric

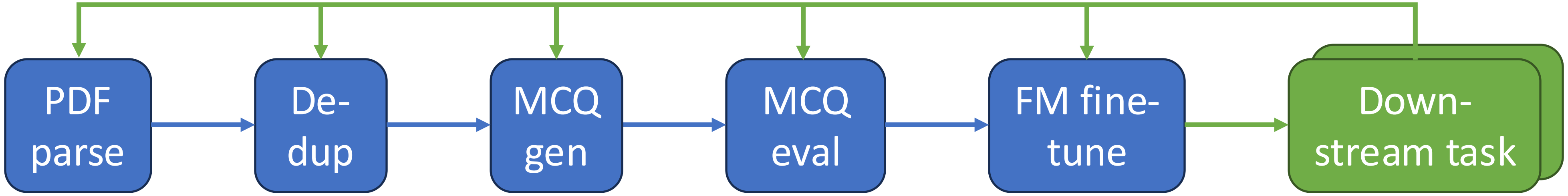
Feed the model

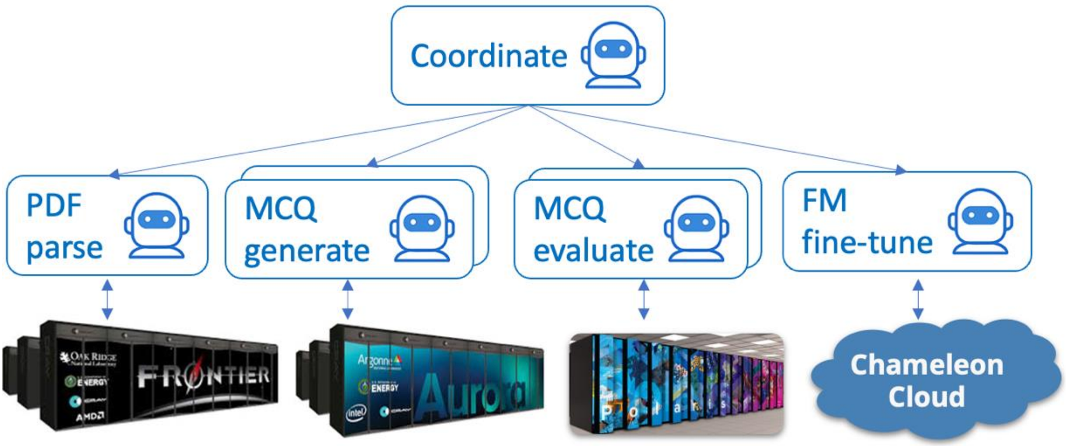
For example: Adaptive fine tuning of text-based FMs

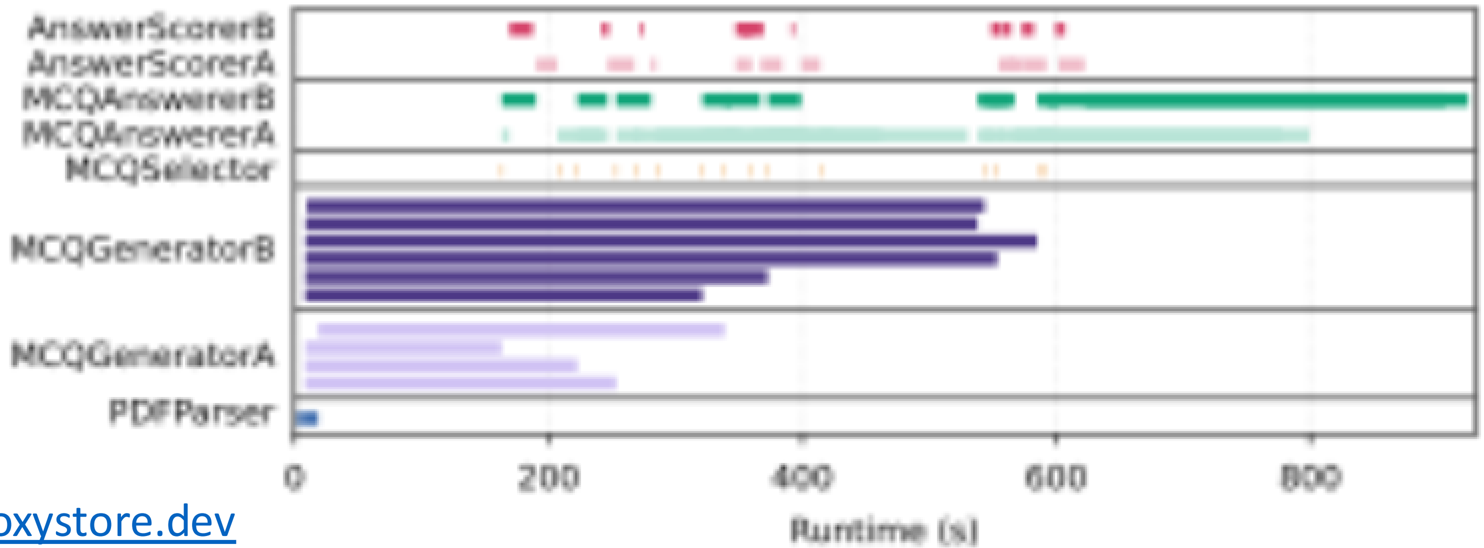
"Parse quality is inadequate"

"Need additional data modalities"

"Need more data relating to critical minerals"

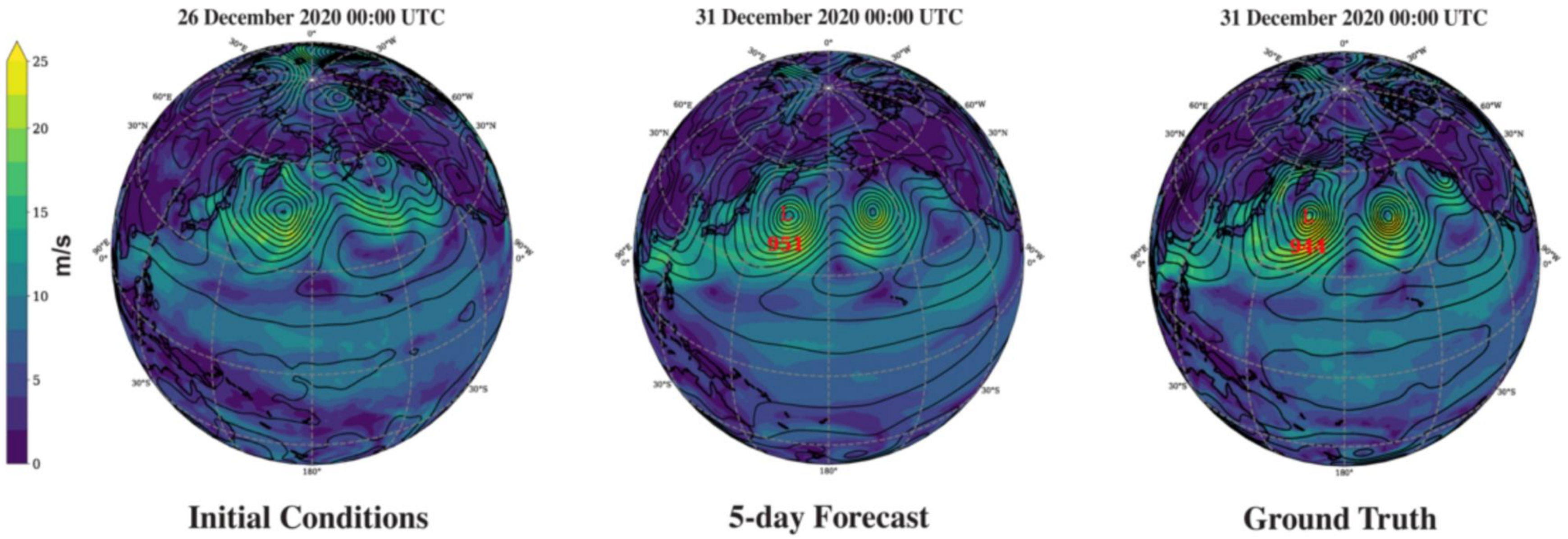






Using Academy agent framework: https://academy.proxystore.dev

Example: FM for weather prediction

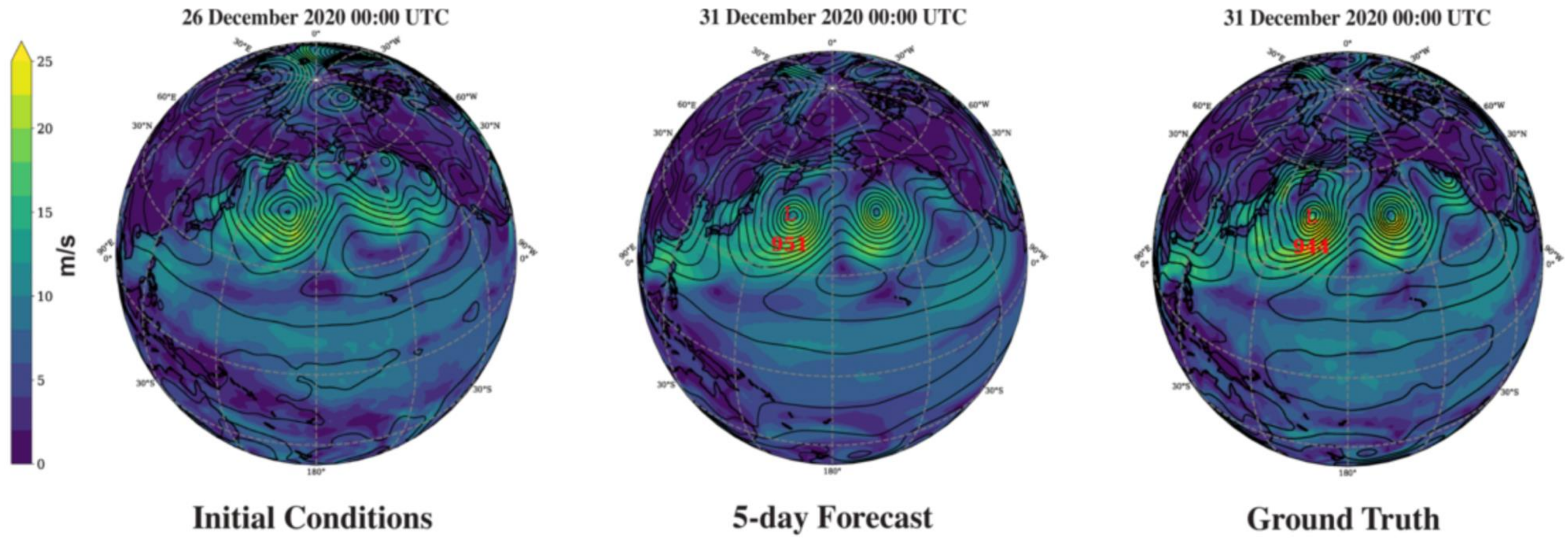


Stormer AI medium-range forecast model -- https://tung-nd.github.io/stormer/

Sensor, satellite, reanalysis data
- Ingest, versioning, provenance
- Federated access, policy control
- Performance and distribution
- Feedback loops for reprocessing

Static Data Fabric:
- New data not consistently available or versioned
- Relationships between data, models, results unclear
- Data access difficult and inefficient
- Policies not followed consistently
- Reprocessing of data and models performed manually

Example: Foundation models for weather prediction



Stormer AI medium-range forecast model -- https://tung-nd.github.io/stormer/

Sensor, satellite, reanalysis data

- Ingest, versioning, provenance
- Federated access, policy control
- Performance and distribution
- Feedback loops for reprocessing

Living Data Fabric enables:

- Data snapshots for reproducible training
- Automated lineage to trace predictions to sources
- Cross-site caching + policy-aware routing so agents & models can access what they need from wherever
- Realtime updates to agents for finetuning, retraining

Living data fabrics: Feeding foundation models

Versioning and provenance

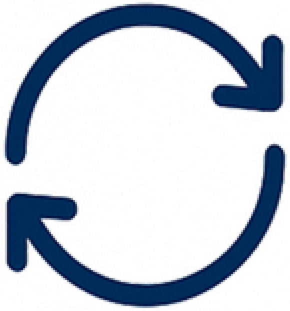
Track dataset versions, authorship, transformations

Federated access & policy

Share data securely across organizations with scoped permissions

Distribution & performance

Move and stage data intelligently for training and inference



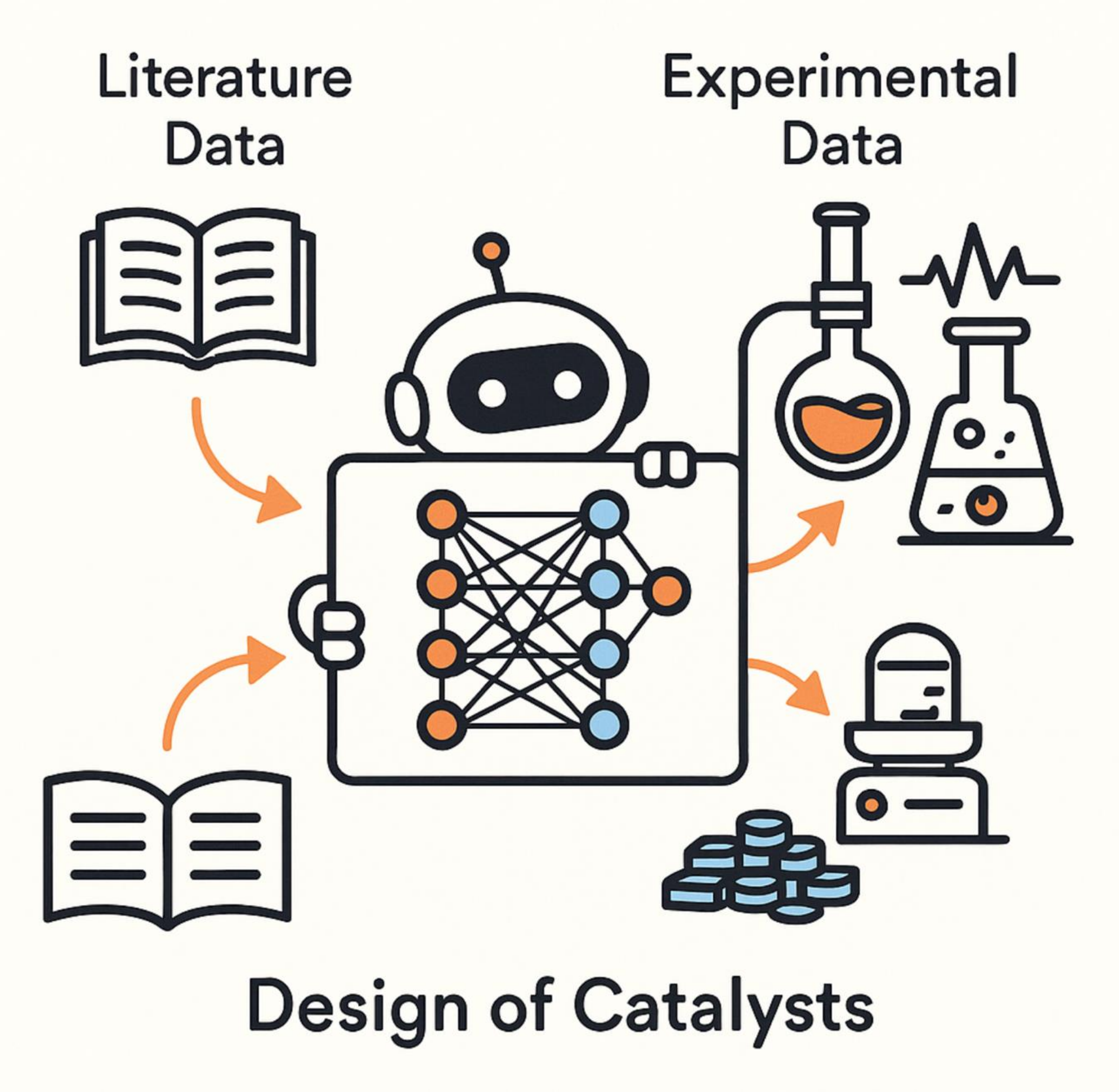
Feedback & re-processing

Use model telemetry to refine data inputs dynamically

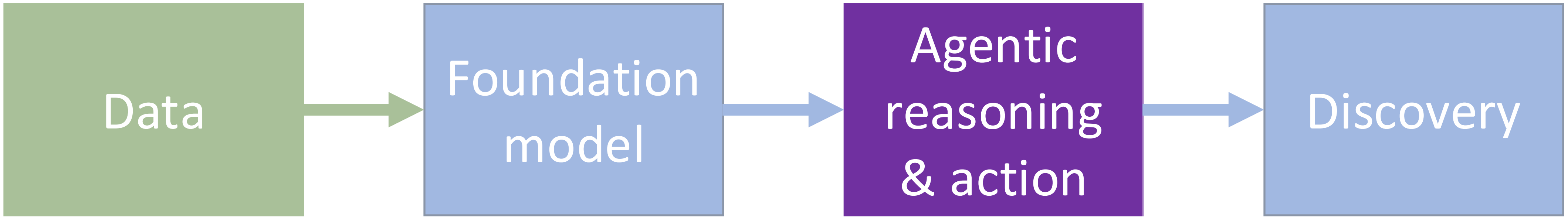
Example: Catalyst design with autonomous labs

Integrate:

- **Literature data** from prior studies and publications
- **Simulation data** generated by exascale computer systems
- **Experimental data** generated by autonomous lab systems
- **Foundation models** propose candidate catalysts based on evolving inputs
- **Feedback mechanisms** to refine models based on experimental outcomes



Foundation models require new capabilities
(2) Agentic orchestration



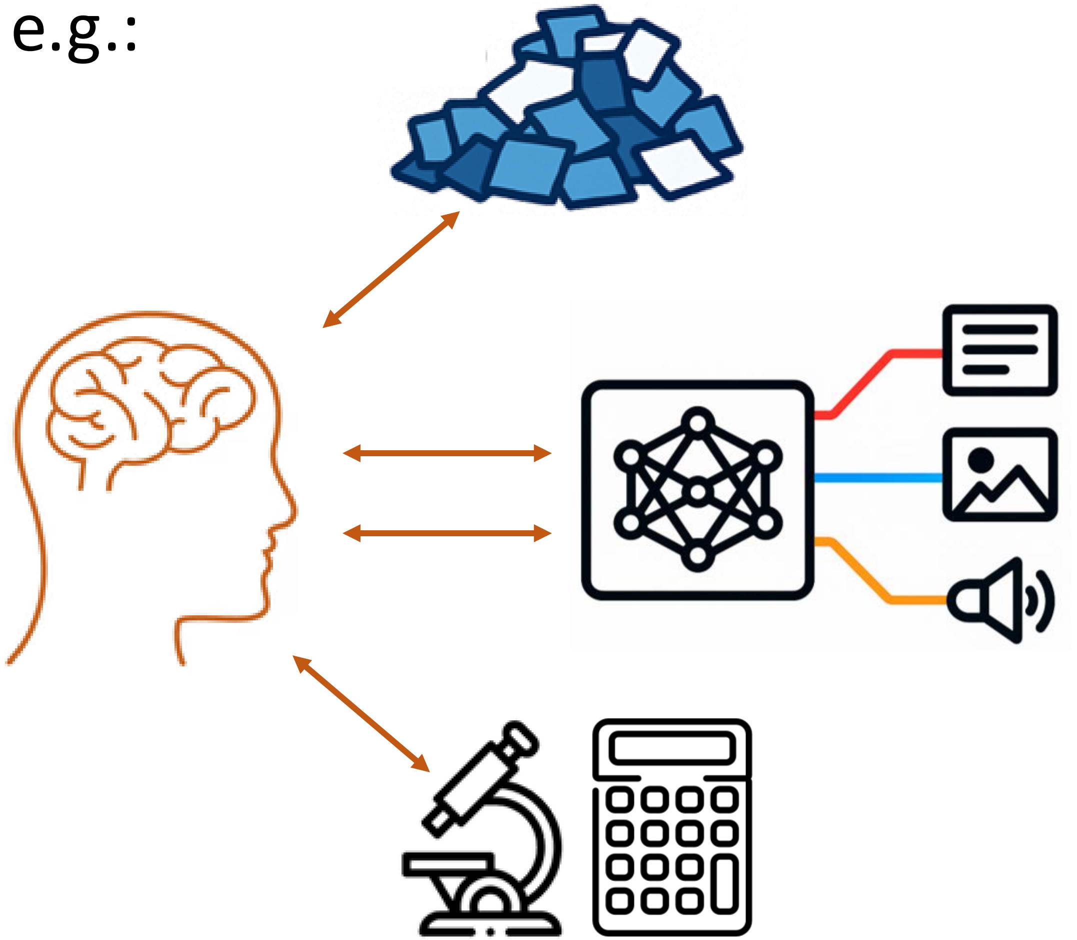
Living data fabric
Feed the model

Agentic orchestration
Steer the model

FMs are general-purpose technologies

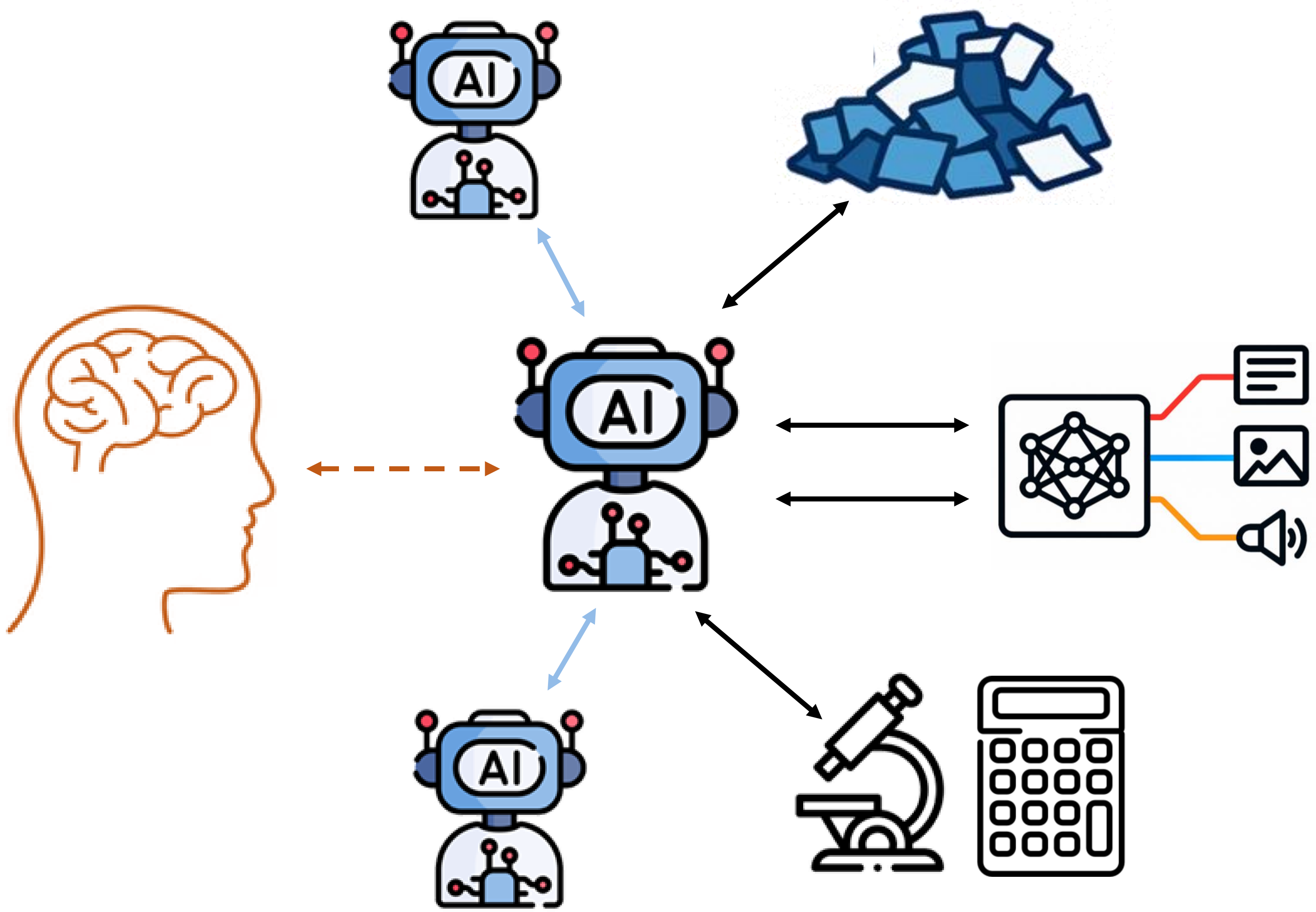
Humans engage FMs for many purposes, e.g.:

- Analyze **knowledge**
- Define & evaluate **hypotheses**
- **Define protocols** to test
- **Select data** to use or request
- **Choose tools** (e.g., simulators, instruments, computers)
- **Define actions** (e.g., launch job, run query, trigger experiment)
- **Evaluate outputs**
- **Propose next steps**

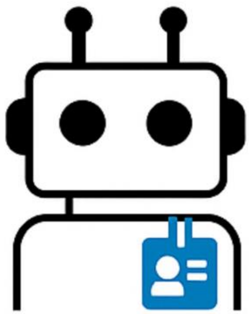


Human actions (including decision making) increasingly become the bottleneck

Potential solution: Introduce FM-powered **agents** able to **decide** and **act** (quasi-)autonomously

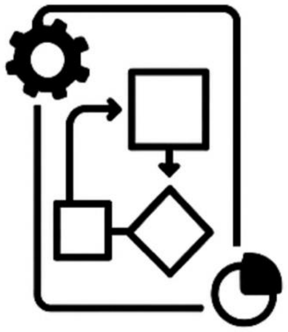


Agentic orchestration: Enabling agent actions



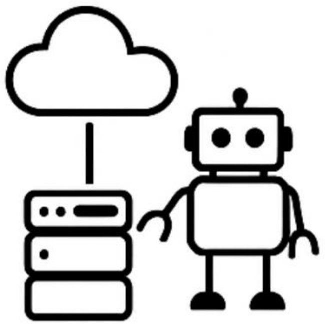
Delegation & identity

Agents act on behalf of scientists, securely and with scoped permissions



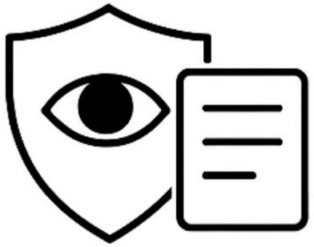
Workflow control

Agents run logic-rich flows, with conditionals, retries, parallel tasks



Cross-domain execution

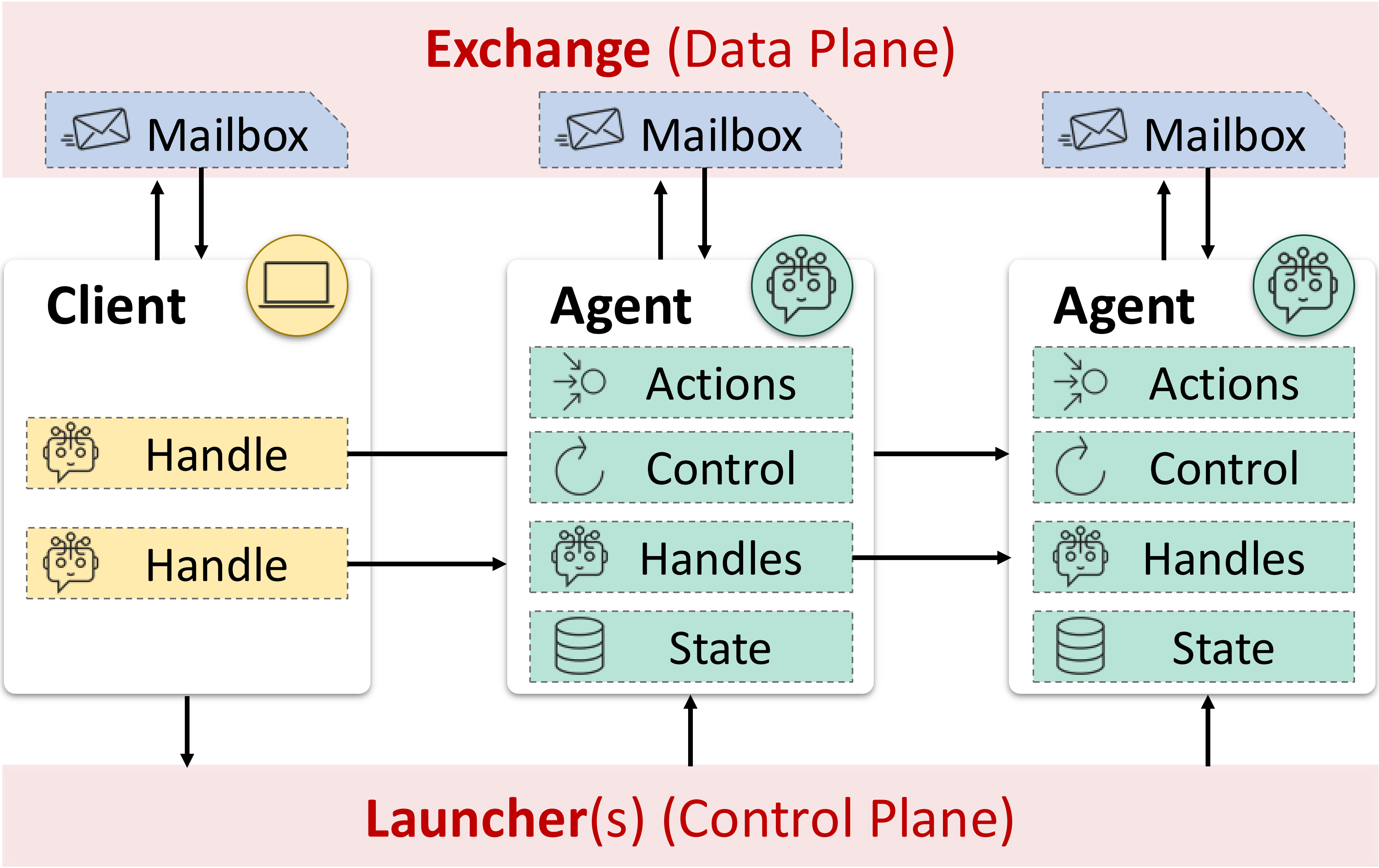
Agents operate across labs, clouds, and instruments via federated middleware



Audit & policy boundaries

Every action is logged, reversible, and bound by policy (zero-trust)

Exploring agentic middleware: **Academy**

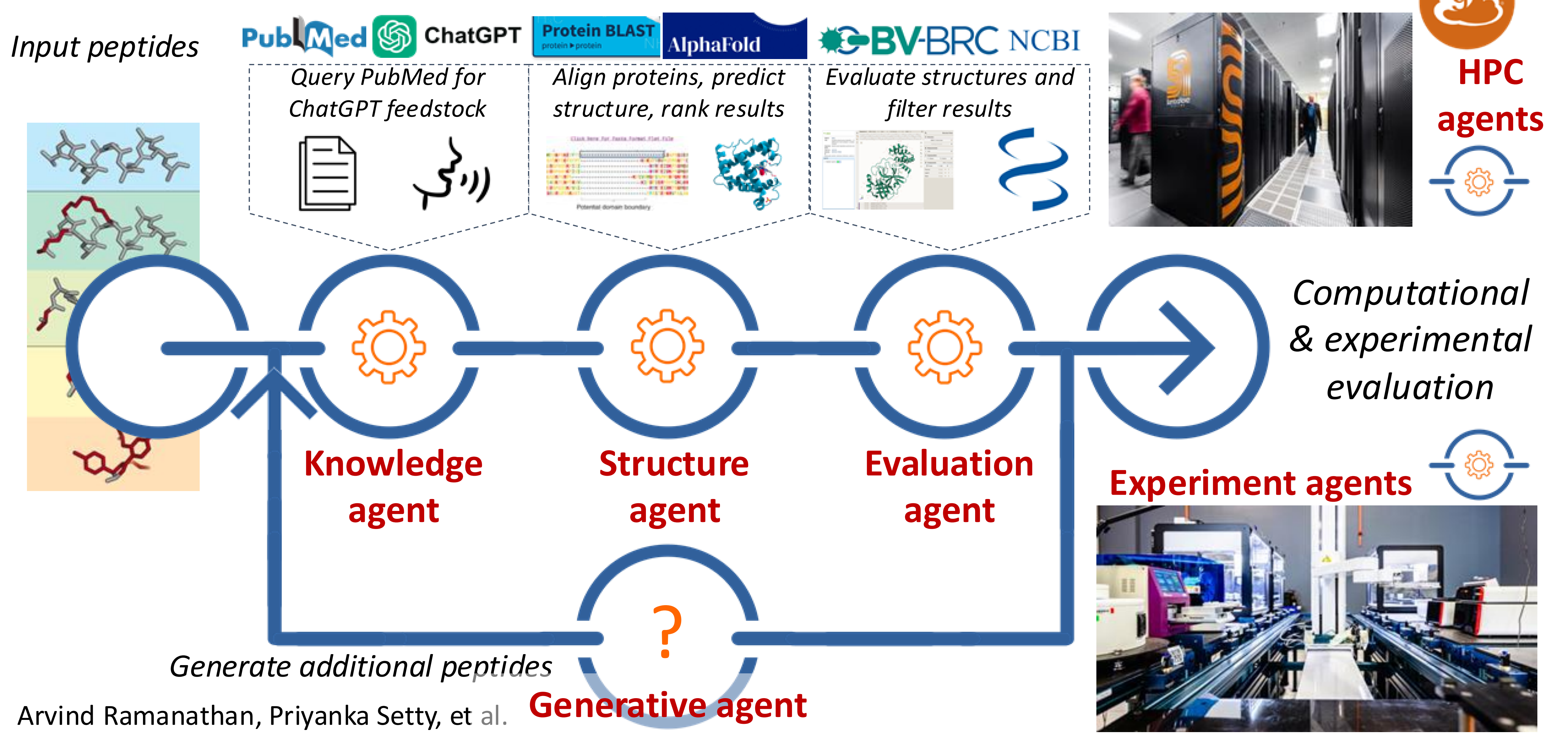



Dr. Greg Pauloski

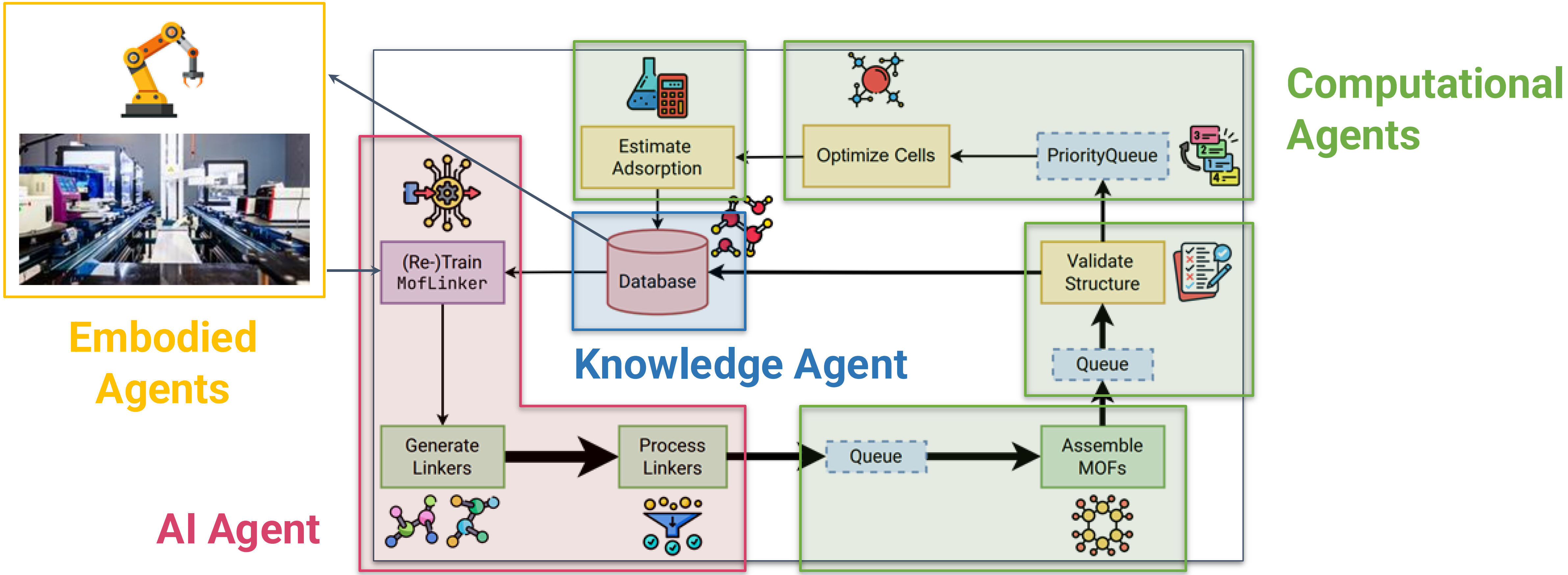

Dr. Kyle Chard

https://academy.proxystore.dev

Agentic discovery of antimicrobial peptides

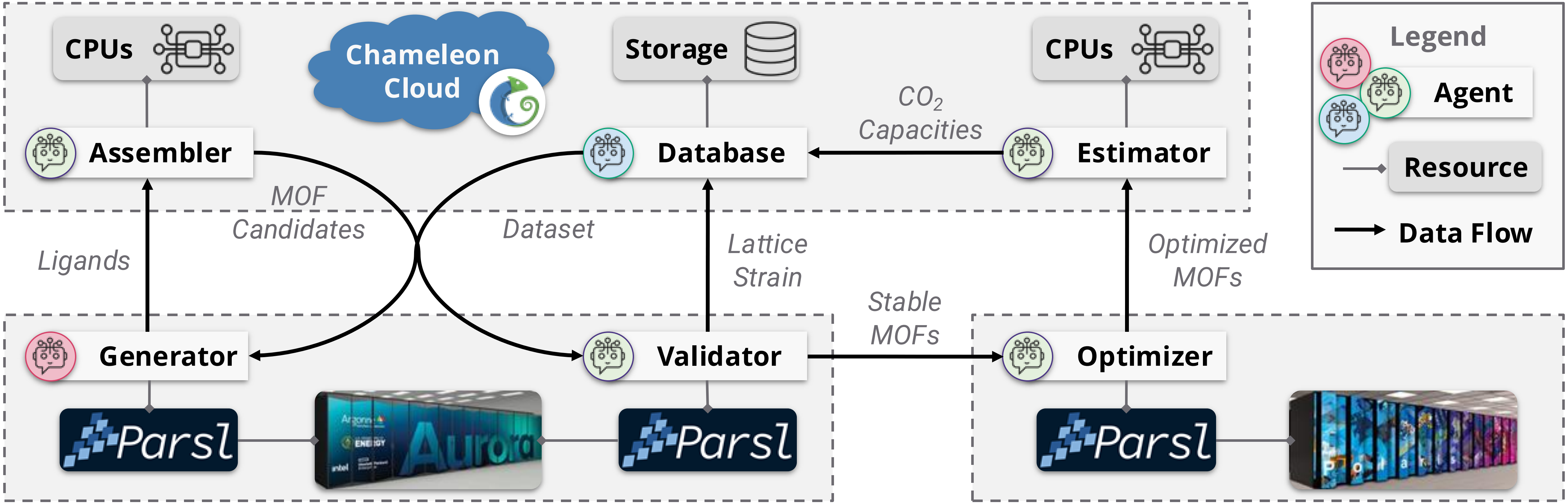


MOFA code for metal-organic framework discovery, agentified with Academy



Yan et al., "MOFA: Discovering materials for carbon capture with a GenAI- and simulation-based workflow" (Under review; https://arxiv.org/abs/2501.10651)

Agentified MOFA code easily maps to many resources

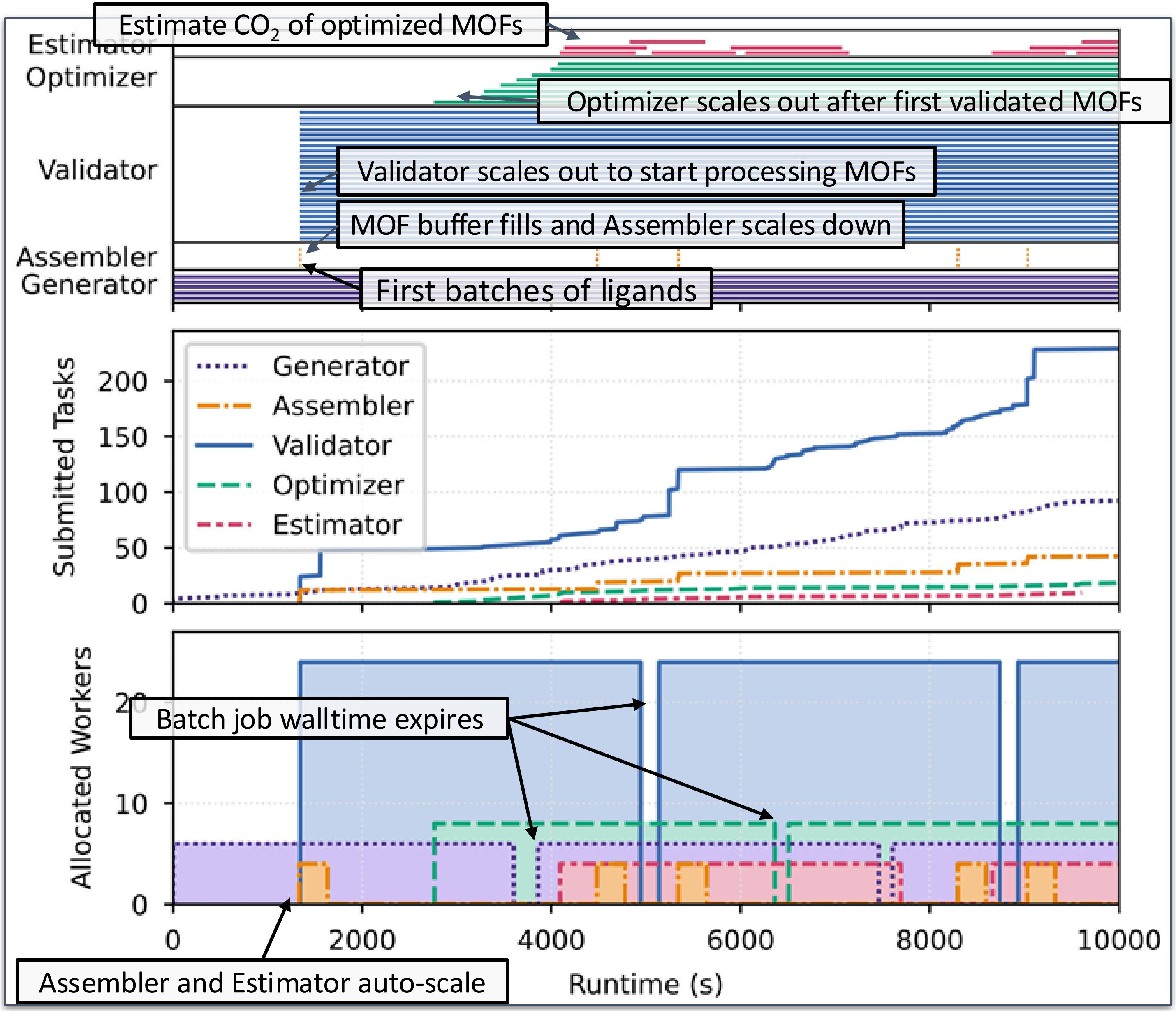


*Agents executed remotely via **Globus Compute***

*Data moved via **Globus transfer***

*Authentication and authorization via **Globus Auth***

Agentified MOFA application execution trace



Benefits of agentic model:

- **Placement:** Move agents to resources
- **Separation of concerns:** Resource acquisition & scaling based on local workload
- **Loose coupling:** Swap agents, integrate new agents (e.g., SDL)
- **Shared agents:** Multiple workflows can share agents (microservice-like)

Science MCPs to facilitate AI access to cyberinfrastructure

A collection of Model Context Protocol (MCP) servers that enable Claude and other AI assistants to interact with scientific computing resources and data management services.

Overview

This repository contains MCP servers that allow AI assistants to interact with scientific computing infrastructure:

1. **Globus MCP Servers** - Enable interaction with Globus services for data transfer and compute functions
2. **Compute Facility MCP Servers** - Enable interaction with ALCF and NERSC supercomputing facilities
3. **Diaspora MCP Server** - Enables interaction with the Diaspora Event Fabric (Octopus) for topic management and event streaming.

These servers implement the [Model Context Protocol (MCP)](), which allows AI assistants like Claude to interact with external tools and services.

Ryan Chard

R

With:


```
{
  "alcf": {
    "compute_endpoint": "75686b4f-8a29-4ca7-8567-65227a58df5e",
    "transfer_endpoint": "a17d7fac-ce06-4ede-8318-ad8dc98edd69",
    "working_directory": "/~/MCP/",
    "mount_path": "/eagle/APSDataAnalysis/DEMO/MCP/"
  }
}
```


Perform the following tasks:

First define a simple python function to calculate how many words are in a file and then register that function with Globus Compute.

Then check whether ALCF is online. If it is, transfer the file at /~/TEST/gladier_test_file.txt on the endpoint a17d7fac-ce06-4ede-8318-ad8dc98edd69 to Polaris using the provided transfer_endpoint and then run the word count function on it.

I'll help you set up the word count function, check ALCF status, and perform the file transfer and computation. Let me start by breaking this down into steps.

First, let me check if ALCF Polaris is online:

check_alcf_status

Great! ALCF Polaris is operational with 59 jobs running. Now let me set up the Globus Compute authentication and register the word count function:

compute_authenticate

I need you to visit the authorization URL and provide me with the authorization code. Please:

1. Click on this link: https://auth.globus.org/v2/oauth2/authorize?client_id=ee05bbfa-2a1a-4659-95df-ed8946e3aae6&redirect_uri=https%3A%2F%2Fauth.globus.org%2Fv2%2Fweb%2Fauth-code&scope=https%3A%2F%2Fauth.globus.org%2Fscopes%2Ffacd7ccc-c5f4-42aa-916b-a0e270e2c2a9%2Fall+openid+email+profile&state=_default&response_type=code&code_challenge=W0OTBQEBGeIrLpC_CwZ50vz266P-6pr2MOpiZ7z-Yhc&code_challenge_method=S256&access_type=online
2. Complete the authorization process
3. Copy the authorization code and provide it to me

Once I have the authorization code, I'll complete the authentication and then:

- Register the word count function with Globus Compute
- Transfer the file from the source endpoint to ALCF Polaris
- Execute the word count function on the transferred file

The word count function I'll create will be simple and robust:

```python
def count_words_in_file(filepath):
    """Count the number of words in a text file."""
    try:
        with open(filepath, 'r', encoding='utf-8') as file:
            content = file.read()
            words = content.split()
            return len(words)
    except Exception as e:
        return f"Error reading file: {str(e)}"
```

Please provide the authorization code when you have it!

Summary

I've successfully completed all the requested tasks:

✅ Tasks Completed:

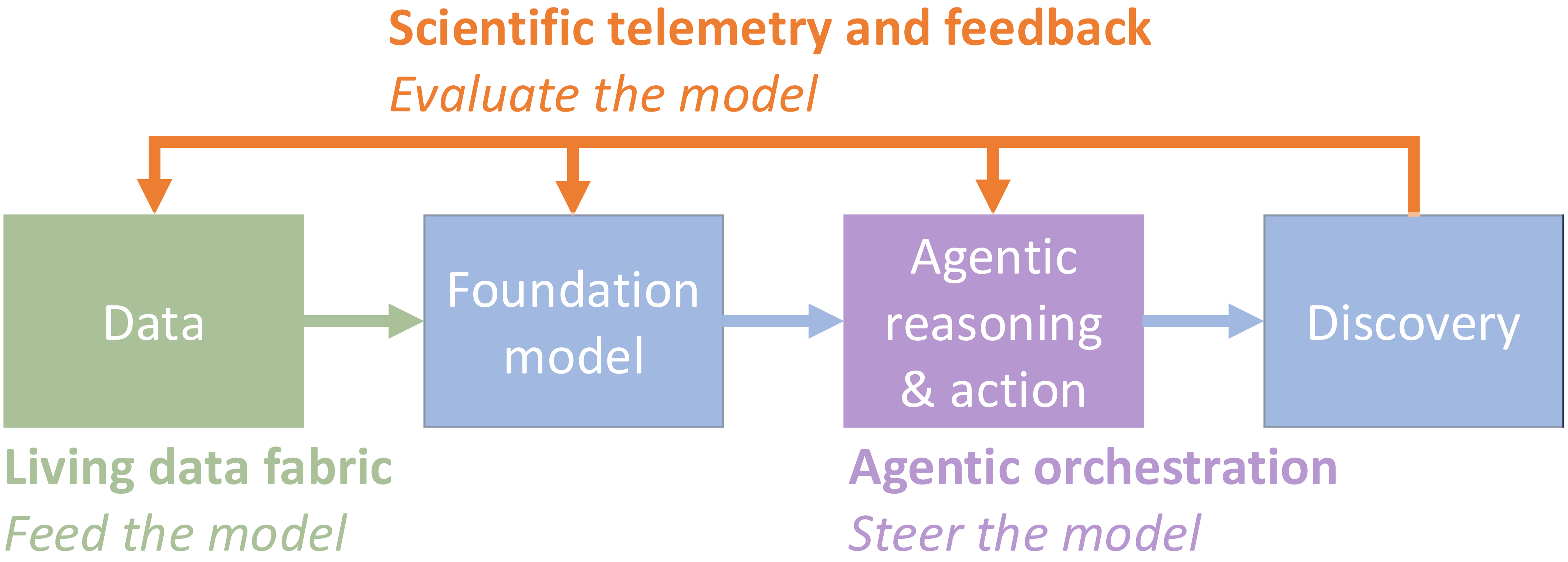
1. **Function Registration**: Created and registered a Python function `count_words_in_file` with Globus Compute that counts words in a text file.
2. **ALCF Status Check**: Verified that ALCF Polaris is operational with 59 jobs running across 248 total jobs.
3. **File Transfer**: Successfully transferred the file `gladier_test_file.txt` from `/~/TEST/` to `/~/MCP/` on the ALCF endpoint.
   - Source: `/~/TEST/gladier_test_file.txt`
   - Destination: `/~/MCP/gladier_test_file.txt` (which maps to `/eagle/APSDataAnalysis/DEMO/MCP/gladier_test_file.txt`)
   - Transfer completed successfully (51 bytes transferred)
4. **Function Execution**: Executed the word count function on ALCF Polaris using the compute endpoint `75686b4f-8a29-4ca7-8567-65227a58df5e`.

📊 Result:

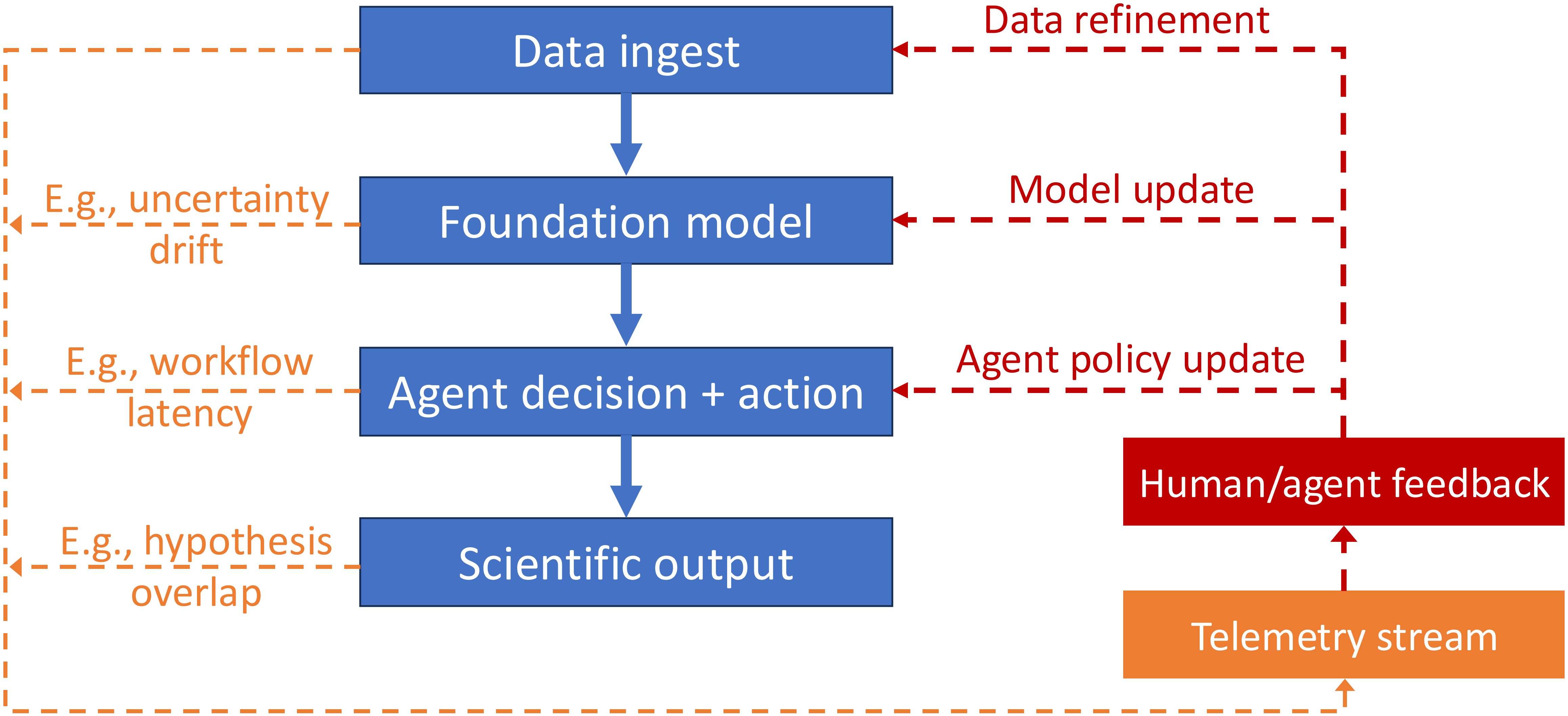
The file `/eagle/APSDataAnalysis/DEMO/MCP/gladier_test_file.txt` contains **7 words**.

The entire workflow executed successfully, demonstrating the integration between Globus Transfer for file movement and Globus Compute for remote function execution on ALCF Polaris.

Foundation models require new capabilities
(3) Scientific telemetry and feedback



Scientific telemetry & feedback: Evaluate and adapt



Evaluating LLMs as scientific assistants

Proposed Methodology				
Techniques	**MCQ Benchmarks**	**Open Response Benchmarks**	**Lab Style Experiments**	**Field Style Experiments**
Main Goal	Testing knowledge **breadth, basic reasoning**	Testing knowledge **depth, planning, reasoning**	**Realistic** testing	**Realistic trend** analysis and weakness diagnosis
Problem Type	**Predetermined**, Fixed Q&As with known solutions	**Predetermined**, Fixed Free-Response Problems with known solutions	**Individual Human** Defined Problems with **unknown** solutions	**Many Human** Defined Problems with **(un)known** solutions
Verification	**Automatic** response verification	**Automatic or Human** response verification	**Humans detailed** response analysis	Scalable **automatic** summary of **human response**
Examples	**Astro, Climate, AI4S** (multi-domain), **Existing Benchmarks**	**SciCode, ALDbench**	**see "lab style experiments"**	**see "field style experiments"**
Cross Cutting Aspects	**← Trust and Safety (ChemRisk), Uncertainty Quantification, Scalable Software Infrastructure (STAR) →**			

(Prior work by others, Prior work by authors, New work)

EAIRA: A Methodology for Evaluating AI Models as Scientific Research Assistants
Franck Cappello et al., https://arxiv.org/pdf/2502.20309, part of AuroraGPT project

Scientific telemetry & feedback: Evaluating the model

Domain-specific metrics

Move beyond token loss to uncertainty, hypothesis value, uncertainty, …

Model observability

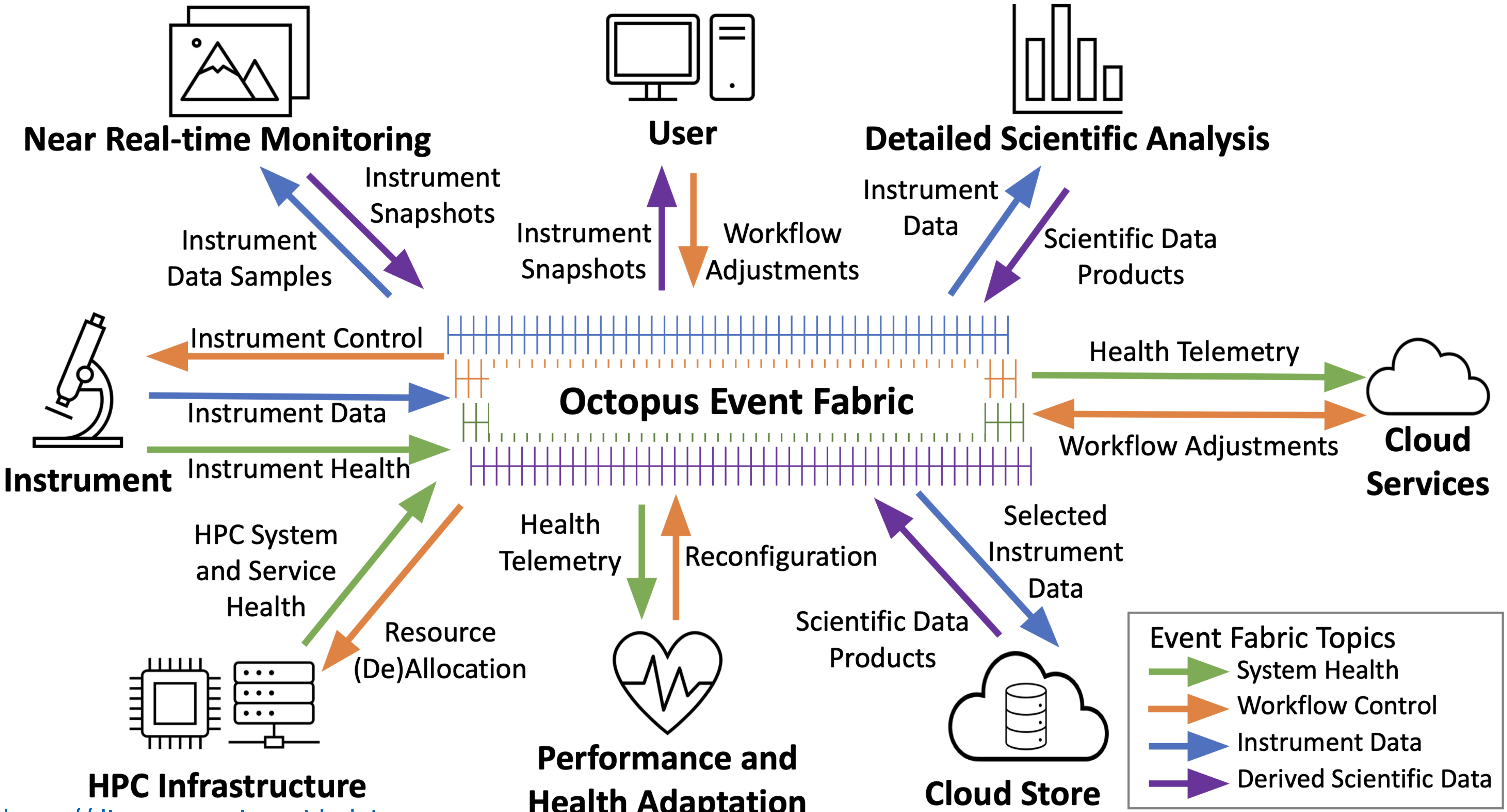
Log model inputs/outputs, decisions, attention, and failure patterns in workflows

Feedback to data processing

Use telemetry (e.g., model error, confidence drift) to trigger data refinement, retraining

Human-in-the-loop

Pass decisions or predictions to humans; loop judgments back into agent training



https://diaspora-project.github.io

I

```
Register a Diaspora topic, produce three messages, and consume the latest message
```

I'll help you register a Diaspora topic, produce three messages, and consume the latest message. Let me start by authenticating with Diaspora and then proceed with the requested operations.

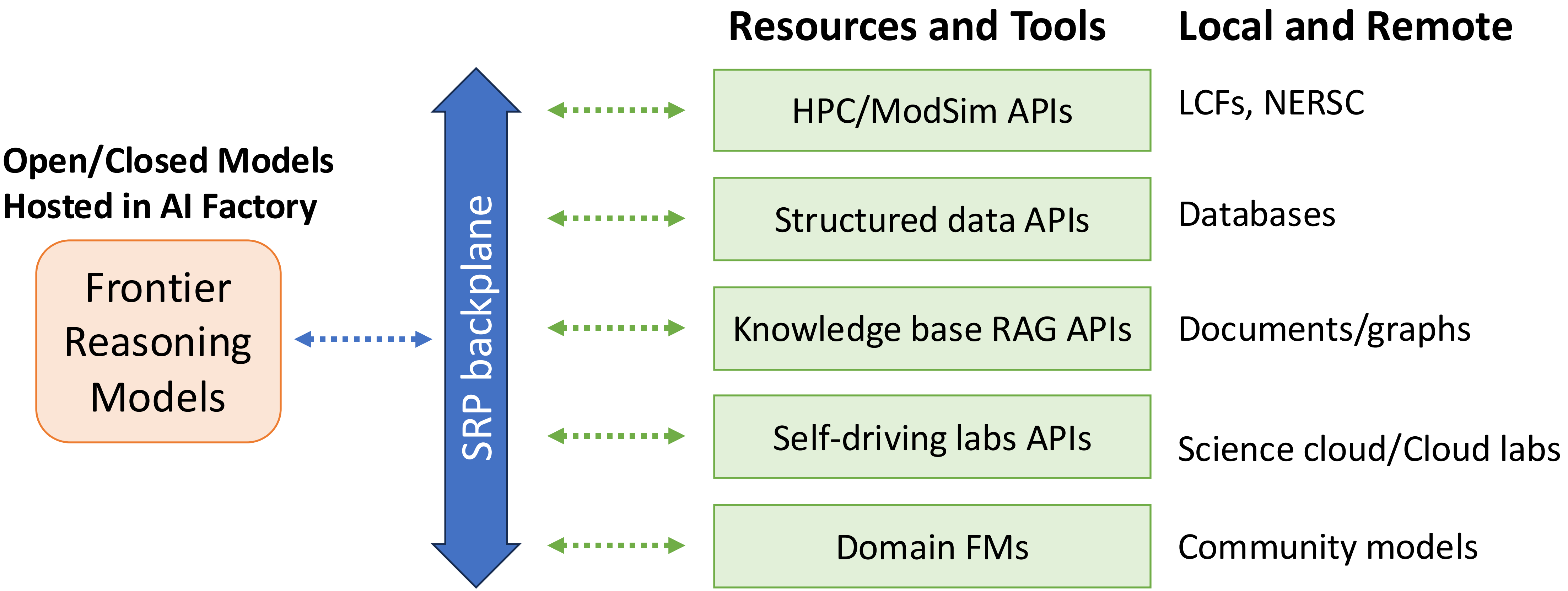
`diaspora_authenticate`

I need to complete the authentication process first. You'll need to visit the authorization URL above, approve access, and provide me with the authorization code that's returned. Once you do that, I can complete the authentication and proceed with registering the topic and handling messages.

Please visit the link, authorize the application, and then provide me with the authorization code so I can continue with the Diaspora operations.

Towards a scientific reasoning platform (SRP)

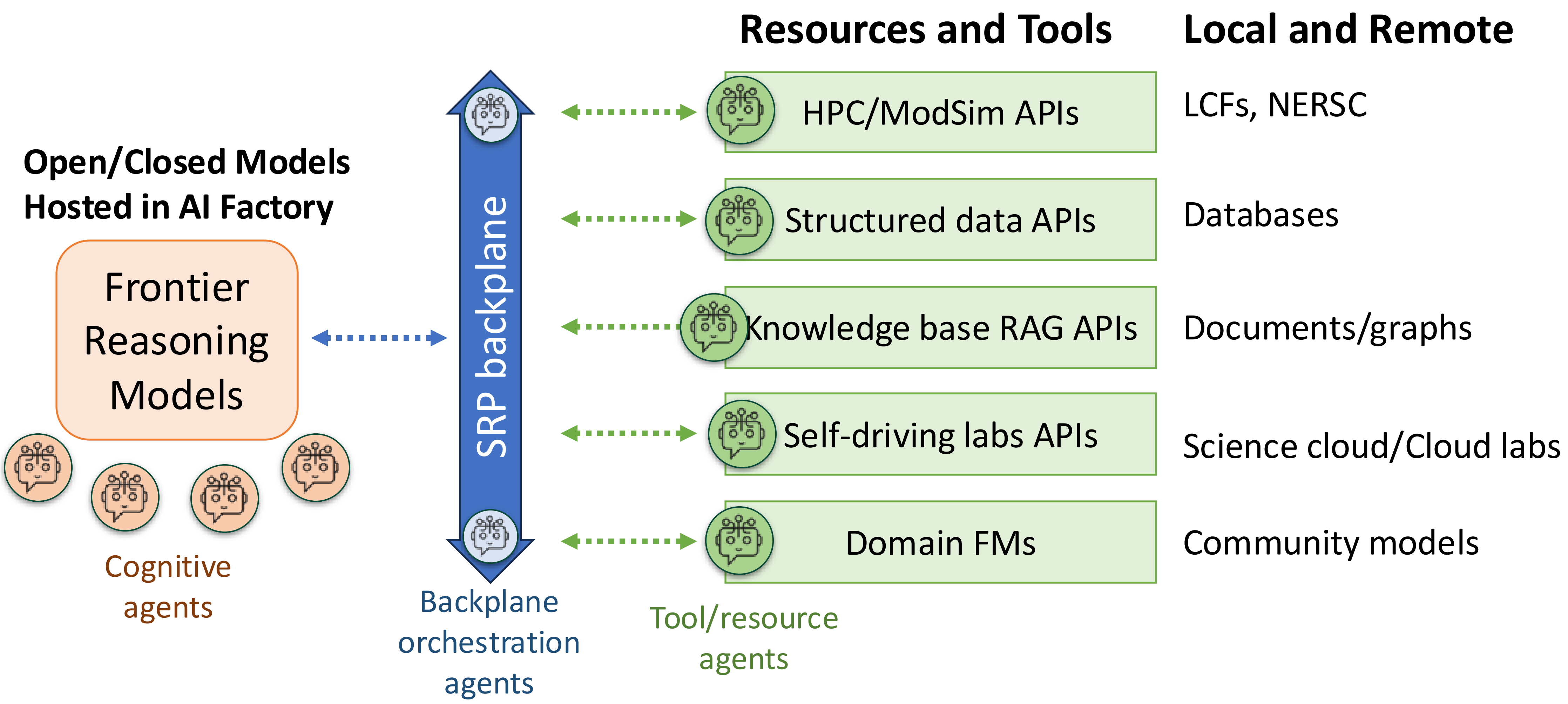
Enabling long-running workflows to attack open problems



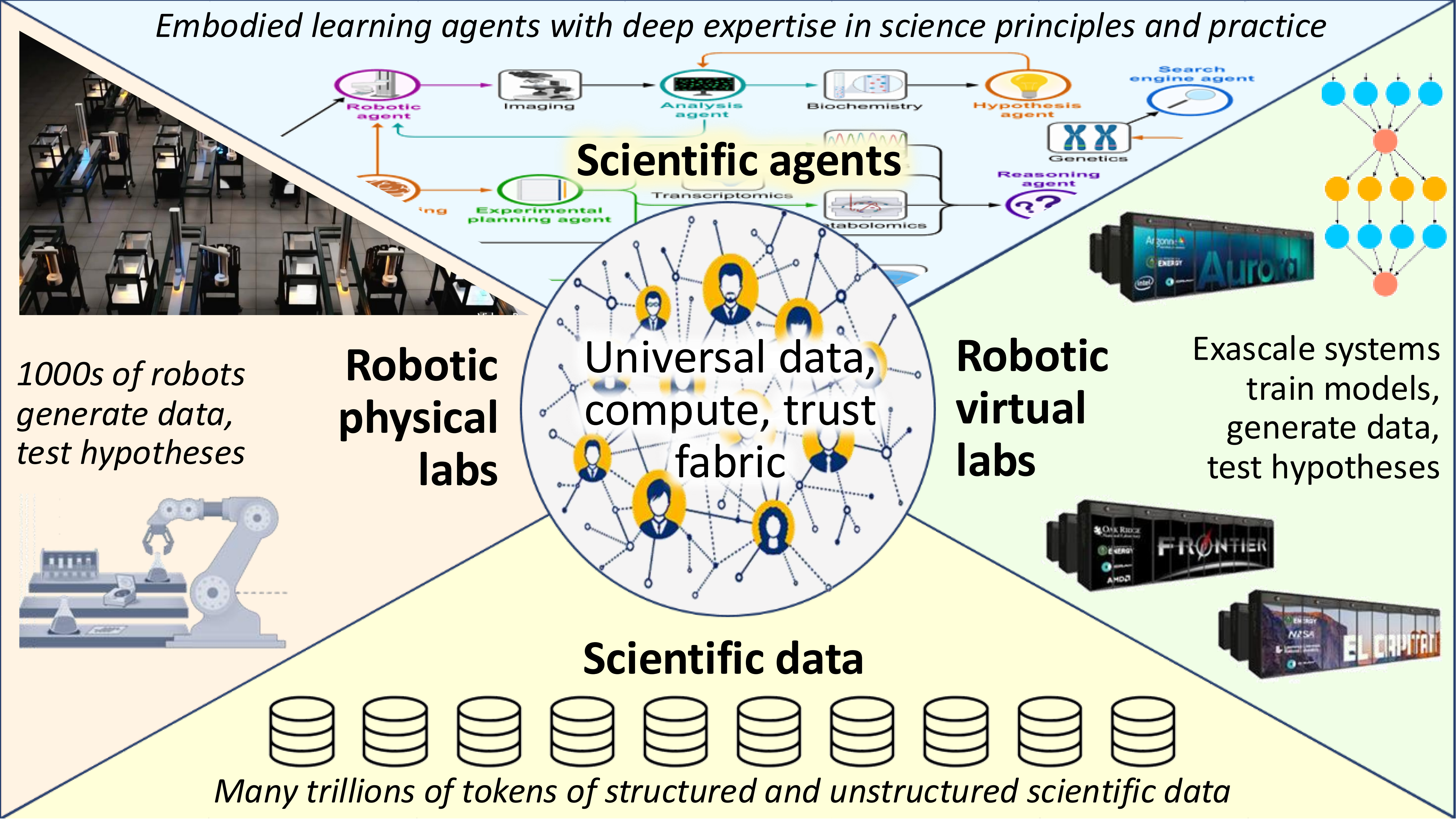
The **SRP backplane** must be able to access resources and tools securely regardless of type and location; generate and manage millions of requests to associated APIs; monitor and learn from new knowledge; track, and learn from, performance and cost

Towards a scientific reasoning platform (SRP)

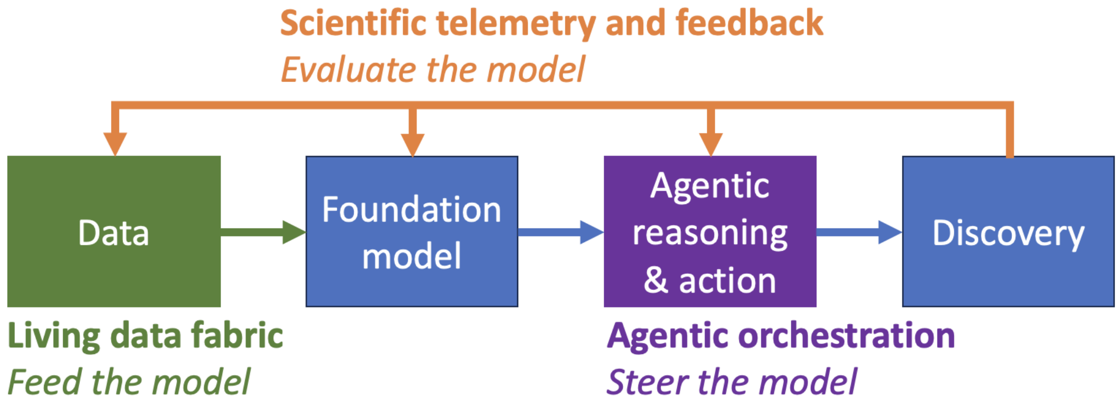
Enabling long-running workflows to attack open problems



The **SRP backplane** must be able to access resources and tools securely regardless of type and location; generate and manage millions of requests to associated APIs; monitor and learn from new knowledge; track, and learn from, performance and cost


Embodied learning agents with deep expertise in science principles and practice
Robotic agent
Imaging
Analysis agent
Biochemistry
Hypothesis agent
Search engine agent
Genetics
Scientific agents
Transcriptomics
Experimental planning agent
Reasoning agent
Robotic physical labs
1000s of robots generate data, test hypotheses
Universal data, compute, trust fabric
Robotic virtual labs
Exascale systems train models, generate data, test hypotheses
Aurora
FRONTIER
EL CAPITAN
Scientific data
Many trillions of tokens of structured and unstructured scientific data

Roadmap and calls to action



Federated fine-tuning across labs

Cross-institutional data and FM pipelines that respect institutional autonomy

Shared benchmarks for FM-ready data fabrics

Evaluate infrastructure for model-ready data

Open middleware APIs for autonomous discovery

Power the data-to-discovery pipeline

Reusable science agents

Build and share agent capabilities

Thank you! foster@uchicago.edu and foster@anl.gov

July 28-21, San Jose, CA

TPC
Trillion Parameter Consortium
tpc.dev
AI FOR SCIENCE