



Interactive Analysis Workbench

Progress and AI Exploration

Jingyan Shi

Computing Center, Institute of High Energy Physics,
Chinese Academy of Science

Shijy@ihep.ac.cn

On behalf of the INK Team

Outline

01 INK Recap

02 Progress of INK

03 AI Exploration at IHEP Computing Platform

04 Summary

INK Recap

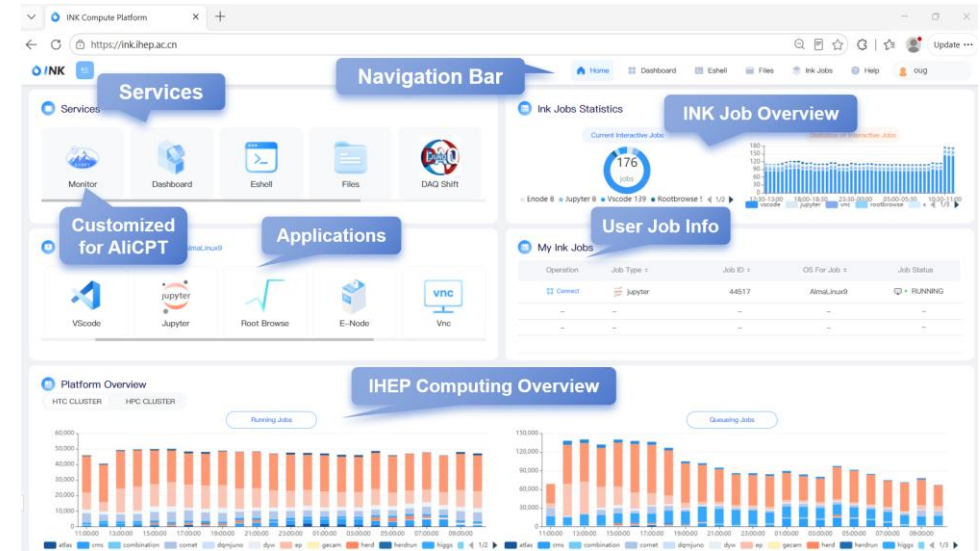
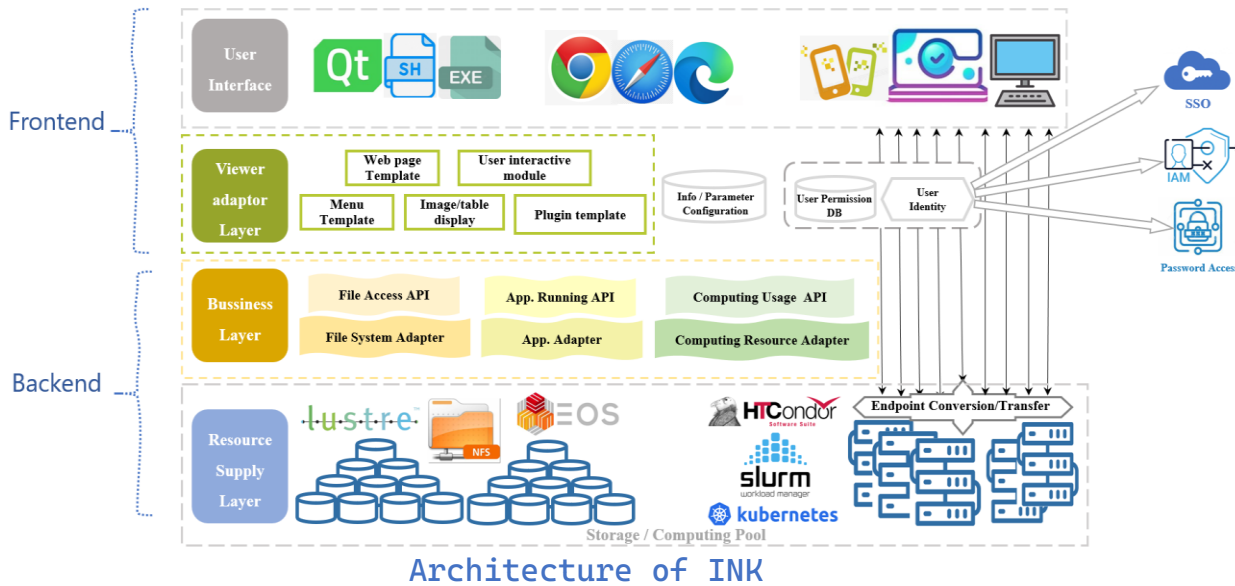
Open the computing and storage of the cluster to the world

Shift backend / offline resources to the frontend / interactive

Enable flexible integration of resources and applications

Provide customized interactive tools for HEP experiments

Offer plus-and-play deployment solution for the HEP computing sites



Visionary Goal: Scientific research with just a network connection

INK Status

From prototype to production

INK has evolved through 22 releases over a year of deployment, growing from an internal tool into an open-source platform embraced by a widening user community.

Continually Developing

22 major and minor versions shipped, each bringing new capabilities, optimizations, and stability refinements

Production @ IHEP

Running in production at for more than one year with growing features and consistently positive user feedback

Open Source <https://github.com/INK-IHEP>

Fully open-sourced – welcoming community contributions and transparent development

Multi Site Deployment

Deployed at Lanzhou University, validating the platform beyond its origin institution

Outline

01 INK Recap

02 Progress of INK

03 AI Exploration at IHEP Computing Platform

04 Summary

INK Progress -- API & Workflow Integration

Rich INK Backend API

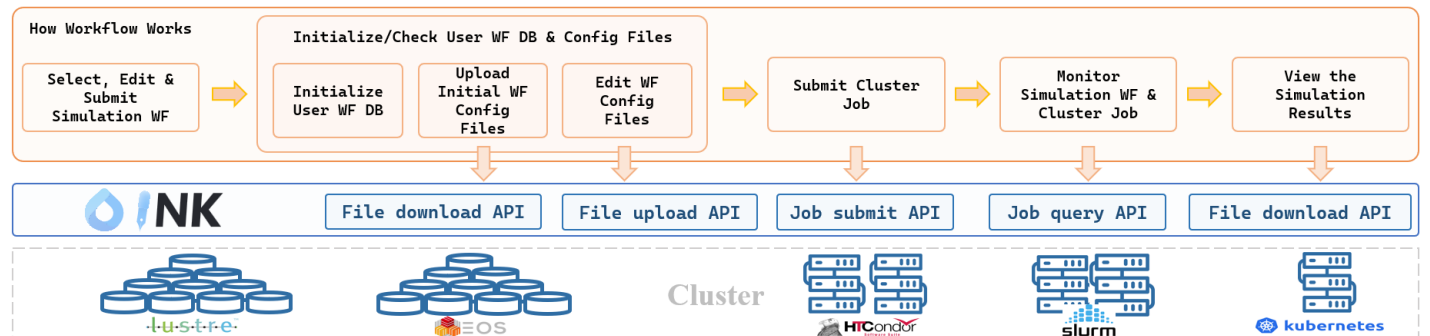
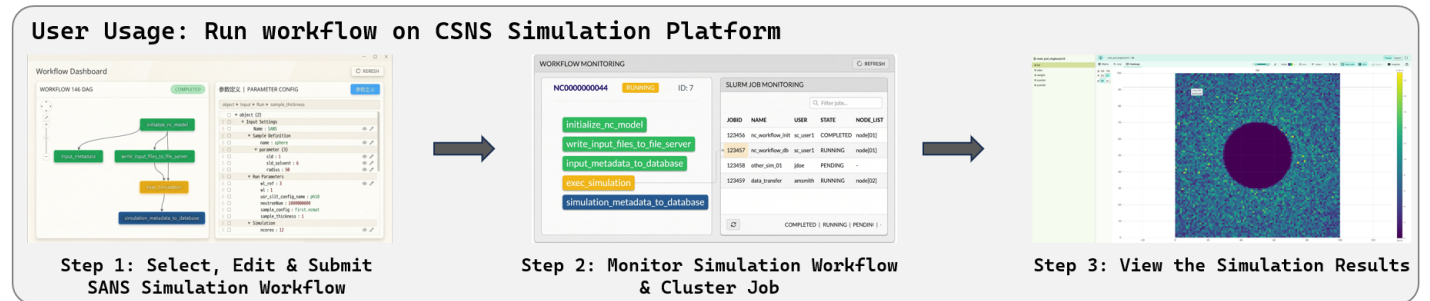
Provide HEP experiment data processing workflows with direct access to cluster resources. Workflows run without installing or configuring cluster software on the user side.

Token based Zero-Config Workflow

Users carry a token and directly run workflows, submit jobs, and access cluster data files – all from their own experiment environment. No SSH, no Kerberos setup, no batch system

Real Example: INK API Support China Spallation Neutron Source(CSNS) Simulation Platform

The platform is positioned to support the simulations of experimental data analysis at CSNS. Its workflow call INK API to runs the simulation task at the cluster.



INK Progress -- Plugin Architecture

Standalone Python Packages

Plugins are standalone packages. Write, install, enable, and restart - no core code modification needed.

1. @hookable Override

Decorate any function with @hookable; override at runtime via register_hook() -all pluggable

2. Automatic Router Mounting

Return APIRouter from get_routers() - INK auto-discovers and mounts at startup

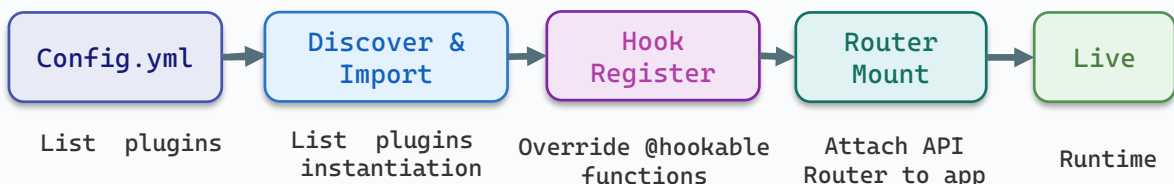
3. Config-Driven Lifecycle

Add to config.yml unified_plugins packages, restart server - loads without touching core code

4. Template Plugin

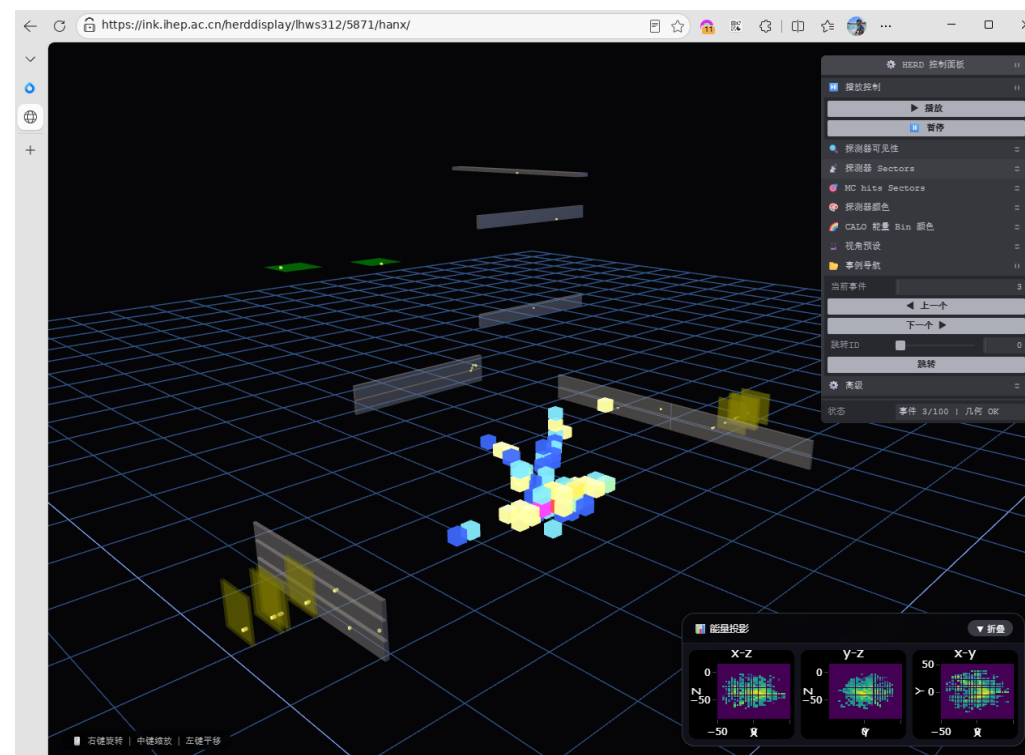
fastink-plugins/template_plugin/ - scaffold with pyproject.toml, tests, full working example

Plugin Loading Flow



First Production: Contributed by HERD physicist

HERD experiment developed their event display web interface as a plugin, accessible directly through INK.



A screenshot of the HERD event display web interface accessed through the INK.

INK Progress -- One-Click Deployment

From Bare Linux Host to Production-Ready HEP Platform

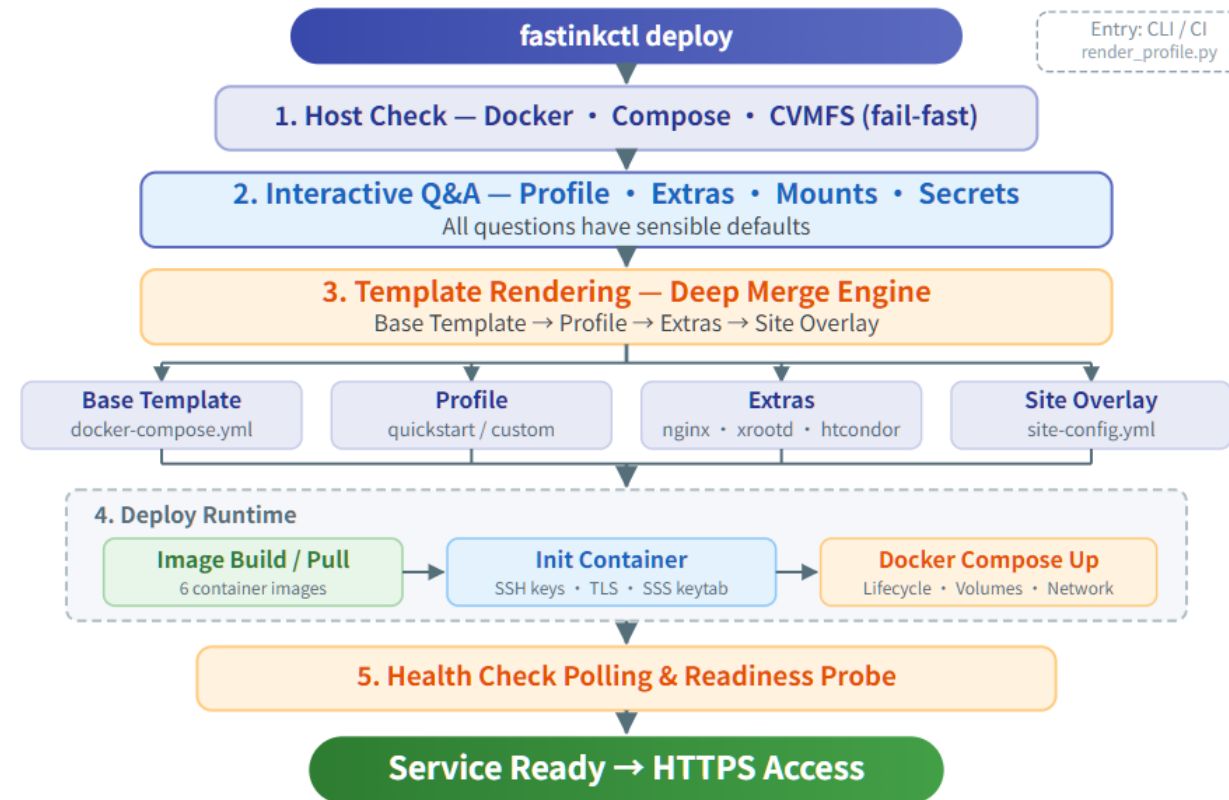
Zero to production with One Command in minutes

Why It Matters

- * Container-native – every service ships as a Docker image. No Kubernetes, no host-level package installs
- * Init container auto – provisions SSH keys, TLS certs, SSS keytabs – zero manual credential setup
- * Deep-merge engine layers Base – Profile → Extras → Site Overlay for zero-config defaults with customization
- * CI/CD-ready – non-interactive mode via `render_profile.py` enables Git-driven auto-deployment

- * Small HEP groups with zero ops background manage 2–3 servers the “INK way” in minutes
- * Ideal for HEP course training – deploy a complete HEP environment with one command

DEPLOYMENT PIPELINE



INK Progress -- Job Lens

Independently developed tool:

Lightweight Observability Collector for Cluster Jobs

From Log Archaeology to Real-time Insight

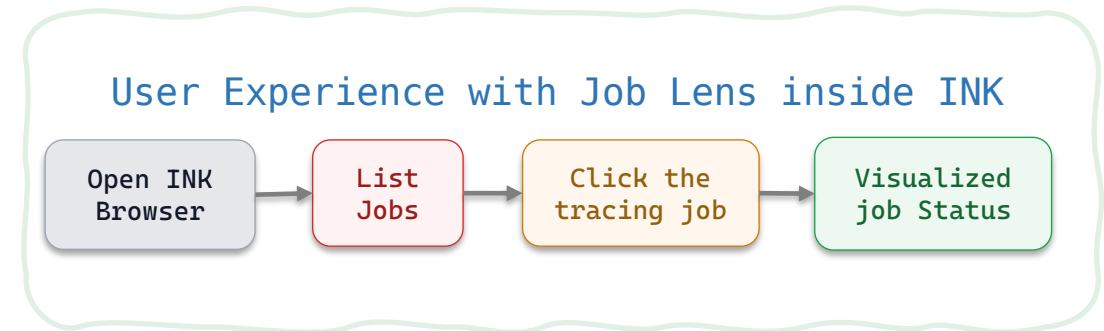
- * Traditional cluster debugging relies on scraping logs after failures.
- * Job Lens closes the observability gap by providing real-time, per-job metrics at the workload level – not the node level.

Zero-Intrusion Job Detection via eBPF

- * Auto-discovers job lifecycles without modifying user workloads or HTCondor/Slurm configuration.
- * Zero instrumentation, zero code changes.

Integrated with INK and production Condor/Slurm

- * full access to all IHEP users



Job Lens Deployment at IHEP

Deploy
Scale

~1200
Nodes

~64000
CPU Cores

Storage:
4TB in Elastic Search

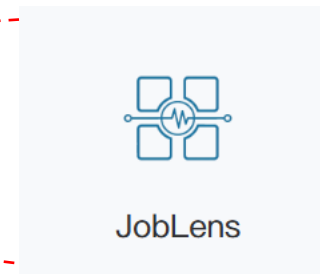
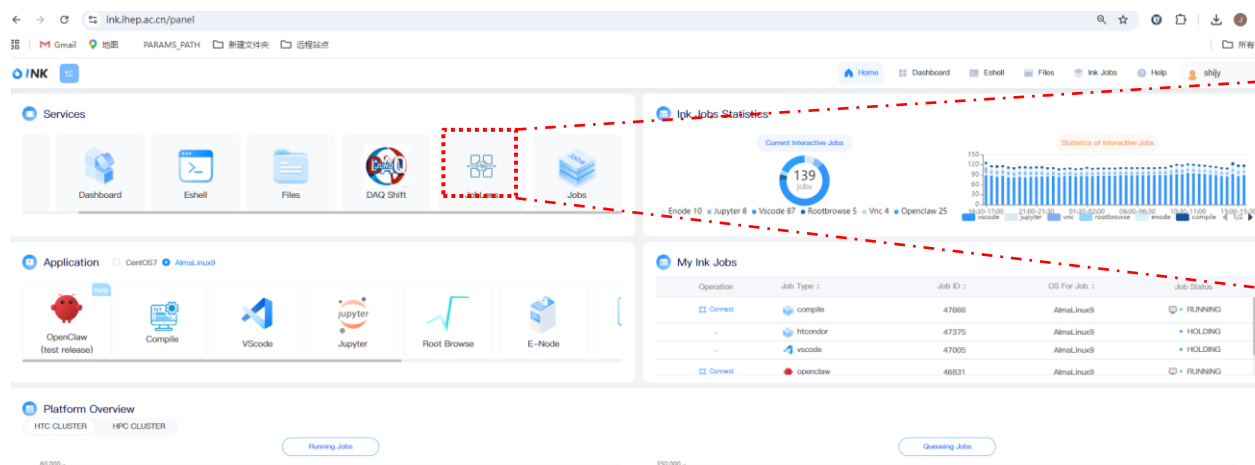
Performance
(Overhead)

CPU Overhead: $< 0.15\%$ per node
(37% of single core on 256-core node)

Memory Overhead: ~145MB
(0.014% of 1TB node memory)

251 Monitored Jobs , 3 collectors/jobs

Current Status



Deployed on INK (Interactive Analysis Workbench) and production Condor/Slurm
---- full access to all IHEP users (with manual trigger mode)

(Exploring Rule-engine-based automatic capture patterns)

Screenshot of Job Lens -- Job Selection

The screenshot displays the Job Lens web interface. At the top, there is a navigation bar with the URL `ink.ihep.ac.cn/joblens` and various utility icons. Below the navigation bar, there are filter controls for 'IHEPCondor', '选择服务节点', and '选择用户组', along with a search box for 'username' and a '刷新数据' button. The main content area features a summary of job statistics: 6195 TOTAL JOBS, 4036 RUNNING, 2159 IDLE, 0 HOLDING, and 0 OTHER. Below the statistics is a table with the following columns: Job ID, Job Status, HepJob_RealGroup, Submit Time, Start Time, Work Node, HepJob_JobType, Owner, and Operation. The 'Operation' column is highlighted with a red dashed box, indicating that users can interact with individual jobs. The table contains 10 rows of job data, all with 'IDLE' status and 'bes_analysis' job type. At the bottom right, there is a pagination control showing '共 6195 条' and '10条/页'.

Job ID	Job Status	HepJob_RealGroup	Submit Time	Start Time	Work Node	HepJob_JobType	Owner	Operation
65140557.0	IDLE	physics	2026-06-12 15:12:48	-	-	bes_analysis	linxingzhu	
65136768.0	IDLE	physics	2026-06-12 12:44:10	-	-	bes_analysis	linxingzhu	
65136768.1	IDLE	physics	2026-06-12 12:44:10	-	-	bes_analysis	linxingzhu	
65136768.2	IDLE	physics	2026-06-12 12:44:10	-	-	bes_analysis	linxingzhu	
65136768.3	IDLE	physics	2026-06-12 12:44:10	-	-	bes_analysis	linxingzhu	
65136768.4	IDLE	physics	2026-06-12 12:44:10	-	-	bes_analysis	linxingzhu	
65136768.5	IDLE	physics	2026-06-12 12:44:10	-	-	bes_analysis	linxingzhu	
65136768.6	IDLE	physics	2026-06-12 12:44:10	-	-	bes_analysis	linxingzhu	
65136768.7	IDLE	physics	2026-06-12 12:44:10	-	-	bes_analysis	linxingzhu	
65136768.8	IDLE	physics	2026-06-12 12:44:10	-	-	bes_analysis	linxingzhu	

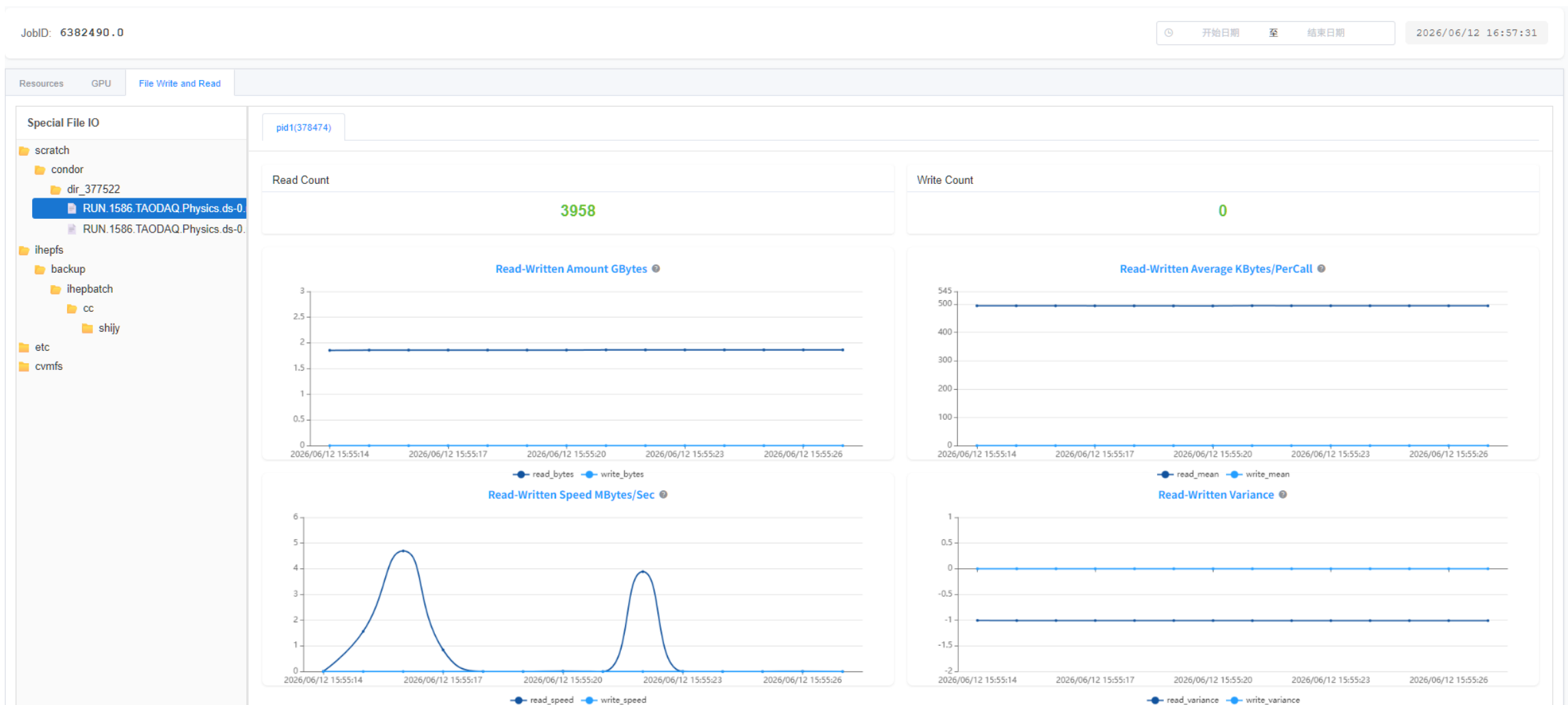
User choose job that he would like to trace

Screenshot of Job Lens -- Resource Usage



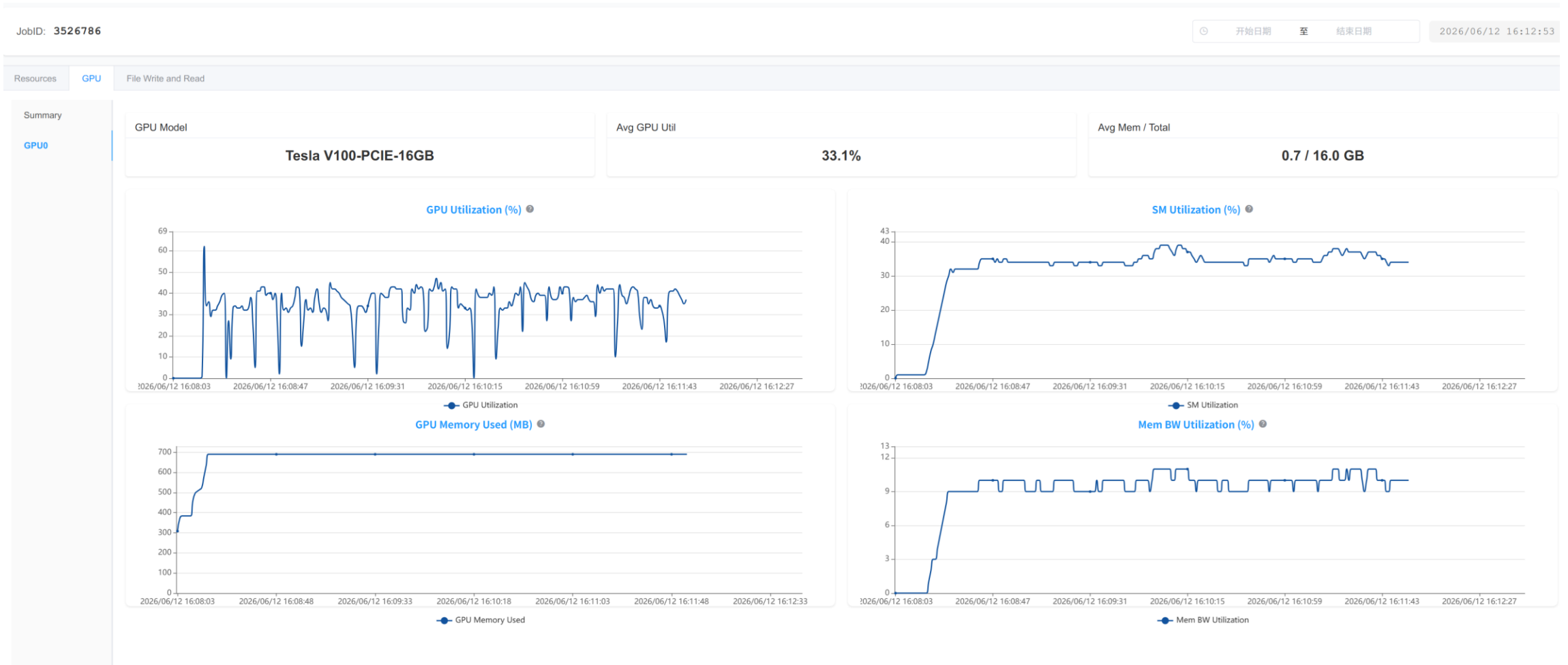
Tracing Job CPU / Memory/ IO Speed/ Network traffic

Screenshot of Job Lens -- File I/O traffic



Tracing file access of the Job: read/write traffic from each file

Screenshot of Job Lens -- GPU Usage



Tracing GPU Usage : GPU Utilization / GPU Memory of each GPU card

Outline

01 INK Recap

02 Progress of INK

03 **AI Exploration at IHEP Computing Platform**

04 Summary

AI Exploration at HTCondor Cluater

OpenClaw + HTCondor

AI agent deployed as an HTCondor job – intelligent computing assistance directly through the batch system

Pre-built Skills:

- * Cluster job submission & query
- * Storage space usage queries
- * Batch system health checks
- * Job tracing from Job Lens

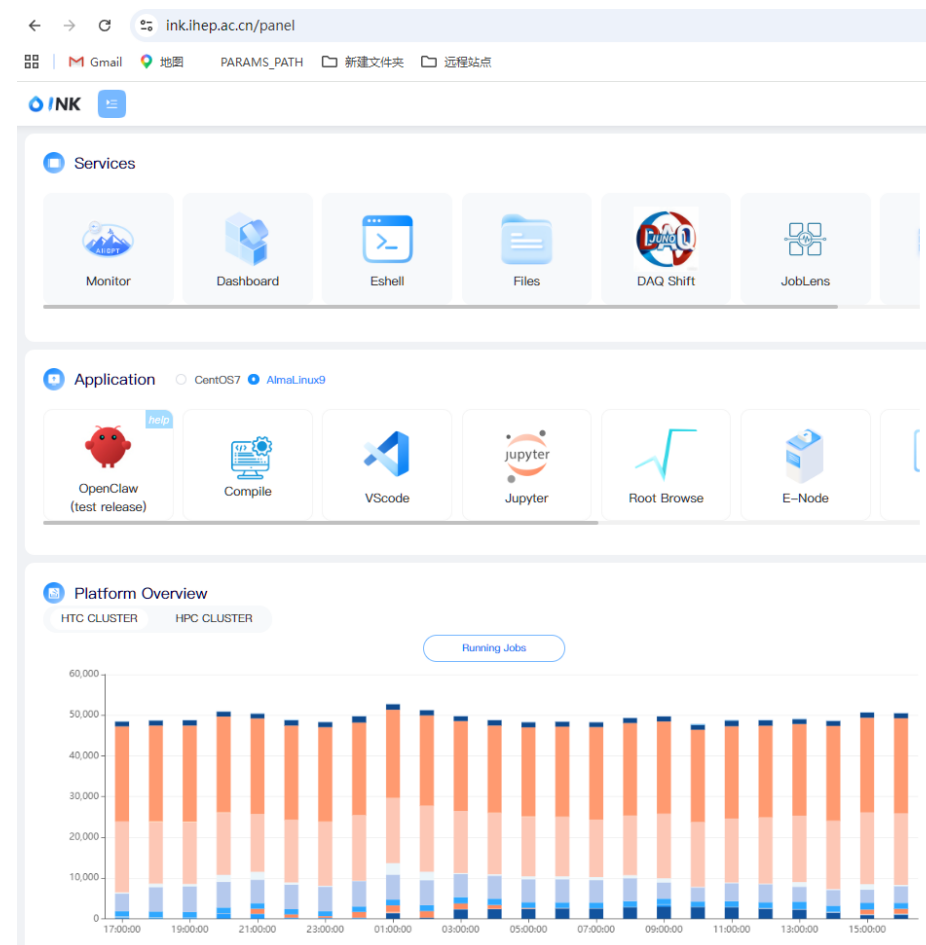
Release in April, 2026

External LLM Models (Safe)

- * Only the scratch directory is exposed as workspace. Experiment data remains protected -- no data leakage possible.

IHEP-Hosted LLM Models (Full Access)

- * Users can directly access experiment data
- * Full flexibility with complete data locality



Screenshot of Openclaw Inside INK

ink.ihep.ac.cn/openclaw/lhws312/60016/shijy/chat?session=main

OpenClaw > main > 聊天

main hepai/deepseek-v4-flash Off

Show me IHEP/HEP Computing skill!

show me your IHEP/HEP Computing skill.

You 2026年6月12日 15:06

Here are my **IHEP / HEP Computing skills**:

Skill	Description
cern-root	ROOT data analysis (TH1/TH2/TH3, TGraph, TF1 fitting, TTree I/O, RDataFrame, RooFit, TMVA, PyROOT)
dirac	DIRAC job/data management in IHEP distributed computing environment — submit/monitor jobs, manage files, proxy auth
hepjob	<code>hep_sub</code> / <code>hep_q</code> / <code>hep_rm</code> / <code>hep_release</code> / <code>hep_edit</code> / <code>hep_clus</code> — job submission, queue querying, deletion on IHEP HPC
hepjob-queue	HPC queue & scheduler mapping — experiment-to-HTCondor schedd/group mapping, queue aliases, user quotas
ihep-computing	IHEP computing platform guide — accounts, login (SSH/gateway), storage systems (<code>/scratchfs</code> , <code>/cvmfs</code> , <code>/workfs</code>), software environment
ink-download	Generate direct download links for files on IHEP cluster (ink storage)
joblens-tap-api	Cluster observability data-fetch layer at <code>http://192.168.82.25:8081</code> — CPU/mem/IO metrics via timeseries, summary, raw endpoints; outputs to <code>/tmp/tap-result.json</code>

7 skills covering the full stack — job submission, data analysis, computing environment, monitoring. Need to use any of them?

Assistant 2026年6月12日 15:06 > Context

6% context used 33.7k / 524.3k

Screenshot of Openclaw Inside INK

The screenshot displays the OpenClaw interface within the INK environment. The browser address bar shows the URL: `ink.ihep.ac.cn/openclaw/lhws312/60016/shijy/chat?session=main`. The interface includes a search bar, a chat input field, and a terminal window. A blue callout box highlights the text: "Show me my job running in juno group now!". The terminal output shows the command: `export SHELL=/bin/bash && source /cvmfs/common.ihep.ac.cn/software/hepjob/setup_hepjob.sh && hep_q -g juno -u shijy 2>&1`. Below the terminal, a "Tool output" section shows the command: `exec`. A "Tool" section displays a table of jobs in the JUNO group:

You have **2 jobs** in the JUNO group:

JobID	Status	Owner	Submitted	Run Time	Script
6382490.0	Running	shijy	12 Jun 15:18	0:01:26	<code>RUN.1586.TAODAQ.Physics.ds-0.mix_stream...</code>
441255.0	Idle	shijy	15 Nov 00:24	—	<code>job-rpm-qa.sh</code>

One job actively running, one still sitting idle since November. Let me know if you want to check the hold reason for the idle one or see the running job's details.

8% context used 43.4k / 524.3k

给 Assistant 发消息 (Enter 发送)

附加文件 (0) 开始 Talk

+ 新会话 导出 发送

Screenshot of OpenCLaw Inside INK

The screenshot shows a web browser window with the URL `ink.ihep.ac.cn/openclaw/lhws312/60016/shijy/chat?session=main`. The interface is titled "OpenCLaw > main > 聊天" (Chat). A search bar is visible at the top right. The chat area shows a user message: "tell me the details of the job 6382490.0." with a timestamp of "You 2026年6月12日 16:41". A system response follows, including a terminal command: `fetch http://192.168.82.25:8081/data/summary?cluster=IHEPCondor&job=6382490.0 -> run jq .` and a "Tool output" section. The tool output displays job information for "Job 6382490.0 — via JobLens" with the following data:

Metric	Value
Cluster	IHEPCondor
First seen	2026-06-12 15:22 CST
Last seen	2026-06-12 15:55 CST
Duration	33 min 8 sec

CPU	Value
Avg	87.9%
Max	102.1%
P99	131.5%

Memory	Value
Avg RSS	623 MB
Peak RSS	720 MB

I/O	Value
Total	18.6 MB

Job finished — no longer in queue.

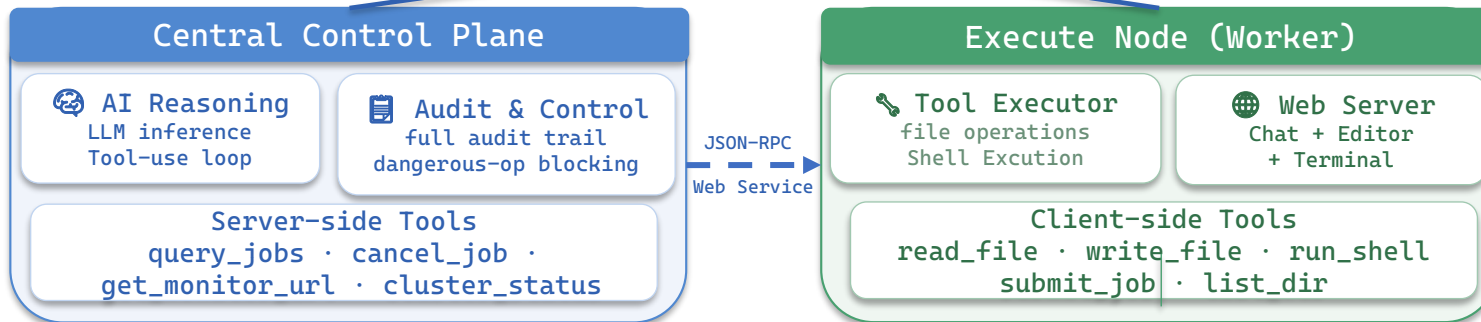
At the bottom of the chat area, a status bar indicates "9% context used 49.2k / 524.3k".

Future Plan : Seed Stage – Centralized AI Agent

Problem: Users install independent AI coding tools
(Cursor, Claude Code, etc.)
– no audit, no control for admins

IHEP Interactive Analysis Workbench (INK)
User Auth · App Portal · Unified HTCondor / Slurm Scheduling

User clicks "AI Agent" → Submits Cluster Job



User Browser – AI Agent Web UI
<https://ink-dev.ihep.ac.cn/agent/{node}/{port}/{user}/>

Chat Panel (dialogue + tool status)

Code Editor (CodeMirror 6)

Terminal (xterm.js)

Architecture mirrors HTCondor

Central Manager (control panel) + Execute Nodes (worker agents)

Each Agent Client is an HTCondor / Slurm Job – Apptainer · UID isolation · Nginx proxy – same model as Jupyter/VSCoDe

Motivation – One Harness, One Goal

- * Shared scientific target: Exp with a big task
- * Multiple members, multiple roles: MC simulation · code debugging · systematics Validation – all feed the same analysis
- * Centralized Agent Harness specialized sub-agents per-task, on-demand
- * Sub-agent scoped: no overreach
- * Unified tokens & Full audit every operation traced reproducible end-to-end

Design Principle – Mirror HTCondor

- * One Central Manager: harness all sub-agents
- * Control Panel: LLM reasoning + audit + safety
- * Execute Node: sub-agents as HTCondor Job
- * JSON-RPC/WS: only tool commands traverse

User Experience

One-click launch

HTCondor scheduled

INK Browser-based

HTCondor

Apptainer

Nginx

JSON-RPC

Rethinking Cluster Operation -- Global Model + Multi-Thin Agents

Rethinking 1 Why Global Model + Multi-Thin Agents?

- * A single LLM context cannot absorb all operational knowledge
- * Real-world cluster ops require topology, real-time metrics, historical incidents, security policies – far beyond any context budget.
- * Possible solution: specialization and structured retrieval. Multiple domain-specific Sub-Agents, each loading only what it needs.
- * A Global Model with 4 layers – topology, runtime, semantics, memory. Context Assembler dynamically constructs the minimal subgraph for each decision.

Multi-Agents

Global Model

Context Assembly

Knowledge Graph

Rethinking 2 Why MCP as the Execution Abstraction?

- * AI should describe what to do, not how on a specific machine.
- * Every operation flows through a centralized MCP Server: type-safe, permission-gated tool interface.
- * AI calls a semantic tool ("check disk usage"); MCP validates permissions, translates to command, dispatches to Thin Agent. Gives us type safety, permission enforcement, backend independence, centralized auditing.

MCP Protocol

Type Safety

Permission Gateway

Centralized Audit

Rethinking 3 The Human Role: More Than Approving

- * LLMs have zero real-world HEP ops experience.
- * Humans in 3 concurrent roles:
 - decision-makers approving high-risk actions,
 - knowledge contributors enriching the World Model, correctors steering the AI. Creates a positive feedback loop: better AI → more trust → more usage → more data → better AI.

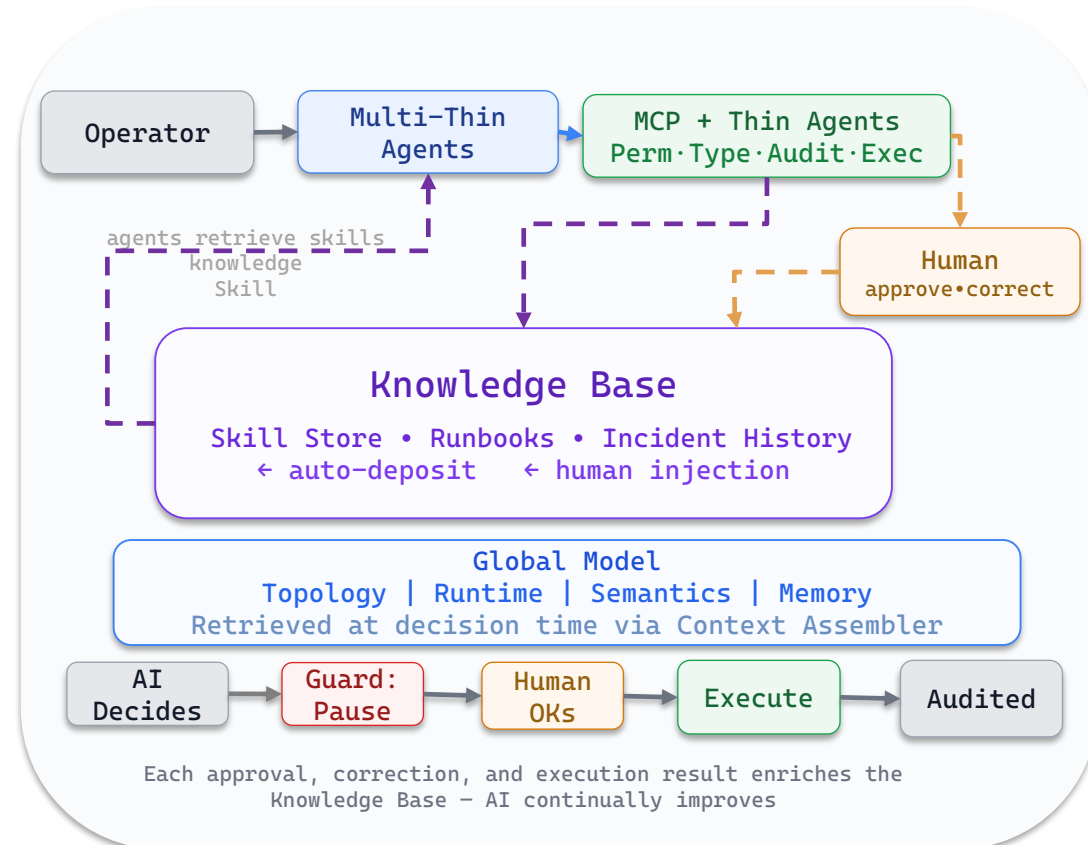
Decision Authority

Knowledge Injection

Correction

Feedback Loop

Global Model + Multi-Thin Agents



Summary

INK

One year in production, growing user base across multiple experiments and sites.

New Capabilities

REST API, JobLens,
One-Click Deploy,
Plugin system – steadily expanding



First AI Steps

OpenClaw deployed via HTCondor,
serving real user

Think Ahead

Seed-stage exploration toward
centralized AI agent
and AIOps control panel

Acknowledgments

Deep thanks to the entire HTCondor team for years of support, and especially to Greg for his help at every turn.

Thank you & Question?