

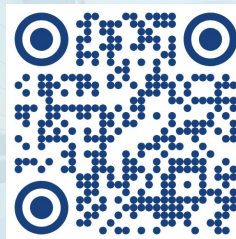


U.S. DEPARTMENT
of ENERGY

DOE American Science Cloud Experiments with NRP

Justas Balcas, Xi Yang
Berkeley Lab, ESnet
Pilots & Prototypes Group

<https://shorturl.at/uFST6>



7NRP Workshop
San Diego
May 5, 2026

Agenda

- DoE American Science Cloud Overview
- DoE Integrated Research Infrastructure and NRP
- Hands-On with IRI API
- AmSC Infrastructure Resource Orchestration Overview and Demo
- Wrap Up + Q&A

Overview of American Science Cloud



The Genesis Mission

A national initiative to build the world's most powerful scientific platform through AI aims to accelerate discovery science, strengthen national security, and drive energy innovation.

Genesis Mission will **develop an integrated platform** that connects the world's best supercomputers, experimental facilities, AI systems, and unique datasets across every major scientific domain. Its goal is to double the productivity and impact of American research and innovation within a decade.



- White House Executive Order launched the mission on 11/24/2025.
- DOE budgets and initial appropriations: \$320M (\$40M AmSC).
- Congress is starting to fund pieces.
- DOE's Genesis Mission Challenges RFA: additional grants \$293M.

The American Science CLOUD (AmSC)

Key building block of the Genesis Mission Platform

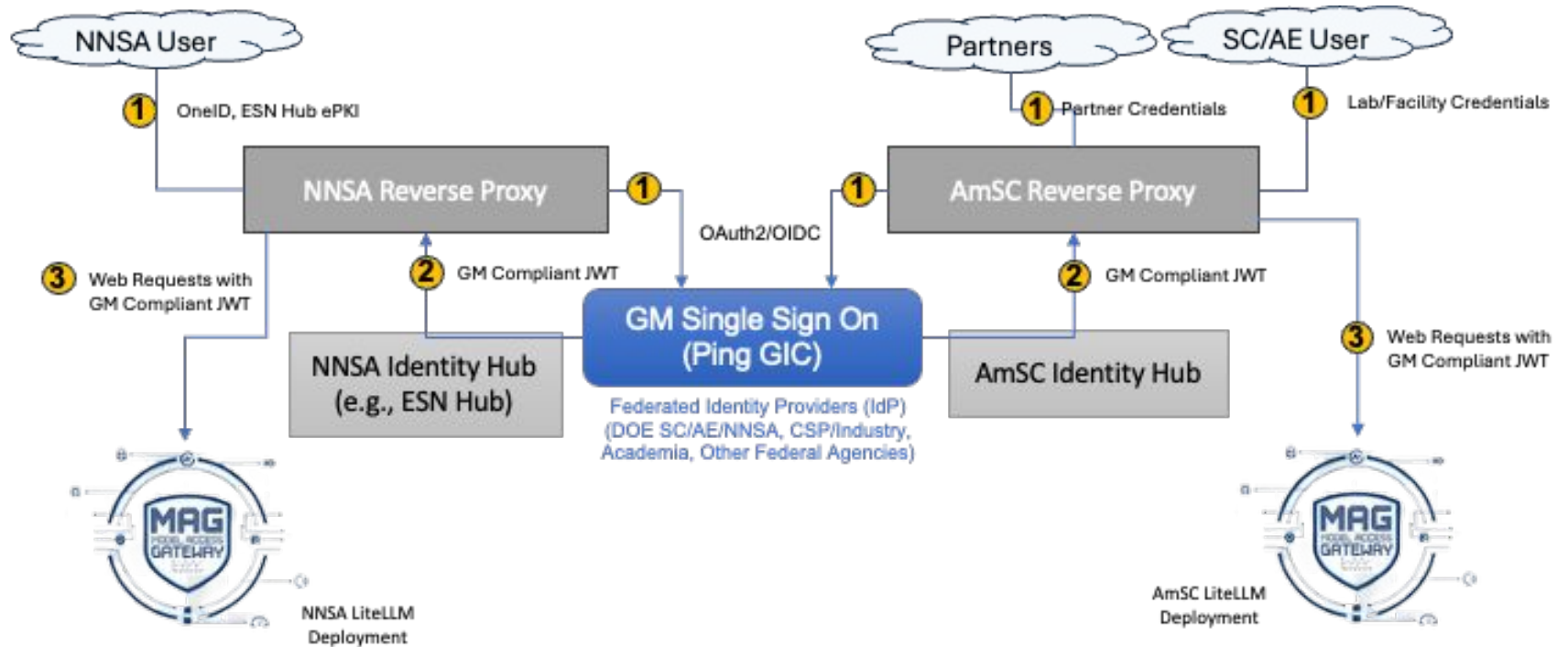
Mission

Deliver a seamless integration of DOE science instruments, computing, experimental, and networking capabilities, along with AI, data, modeling, and simulation software tools, through a single advanced programming interface - **the AmSC API**.

What We Deliver

- ✓ Unified APIs for data, model, and workflow access
 - ✓ Secure, federated identity (Virtual Organization model)
 - ✓ Seamless interoperability between DOE facilities and CSPs
- Reduce “time to insight” from months to days through composable AI-driven workflows**

Integrated AmSC/NNSA Identity Architecture



1-51

AmSC Minimum Viable Product (MVP)

A unified, federated AmSC platform integrating HPC, experimental, and cloud resources under one API and one identity — delivered in 12 months (by 9/30/26) in support of the Genesis Mission and ModCon*.

- **Federated identity (VO):** single sign-on and policy-based access
- **Common AmSC API:** REST + Python SDK + dashboards + MCP, usable by both scientists and AI agents
- **Data, At-Scale, and Model Services:** federated Data Catalog and Data Movement; curated stacks and APIs for training, fine-tuning, experiment tracking, model registry, and inference
- **Infrastructure Integration:** K8s platforms/interworking, IRI facilities, WAN Data Fabric, secure enclaves, cloud partners integration
- **Agentic Workflow PoC:** end-to-end discovery → training → reasoning-based inference across DOE and cloud resources, with cross-site monitoring and AI4Ops

** ModCon is a separate DOE initiative (the Transformational AI Models Consortium) that develops large, multimodal AI models for scientific discovery. AmSC is its platform backbone.*

AmSC MVP Implementation Timeline

3mo

Federated Access Foundation

- Virtual Org Setup
- Federated Identity
- IRI Integration

✓ First operational REST API and Python client and workflow execution.

6mo

Science Team Onboarding

- API Gateway
- Data Catalog
- Workflow

✓ Two to three research teams execute end-to-end workflows.

9mo

Intelligent Workflow

- Agentic Workflow
- AmSC API v1
- IRI & Orchestration

✓ First early agentic workflow framework autonomous discovery and execution.

12mo

Minimum Viable Product

- ASCR/CSP Link
- Unified Discovery
- MVP Release

✓ Link IPs and CSPs into a unified pre-production environment.

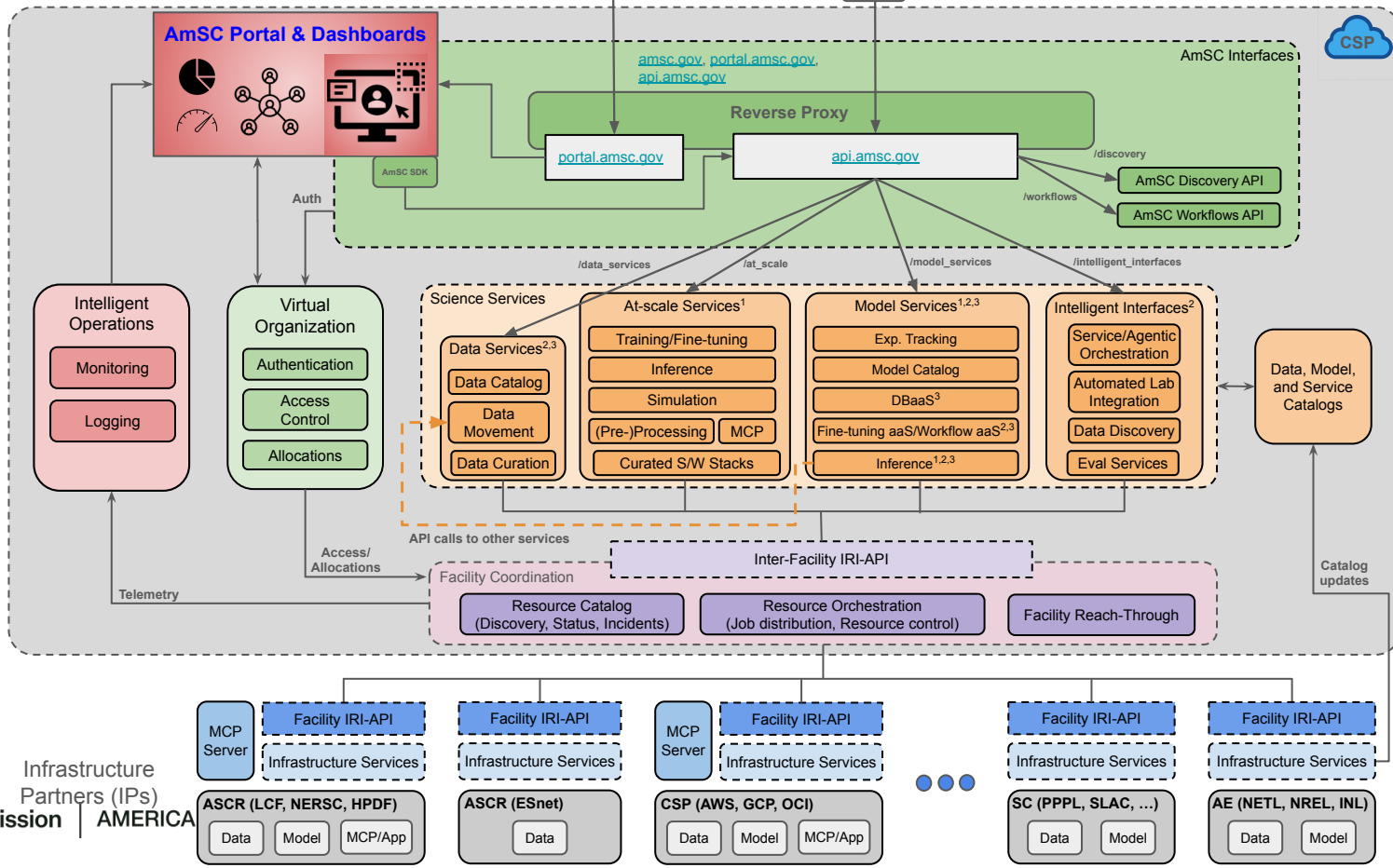
Establish a foundation for future expansion to additional partners and services.

AmSC Platform Architecture

Users, Operators, Managers

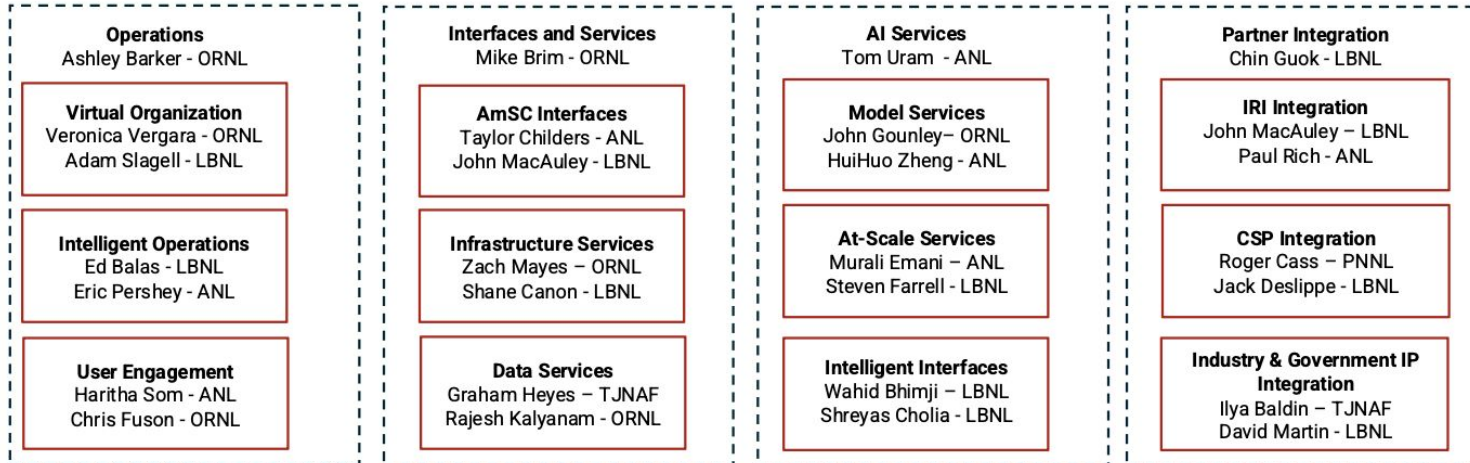
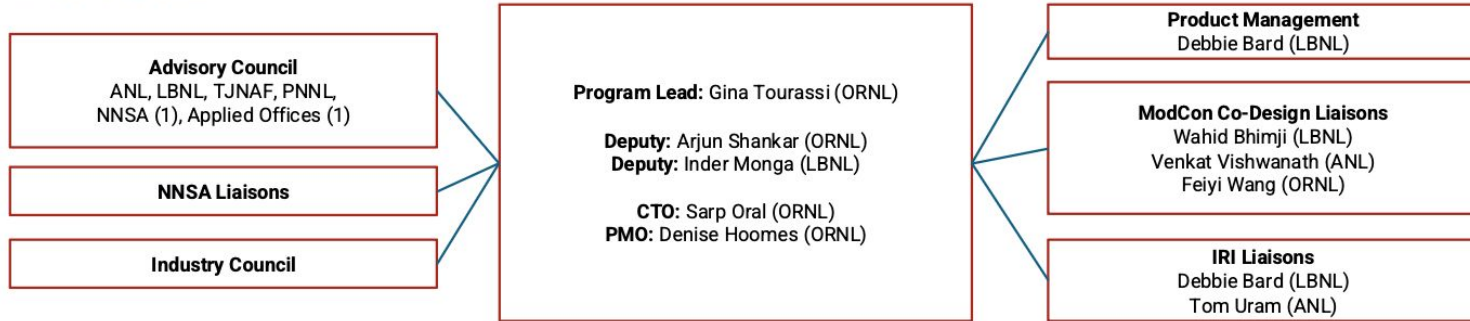


Software & Services from
IPs and 3rd Parties



AmSC Organizational Hierarchy

AmSC Org Chart



Infrastructure Services



Container Platforms

Federated Kubernetes across DOE sites and CSPs with common schema.



Storage Interfaces

S3 and Globus interfaces over heterogeneous facility storage.



Secure Enclaves

Sensitive and controlled-data execution environments (e.g., CITADEL).



Job Scheduling

Common interface policies backend to local facility schedulers.



Resource Orchestration

Metadata schema, discovery, credentials, and job placement.



WAN Data Fabric

ESnet L3VPN and SENSE-orchestrated cloud interconnects.

Infrastructure Partners (IP)

- **ASCR HPC:** OLCF (Frontier), NERSC (Perlmutter), ALCF (Aurora), HPDF
- **ASCR Networking:** ESnet
- **Other Office of Science labs:** PPPL, SLAC, ...
- **Applied Energy labs:** NETL, NREL, INL
- **Commercial CSPs:** AWS, GCP, OCI, ...

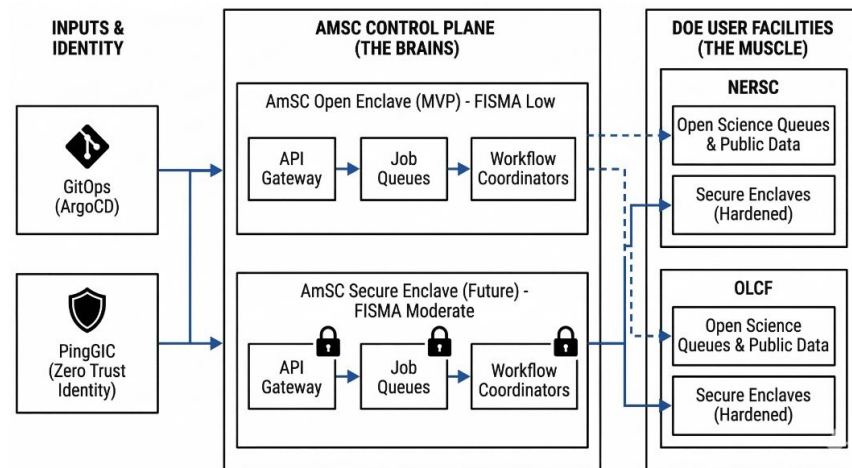
IPs plug into AmSC through a common abstraction layer - the IRI-API (DOE Integrated Research Infrastructure API)

AmSC, IRI and NRP









AmSC Platform Elements







- **Everything in Code (GitOps)**
 - Manage infrastructure & apps via versioned configs
- **Zero Trust**
 - Replace “trusted networks” by secure VPNs + Short-lived tokens
- **Control Plane (*The Brains*)**
 - Orchestration, Identity Mgmt, API Routing
- **Facilities (*The Muscle*)**
 - Heavy computation & petascale-scale data movement
- **Data Plane**
 - High-speed science data routes (e.g., via ESnet/L3VPN)



AmSC has a clearly defined Service Readiness schema

	Label	User Access & Expectation	API Contract & Stability (RFC Focus)	Operational Support & SLA
	Concept	Core author(s) only. Code may not exist yet; entirely theoretical or a local hack. Expectation: It might never be built or merged.	Undefined or highly fluid. Currently in the RFC drafting phase or utilizing hardcoded mocks.	None. Does not exist in any shared deployment environment.
	Sandbox	AmSC staff only. No external users.	Unstable. Rapid, undocumented breaking changes.	None. Local/Dev environments only.
	Incubator	Initial testing by select beta-testers (e.g., specific IP/Model teams). Expect rapid evolution; do not build mission-critical workflows on it.	Experimental (v0.x). Breaking changes allowed. No backward compatibility guarantees.	Best Effort. Business hours only. No on-call expectations if the service goes down.
	Candidate	Broad testing by all users. Fairly stable. Users can build it and kick the tires, but minor breaking changes might still occur before graduation.	Stabilizing (v0.9.x). API contract is established. Breaking changes require a notification window (e.g., 2 weeks) to consumers (e.g. downstream projects, ModCon or IPs).	High Priority. Monitored by the core team during business hours. Bugs are prioritized in the next sprint.
	Graduated	Broad use by all authorized AmSC users. Fully stable, whole-number release.	Locked (v1.0+). Strict backward compatibility enforced. Breaking changes require a major version bump and a formal deprecation period.	Production SLA. 24/7 telemetry monitoring. On-call response times are dictated by the component's specific criticality tier
	Archived	The component is being retired. Users must migrate.	Frozen. No new features, only critical security patches.	Sunset. Support ends on a predefined date.

AmSC has a clearly defined Service Readiness schema

	Label	User Access & Expectation	API Contract & Stability (RFC Focus)	Operational Support & SLA
	Concept	Core author(s) only. Code may not exist yet; entirely theoretical or a local hack. Expectation: It might never be built or merged.	Undefined or highly fluid. Currently in the RFC drafting phase or utilizing hardcoded mocks.	None. Does not exist in any shared deployment environment.
	Sandbox	AmSC staff only. No external users.	Unstable. Rapid, undocumented breaking changes.	None. Local/Dev environments only.
	Incubator	Initial testing by select beta-testers (e.g., specific IP/Model teams). Expect rapid evolution; do not build mission-critical workflows on it.	Experimental (v0.x). Breaking changes allowed. No backward compatibility guarantees.	Best Effort. Business hours only. No on-call expectations if the service goes down.
	Candidate	Initial testing by select beta-testers (e.g., specific IP/Model teams). Expect rapid evolution; do not build mission-critical workflows on it.	Experimental (v0.x). Breaking changes allowed. No backward compatibility guarantees.	Best Effort. Business hours only. No on-call expectations if the service goes down.
	Graduated	Core. Fully stable, wide number release.	Stable. Breaking changes require a major version bump and a formal deprecation period.	24/7 telemetry monitoring. On-call response times are dictated by the component's specific criticality tier
	Archived	The component is being retired. Users must migrate.	Frozen. No new features, only critical security patches.	Sunset. Support ends on a predefined date.

Everything we are showing/giving access to this week is in the "Incubator" stage. We invite you to test these services in your role as AmSC Beta Users and welcome your feedback!

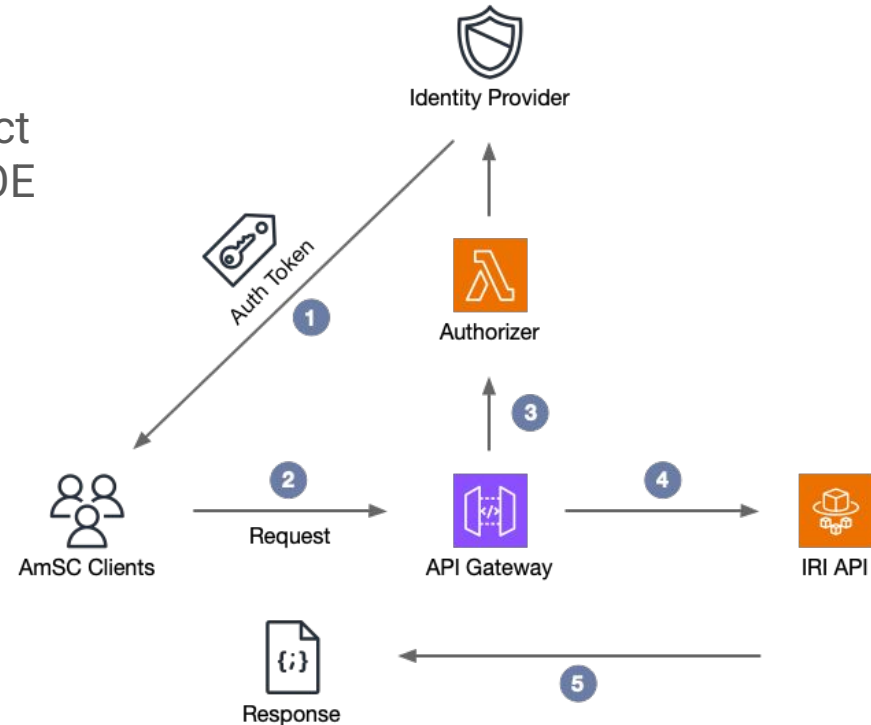
Cross-DOE and CSP Compute Access

Integrating with AmSC sites and facilities

Goal: Enable automated workflows across all AmSC sites

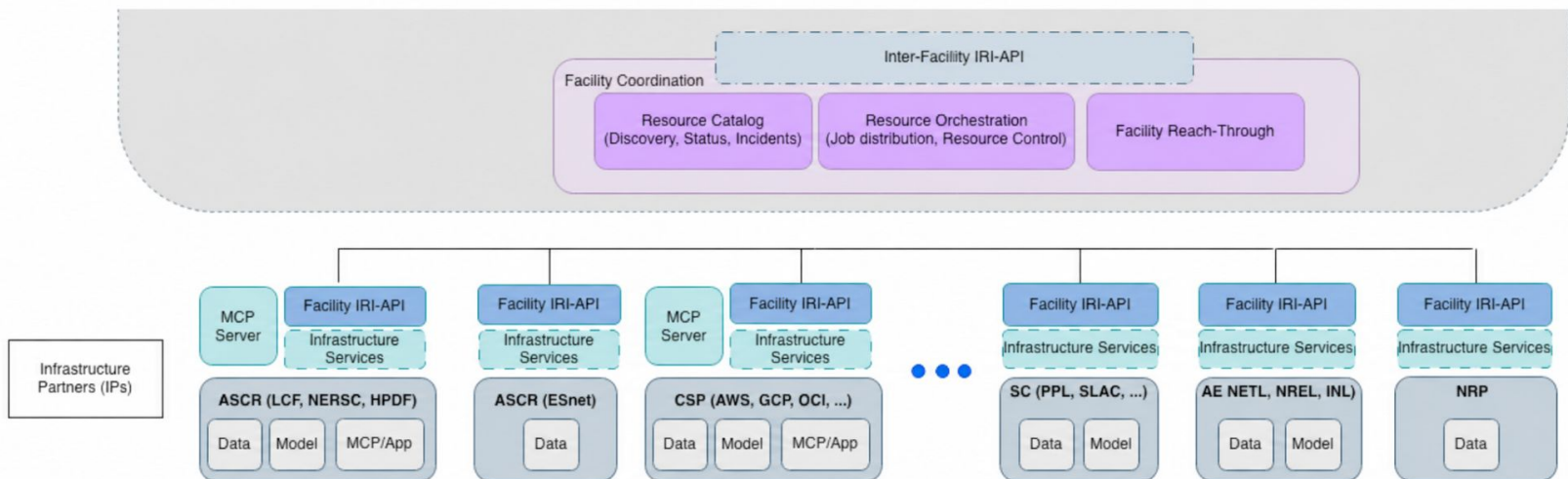
AmSC services all use a common API to interact with AmSC IP sites, including HPC facilities, DOE lab resources and CSPs. Endpoints include:

- resource status
- job management
- local file management
- allocation information
- resource orchestration
- WAN management



Accessing “The Muscle”

- Under-pinning the AmSC Platform
- IRI-API provides access to DOE facilities, Cloud Service Providers, etc.



IRI Interfaces Charter

- Gather requirements, define use cases, and specify unified interfaces for ~~the DOE ASCR~~ Facilities.
- Tasked with enhancing the "user experience" for interaction and management of complex workflows across ASCR facilities
 - Goal is to facilitate a more cohesive and efficient research infrastructure.
- Consider applicable standards and, where available, build upon the work of other organizations to deliver a reference IRI interface implementation.

Launched May 2024

Interfaces Technical Subcommittee

- Design a **minimal functional API** and deploy it at multiple sites
 - Review existing API schemas
 - Propose IRI schema: endpoints, architecture, infrastructure,...
 - Implement MVP
- Explore how to align **Jupyter** across sites

Co-Chairs: John MacAuley, Paul Rich

IRI Development - How and What

- The How
 - The Interfaces Subcommittee co-designs the IRI specification
 - Ideas are discussed, prototyped, and battle-tested before they land in the spec
 - Decisions are not sacred - we revisit and refine them as we learn
- The What
 - HTTP/JSON APIs shaped by pragmatic REST principles
 - An OpenAPI specification serves as the formal contract and documentation
 - Reference implementations jump-start adopters and boost development velocity
 - Multiple independent implementations validate interoperability and the spec itself

IRI Facility Specification (version 1)

Facility



- Facility
- Site
- Location
- Resource

Allocation



- Project
- UserAllocation
- ProjectAllocation
- AllocationEntry
- Capability
- Resource

Compute



- Job
- Resource

Status



- Incident
- Event
- Resource

Filesystem



- File
- Path
- Resource

V1 - Job Submission

- PSI/J Initial Approach
- Supported operations
 - Submit
 - Live status
 - Delete/Cancel
- Workload Managers
 - SLURM API
 - PBS API
 - Kube Kueue
 - HTCondor

compute

POST	/api/v1/compute/job/{resource_id}	Submit Job	🔒	▼
PUT	/api/v1/compute/job/{resource_id}/{job_id}	Update Job	🔒	▼
GET	/api/v1/compute/status/{resource_id}/{job_id}	Get Job Status	🔒	▼
POST	/api/v1/compute/status/{resource_id}	Get Job Statuses	🔒	▼
DELETE	/api/v1/compute/cancel/{resource_id}/{job_id}	Cancel Job	🔒	▼

V1 - Local FileSystem Operations

- Local Operation Only
 - Complimentary to transfer frameworks
- Asynchronous only
- Similar to operations through SSH

filesystem			^
PUT	/api/v1/filesystem/chmod/{resource_id}	Put Chmod	🔒
PUT	/api/v1/filesystem/chown/{resource_id}	Put Chown	🔒
GET	/api/v1/filesystem/file/{resource_id}	Get File	🔒
GET	/api/v1/filesystem/stat/{resource_id}	Get Stat	🔒
POST	/api/v1/filesystem/mkdir/{resource_id}	Post Mkdir	🔒
POST	/api/v1/filesystem/symlink/{resource_id}	Post Symlink	🔒
GET	/api/v1/filesystem/ls/{resource_id}	Get Ls Async	🔒
GET	/api/v1/filesystem/head/{resource_id}	Get Head	🔒
GET	/api/v1/filesystem/view/{resource_id}	Get View	🔒
GET	/api/v1/filesystem/tail/{resource_id}	Get Tail	🔒
GET	/api/v1/filesystem/checksum/{resource_id}	Get Checksum	🔒
DELETE	/api/v1/filesystem/rm/{resource_id}	Delete Rm	🔒
POST	/api/v1/filesystem/compress/{resource_id}	Post Compress	🔒
POST	/api/v1/filesystem/extract/{resource_id}	Post Extract	🔒
POST	/api/v1/filesystem/mv/{resource_id}	Move Mv	🔒
POST	/api/v1/filesystem/cp/{resource_id}	Post Cp	🔒
GET	/api/v1/filesystem/download/{resource_id}	Get Download	🔒
POST	/api/v1/filesystem/upload/{resource_id}	Post Upload	🔒

Full Site Validation

- Facilities tested periodically:
 - API Readiness
 - Full workflow lifecycle
 - Includes Real Filesystem and Compute Ops.

Operational Sites:

- NERSC, ALCF, ESnet(CSP)

Work in Progress:

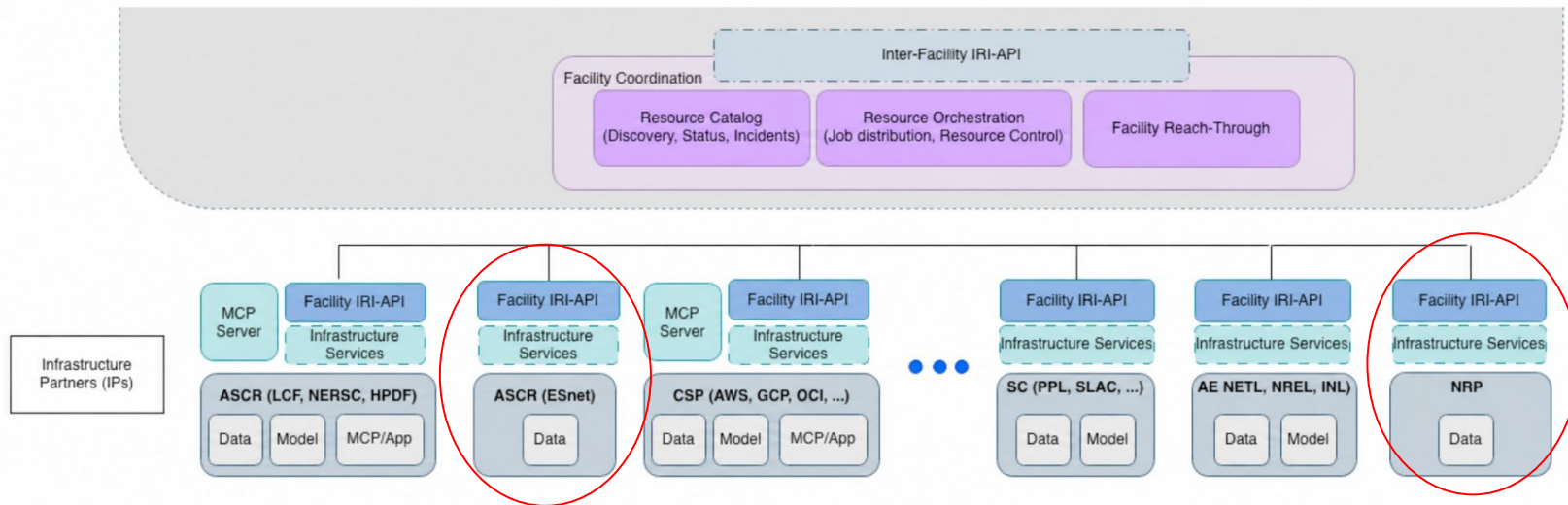
- ORNL, PNNL, SLAC, BNL

operationId	DemoAdapter	NERSC	ALCF	ESnet_West	ESnet_East	ORNL	PNNL	SLAC	BNL
GET /api/v1/facility	PASS	PASS	PASS	PASS	PASS				
GET /api/v1/facility/sites	PASS	PASS	PASS	PASS	PASS				
GET /api/v1/facility/sites/{site_id}	PASS	PASS	PASS	PASS	PASS				
GET /api/v1/status/events	MISSING	MISSING	PASS	MISSING	MISSING				
GET /api/v1/status/events/{event_id}	MISSING	MISSING	PASS	MISSING	MISSING				
GET /api/v1/status/incidents	PASS	PASS	PASS	PASS	PASS				
GET /api/v1/status/incidents/{incident_id}	PASS	PASS	PASS	PASS	PASS				
GET /api/v1/status/resources	PASS	PASS	PASS	PASS	PASS				
GET /api/v1/status/resources/{resource_id}	PASS	PASS	PASS	PASS	PASS				
GET /api/v1/account/capabilities	PASS	PASS	FAIL	PASS	PASS				
GET /api/v1/account/capabilities/{capability_id}	PASS	PASS	FAIL	PASS	PASS				
GET /api/v1/account/projects	FAIL	PASS	PASS	PASS	PASS				
GET /api/v1/account/projects/{project_id}	PASS	PASS	PASS	PASS	PASS				
GET /api/v1/account/projects/{project_id}/project_allocations	PASS	PASS	PASS	PASS	PASS				

Soon!
Work in Progress

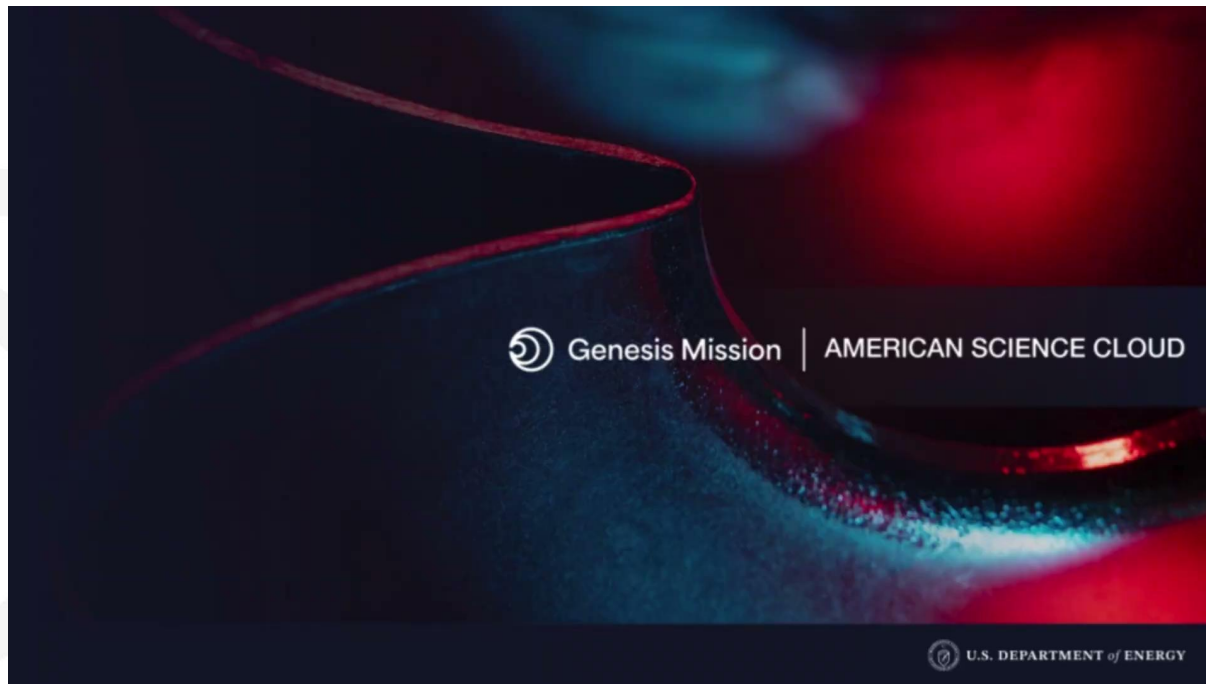


Accessing “The Muscle” on ASCR/CSP and NRP



Pre-Recorded Demos

https://drive.google.com/drive/folders/1SWtfeB_-fjSQJNwdQNnwvVVsOEaRah9b?usp=sharing



Hands On Time

Use NRP Jupyter Hub:

<https://training.nrp-nautilus.io/hub/login>

Jupyter notebooks: <https://github.com/doe-iri/iri-facility-api-examples>

Git clone <https://github.com/doe-iri/iri-facility-api-examples>

Cd iri-facility-api-examples/

[./start-notebook.sh](#) # Only if running locally, not JupyterHub

Hands On Time

1. **Login-nrp.ipynb (Write your own creds)**
2. **Compute-jobs.ipynb (Submit a compute job)**
3. **Filesystem.ipynb (Filesystem operations)**

V2. Chat with IRI Endpoints

```
Submit an IRI compute job with the following configuration:
Resource: esnet-east (fb0aafe1-c780-55c0-b635-a7121f1b0ce5)
Run a bash login shell in /data/home/jbalcas that:
- Prints system info (time, hostname, user)
- Dumps environment variables (sorted)
- Writes a log file to /data/home/jbalcas/esnet_iri_test.log
```

```
Capture stdout and stderr into:
- /data/home/jbalcas/esnet_iri_test_stdout.log
- /data/home/jbalcas/esnet_iri_test_stderr.log
```

```
Resources:
- 1 node, 1 process, 1 CPU core
- 256MB memory
- Exclusive node usage
```

```
Scheduler settings:
- Queue: debug
- Account: interactive
- Max duration: 600 seconds
```

I'll submit that job to the ESnet East resource for you: Perfect! Your job has been successfully submitted to ESnet East. Here are the details:

Job Submission Success

```
Job ID: 8a935d40-17cf-4d47-a070-24920da56fb1
Resource ID: fb0aafe1-c780-55c0-b635-a7121f1b0ce5
Status: New
Job Name: iri-test-job-20260326-150708
```

The job is a smoke test that will:

```
Display system information (date, hostname, user)
Output environment variables
Create a log file at /data/home/jbalcas/
```

Here's the facility information for NERSC:

National Energy Research Scientific Computing Center (NERSC)

Description: NERSC is the mission science computing facility for the U.S. Department of Energy Office of Science, the nation's largest supporter of basic research in the physical sciences.

Short Name: NERSC

NERSC is a major computing center that provides high-performance computing resources for scientific research funded by the Department of Energy. Would you like more information about NERSC's resources, capabilities, or other details?

```
Get job status on ESnet East for Job ID: 8a935d40-17cf-4d47-
a070-24920da56fb1
Resource ID: fb0aafe1-c780-55c0-b635-a7121f1b0ce5
Job Name: iri-test-job-20260326-150708
```

I'll get the job status for you on ESnet East. Perfect! Here's the job status for **iri-test-job-20260326-150708**:

Job Status Summary:

```
Job ID: 8a935d40-17cf-4d47-a070-24920da56fb1
Resource ID: fb0aafe1-c780-55c0-b635-a7121f1b0ce5
Site: ESnet East
Current State: ● ACTIVE/RUNNING
Exit Code: -1 (still running)
Start Time: 2026-04-06T14:30:42Z
Completion Time: Not yet completed
Metadata:
```

```
Active: 1
Succeeded: 0
Failed: 0
```

The job is currently running on ESnet East and has been active since April 6, 2026 at 14:30:42 UTC.



Roadmap to V2: Static resource specification

Enhance IRI resource descriptions and align with AmSC resource cards to allow for a consistent discovery strategy. This includes (mostly) static information such as:

- More detailed definition of a resource:
 - Compute (Node, CPU, GPU, container) + capacity, limits, and queuing policies.
 - Storage (direct connect, NFS) + capacity, mounts, limits, retention policies.
 - DTN (Globus, XRootD) endpoints.
 - Networking connectivity, Service definitions (i.e., inference).
 - Flexible resource definitions with resource-specific extensions.
- CUI capabilities for resources. (Controlled unclassified information)
- Facility topology with connectivity modeling (DTN <-> storage <-> compute)

This would allow workflows to determine job capabilities and associated policies at each facility, storage and transfer capabilities, and which DTN endpoints are used to transfer data to/from storage associated with a specific compute resource.

Roadmap to V2: Dynamic resource specification

In support of workflow/resource orchestrations, we will add situational awareness of resources at a Facility. This includes real-time visibility into:

- Current compute resource capacities (Total/InUse/Idle/Min/Max CPU, GPU), job queue access, and wait times.
- Storage resource capacities, limits for project, home, and scratch directories.

This would allow us to estimate the best location for job runtime, forward jobs to correct facilities, and optimize delivery time.

Roadmap to V2: Four Tier Deployment and Exc Model

- Control Plane Services (Brain-resident and maintained services)
- User-Deployed Services (Brain-resident deployed services)
- Facility Maintained Services (Inference Service / long-running services)
- User-Deployed Services on Facilities (Custom MCPs, long-running services)

To enhance and allow users to deploy their own services either on AmSC or Facilities in a unified way, minimally this includes:

- Container capabilities
- Upload container
- Custom Job Scripts
- Persistent services

Summary

DOE-IRI: <https://github.com/orgs/doe-iri/repositories> (api-docs, api-ref, api-examples, api-toolkit)

New API Requests: <https://github.com/doe-iri/iri-facility-api-docs/issues/new?template=newapi.yml>

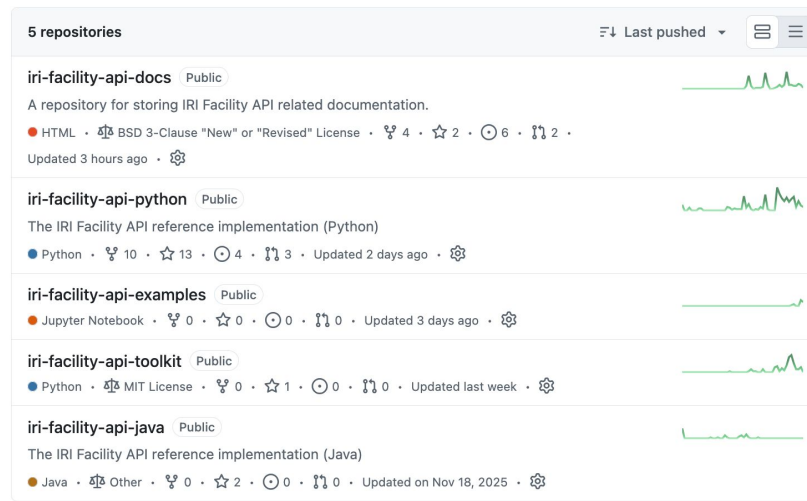
V2 Requests in Progress

3 Facilities fully compliant with V1 Specification

- NERSC: <https://api.iri.nersc.gov/>
- ALCF: <https://api.alcf.anl.gov/>
- ESnet East: <https://iri-dev.ppg.es.net/>
- ESnet West: <https://esnet-west.sdn-sense.net/>

4 Other facilities finalizing V1 Implementation

Link to slides: <https://shorturl.at/h6e47>



A screenshot of a GitHub repository page showing five repositories under the 'doe-iri' organization. The repositories are listed in a table-like format with their names, descriptions, and various statistics. The repositories are: 'iri-facility-api-docs', 'iri-facility-api-python', 'iri-facility-api-examples', 'iri-facility-api-toolkit', and 'iri-facility-api-java'. Each repository has a small green line graph icon to its right, indicating its activity or popularity over time.

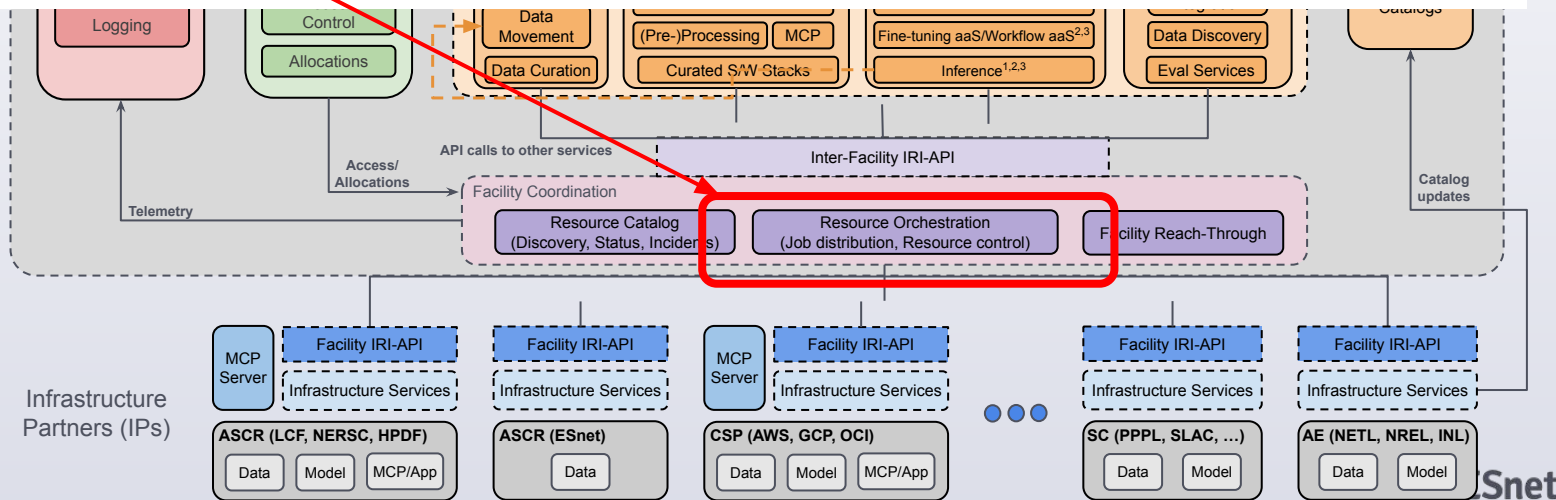
Repository Name	Public	Description	Language	Stars	Forks	Issues	Updated
iri-facility-api-docs	Public	A repository for storing IRI Facility API related documentation.	HTML	4	2	6	Updated 3 hours ago
iri-facility-api-python	Public	The IRI Facility API reference implementation (Python)	Python	10	13	4	Updated 2 days ago
iri-facility-api-examples	Public	Jupyter Notebook	Jupyter Notebook	0	0	0	Updated 3 days ago
iri-facility-api-toolkit	Public	The IRI Facility API reference implementation (Python)	Python	1	0	0	Updated last week
iri-facility-api-java	Public	The IRI Facility API reference implementation (Java)	Java	2	0	0	Updated on Nov 18, 2025

AmSC Infrastructure Resource Orchestration

A background graphic consisting of a network of interconnected nodes and lines, rendered in shades of blue and white, suggesting a complex infrastructure or data network.

AmSC-IRO: Infrastructure Resource Orchestration

- **AmSC-IRO** is the **middleware** that coordinates the discovery and provisioning of IRI and CSP resources across the facilities, ESnet and cloud interconnects.
- While a **scientific workflow** defines **what data and process** to run and **in what order**, AmSC-IRO interprets the **workflow intent** to determine **where and how** the underlying **infrastructure services** are called to provide the needed resources.



API Driven Workflow User/Agent Interaction

- **Resource Catalog for Discovery and Registry**
 - Workflow assistance with facility list, status, capabilities, resource availability and API specs
 - Data Services mapping of data access endpoints at IRI facilities
 - WAN Data Fabric endpoints and cloud interconnects for inter-facility data transfer / streaming
 - Registry for API endpoints for various infrastructure services
 - Registry for per-user and per-workflow policies
- **Cross-Facility Job Placement**
 - Orchestrated job placement based on facility status, resource availability and allocation.
 - Provide abstraction and consistency with orchestrated job placement with multi-facility.
 - Coordinate job placement with network and data transfer endpoints.
- **Networked Multi-Service Orchestration**
 - Interconnect distributed IRI compute and data jobs with the WAN Data Fabric service.
 - Interconnect cloud-to-facility and inter-cloud workloads over WAN Data Fabric service.
 - Workflow assistance with compute → data → network service interdependency orchestration.

AmSC-IRO Live Demo



SENSE-O AmSC-IRO

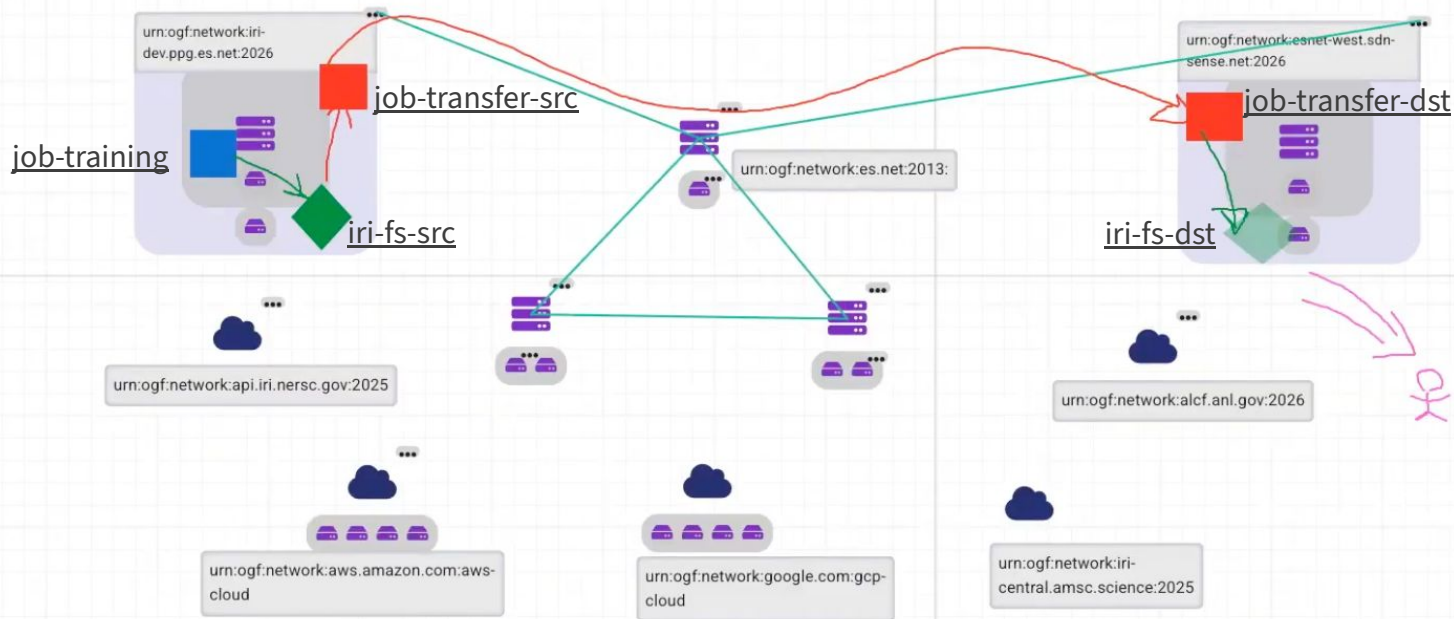
Model Visualization



Open Details

Top Level

Search topology...



AmSC-IRO Live Demo

- AmSC-IRO
 - SENSE-O based core orchestration engine
 - Web portal and Intent-based Northbound API
 - Keycloak SSO for authN and authZ
 - Support foreign tokens (like Globus) with token exchange
- IRO Workflow Client Toolkit
 - Python SDK with workflow examples
 - Tool-based pluggable with multi-providers, multi-facilities
 - Support IRI and IRO API
 - Rich resource discovery and interworking functions
- Demo through Jupyter notebook
 - Multi-provider authN configs
 - Workflow interactions via the IRO toolkit

Wrap up

Q&A

- AmSC vs NRP - What's common
 - Many similar workflows
 - Multi-sites
 - Distributed computing
 - Large k8s operations
 - AI workloads
- AmSC vs NRP - What's different
 - Multi-facilities and providers
 - More heterogeneous
 - AuthN/AuthZ more complicated
 - Heavier HPC resources
 - More AI model team involvement
 - Public cloud partners



DOE IRI API Spec



IRI / IRO Client Toolkit



AmSC-IRO Manual



AmSC-IRO Intro Video