

# The Fornax Initiative: Cloud Computing with Astrophysics Data Jan. 2025

Tess Jaffe  
GSFC

Fornax Project Scientist



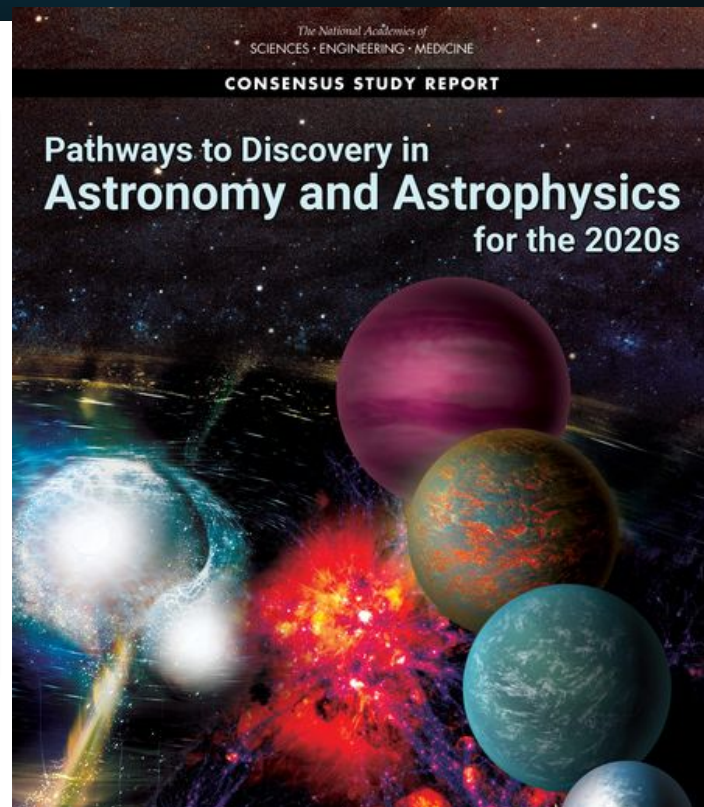
# The challenge



“The importance of joint analysis of observations from different facilities and wavelengths, and of sophisticated archiving with associated science platform tools, will grow dramatically over the next decade.”

The Astrophysics Division sees a common need in our overlapping communities for:

- multi-archive analyses on large data sets (“big data”);
- complex analyses that require a lot of compute (“big compute”);
- collaboration tools and tutorials for non-experts (“big community”).



# NASA Astro data in the cloud

(independent of Fornax)



Many NASA Astrophysics datasets are now hosted in Amazon's AWS with free access:

- [HEASARC](#) mirrors most data to AWS (including Swift, Fermi, Chandra, etc.)
- [IRSA](#) hosts in the cloud: Spitzer mosaic images; images and catalogs from WISE/NeoWISE/unWISE; and the OpenUniverse 2024 matched Rubin and Roman simulated skies.
  - (Euclid will release 2PB in FY26, and a further 6PB in FY28)
- [MAST](#) hosts data from GALEX and Kepler/K2, and public data from current missions Hubble, TESS and [JWST](#).
  - (Roman expects 1PB after the first 6 months (~FY27), then another ~2PB per year)
- For comparison, Rubin Observatory (NSF-DoE) will release roughly 1.5PB per year from ~FY26, with a final data release of 15PB after 10 years of operation

# NASA Astro data in the cloud

(independent of Fornax)



Many NASA Astrophysics datasets are now hosted in Amazon's AWS with free access:

- [HEASARC](#) mirrors most data to AWS (including Swift, Fermi, Chandra, etc.)
- [IRSA](#) hosts in the cloud: Spitzer mosaic images; images and catalogs from WISE/NeoWISE/unWISE; and the OpenUniverse 2024 matched Rubin and Roman simulated skies.
  - (Euclid will release 2PB in FY26, and a further 6PB in FY28)
- [MAST](#) hosts data from GALEX and Kepler/K2, and public data from current missions Hubble, TESS and [JWST](#).
  - (Roman expects 1PB after the first 6 months (~FY27) then another ~2PB per year)

- For fin **The Fornax Initiative is about how best to analyze these data (which are curated and served by the thematic archives.)** with a

# Fornax: compute in the cloud, proximate to cloud-held Astrophysics datasets

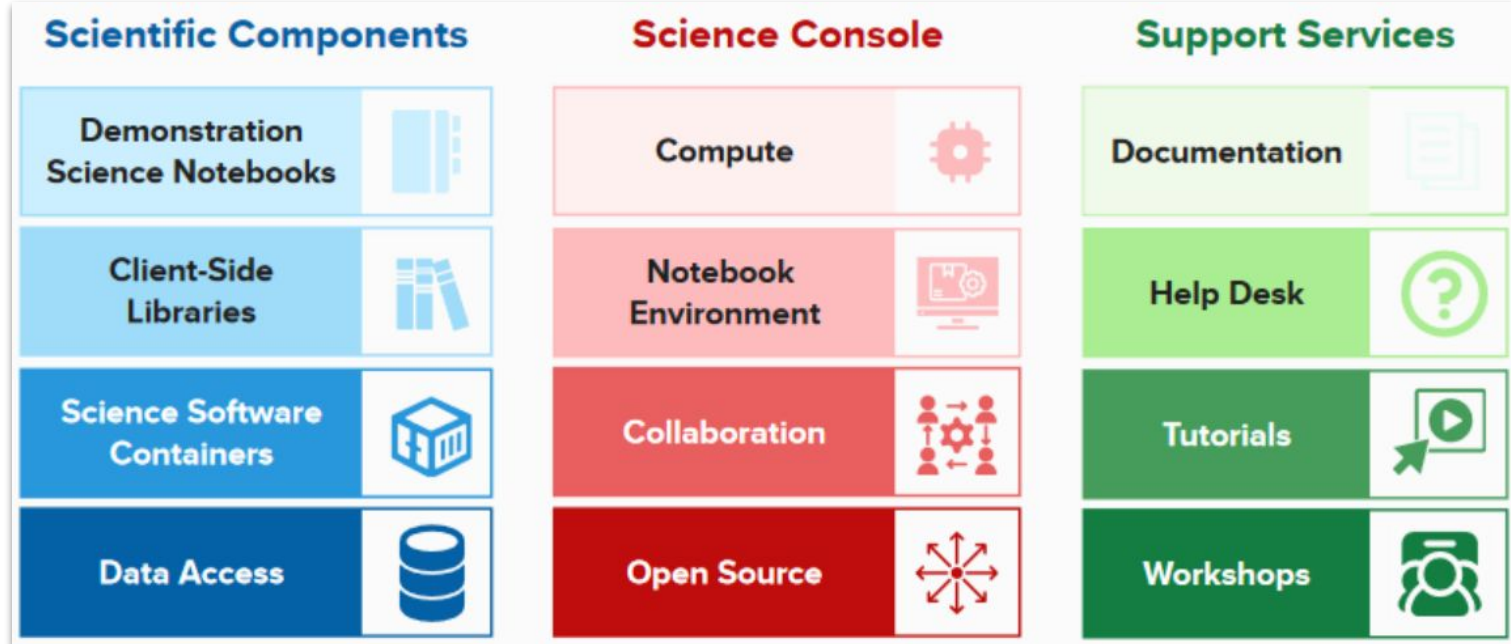


Fornax aims to provide science users with

- Access to a **fixed allocation of cloud resources** for storage, compute, etc, for free, through a standard web browser: no special hardware or software needed.
- **Pre-built astronomy analysis tools**, with notebooks that users can modify and run **proximate to NASA and non-NASA data in the cloud**;
- **Tutorial notebooks** and documentation that allow users unfamiliar with particular techniques to learn those techniques and use them to analyze data.
- Ability for users to **upload** (limited amounts of) their own data or code to run, and **export** results
- Ability to form **collaboration** groups that share notebooks, data files, etc, while keeping them private to your collaboration.

NASA Astrophysics is developing Fornax in the Amazon (AWS) cloud

# The three pillars of the Fornax Initiative

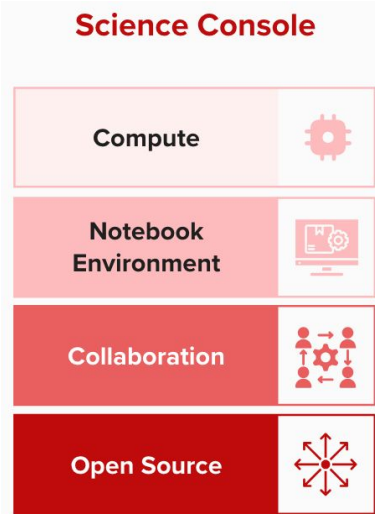


# The Fornax Initiative: Science Console



A web-based application that users log into for access to cloud computing, data storage, and interactive data analysis in JupyterLab.

- Common infrastructure elements.
- Collaborating with other platform projects.
- Using whatever already exists where possible.



- Open sourcing whatever we develop.
- Deployable elsewhere,
  - i.e. a university department could deploy their own Fornax Science Console on their own AWS account.

# Portal including a dashboard to view resources:



The screenshot displays a JupyterLab environment with a dashboard overlay. The dashboard provides a comprehensive overview of resource usage and system status. A sidebar menu on the left includes options for Dashboards, Home, Compute, Launch EC2s, JupyterHub, Documentation, Quickstart, FAQ, Help, and Community. A blue arrow points from the 'Dashboards' menu item to the dashboard window.

### Dashboard Metrics

Metric	Value	Unit	Additional Info
Total Spend	92.75	BU	Past 126 days
Credits Remaining	907.25	BU	239 days remaining
Compute Spend	22.37	BU	
Egress Spend	0.51	BU	
Storage Spend	65.07	BU	
IPv4 Spend	4.80	BU	
Total Egress	5.616	GB	
Storage Allocated	100	GiB	
Running EC2s	0		
Stopped EC2s	0		
Running Jupyter Servers	0		

The background shows a terminal window with system statistics and a file browser displaying the project structure, including notebooks like 'light\_curve\_classifier.md' and 'light\_curve\_generator.md'.







# The Fornax Initiative: Scientific Components



The astrophysics-specific elements required to enable science in the cloud, including Python notebooks that demonstrate access to cloud-hosted NASA mission data, curated astrophysics software environments, and cloud-native services to support common astronomy workflows.

## Scientific Components

Demonstration Science Notebooks	
Client-Side Libraries	
Science Software Containers	
Data Access	

All these components can be used *à la carte*:

- Notebooks are portable.
- Software is open source.
- Software environments will be in public container registries.
- Data can be accessed from anywhere.

# The Fornax Initiative: Scientific Components

Notebooks demonstrating real science use cases:



The screenshot displays a JupyterLab environment. On the left, a file browser shows a directory structure for 'fornax-demo-notebooks/light\_curves/'. The main area is split into three panes:

- Terminal 2:** Shows system statistics at 19:09:52. The load average is 4.99, 3.31, 1.48. Tasks: 17 total, 5 running, 12 sleeping, 0 stopped, 0 zombie. CPU usage: 98.7% (98.7% user, 1.2% system, 0.0% idle, 0.0% wait, 0.0% hardware, 0.0% software, 0.0% steal). Memory usage: 15524.5 MiB total, 1478.6 MiB free, 2536.7 MiB used, 11509.2 MiB buffer/cache. Swap: 0.0 MiB total, 0.0 MiB free, 0.0 MiB used, 12656.4 MiB available.
- multiband\_photometry.md:** Contains Python code for multiprocessing. It defines a function to calculate flux and uses a Pool to parallelize the calculation.
- ML\_AGNzoo.md:** Contains Python code for UMAP visualization. It defines 'first' and 'last' indices, plots 'Normalized Flux (mean r band)', and creates a figure with a scatter plot of 'mean brightness' vs 'embedding\_[:,0]'.

### 3) Learn the Manifold

Now we can train a UMAP with the processed data vectors above. Different choices for the number of neighbors, minimum distance and metric can be made and a parameter space can be explored. We show here our preferred combination given this data. We choose manhattan distance (also called the [L1 distance](#)) as it is optimal for the kind of grid we interpolated on, for instance we want the distance to not change if there are observations missing. Another metric appropriate for our purpose in time domain analysis is Dynamic Time Warping (DTW), which is insensitive to a shift in time. This is helpful as we interpolate the observations onto a grid starting from time 0 and when discussing variability we care less about when it happens and more about whether and how strong it happened. As the measurement of the DTW distance takes longer compared to the other metrics we show examples here with manhattan and only show one example exploring the parameter space including a DTW metric in the last cell of this notebook.

```
[ ]: plt.figure(figsize=(18,6))
      markersize=200
      mapper = umap.UMAP(n_neighbors=50,min_dist=0.9,metric='manhattan',r
ax1 = plt.subplot(1,3,2)
ax1.set_title(r'mean brightness',size=20)
cf = ax1.scatter(mapper.embedding_[:,0],mapper.embedding_[:,1],s=ma
plt.axis('off')
```

# The Fornax Initiative: Scientific Components



Python client for data search and retrieval:

```
pyvo.utils.download_file(record, 'aws')
```

Under the hood:

- Service layer returns *multiple* access options.
- Client layer selects the right access method for cloud versus on-prem.

Cloud-optimized large catalog cross matching (collaboration with [LINCC Frameworks](#)) to enable workflows like:

```
img = gaia
    .query("pm > 10")
    .crossmatch(ztf)
    .joint(ztf_sources)
    .for_each(varstar_classify)
    .query("pRRLy > 0.95")
    .skymap()
hp.mollview(img)
```

See <https://lsdb.readthedocs.io/>.

Or partial FITS reads out of S3 in astropy:

```
from astropy.io import fits

# URI of a 213 MB FITS file hosted in a free Amazon S3 cloud storage bucket
uri = "s3://stpubdata/hst/public/j8pu/j8pu0y010/j8pu0y010_drc.fits"

# Download the primary header
with fits.open(uri) as hdul:
    # Download a small cutout
    myslice = hdul[2].section[10:12, 20:22]
```

*The entire 213 MB file is never downloaded. Only the chunks you want are transferred.*

# The Fornax Initiative: Support Services



A program of engagement with the astronomical community, including a Helpdesk and training opportunities

User documentation under development at:

<https://fornax-navo.github.io/fornax-documentation/>

(We are working in an open source and transparent way!)

Workshops coming in the next few years to a meeting near you.

## Support Services

Documentation



Help Desk



Tutorials



Workshops



# Working in the cloud is different from working on your laptop!



- If you start your cloud session and then go to lunch before beginning a compute task, you are wasting resources – could be a lot!
- Downloading (egress) generally costs resources. The cloud stores huge volumes of data, so you easily burn up your resources by working inefficiently or downloading more than you need. Use cloud-optimized libraries for streaming data into your session, manage wisely what data you do move.
- Don't assume that a Jupyter notebook developed on your laptop will perform well in the cloud. Workflows requiring significant data I/O, memory management, or CPU parallelization are very different in the cloud. Our tutorial notebooks will help.

# Example and tutorial notebooks



Best places to get started:

- Working with TESS data in the cloud: [TESS intro using TIKE](#)
- [IRSA cloud access tutorial introduction](#) (Parquet basics using AllWISE – you can run this on a laptop)

Other notebooks:

- [Light Curve Generator](#), to run on Fornax or elsewhere:
  - Query unWISE data in Parquet format on AWS. 500k light curves in 4 hours (<< 100 day est. for original code to query FITS files at NERSC)
- [AllWISE Source Catalog Examples](#) on IRSA's github
- [5-minute Quick Start with LSDB/HATS](#) on lincc-frameworks github – this was run on Fornax as a demo for the IVOA meeting in November 2024
- Cross-match catalogs from ZTF and Pan-STARRS using LSDB (a useful package for large-catalog cross-match): [ZTF-Pan-STARRS X-match](#)

# Sign me up...?



- Fornax is still developing user management tools and security, so we are not yet open.
- Beta testing opportunities will be expanded later this year, stay tuned.
- Keep an eye on the website: <https://pcos.gsfc.nasa.gov/Fornax/> (or just search for “NASA Fornax”) for news.

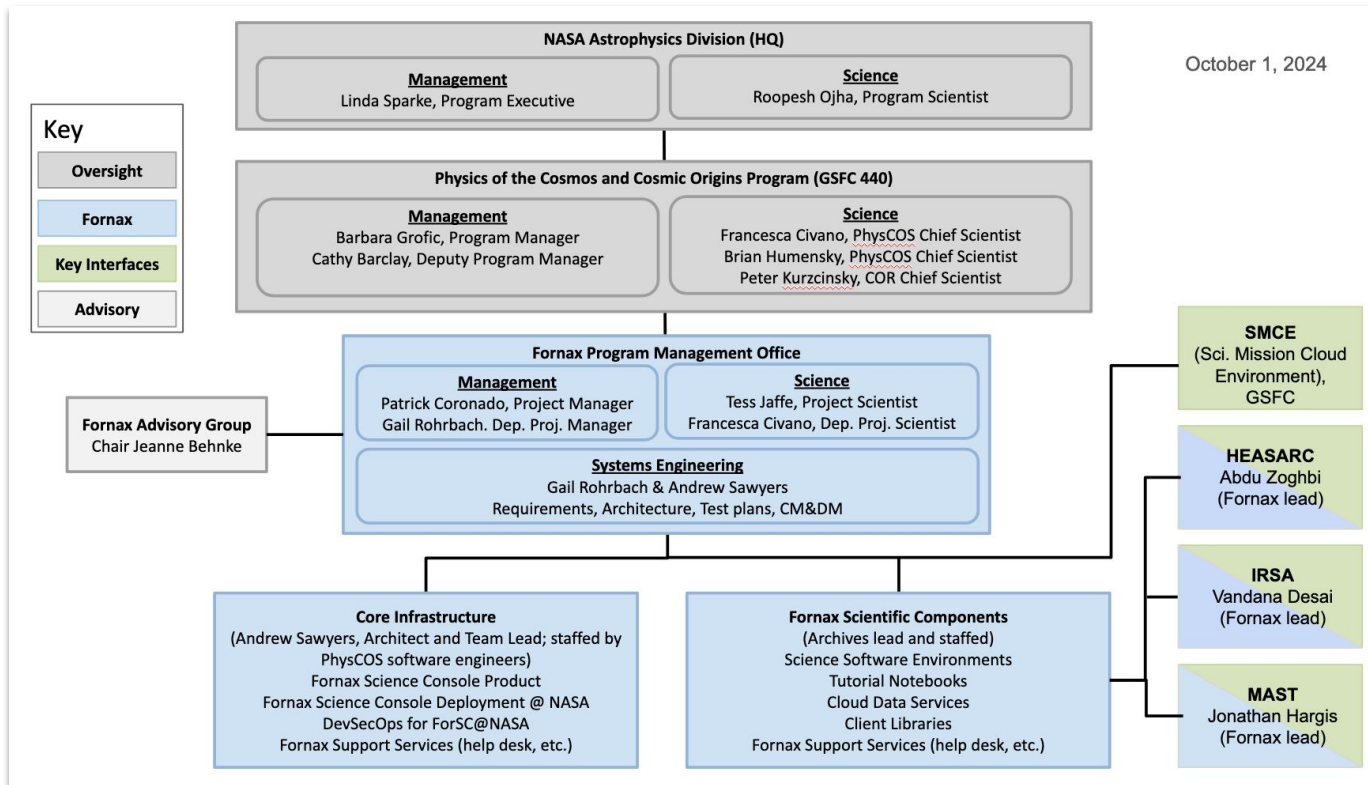
Dates	Phase
FY 22-23	Design
FY 24	Prototype
FY 25	Internal Beta: 10's of users
FY 26	External Beta: 100 users
FY27	Initial Operations: 100s of users
FY28	Full Operations: 1000 users or more

# Extra slides

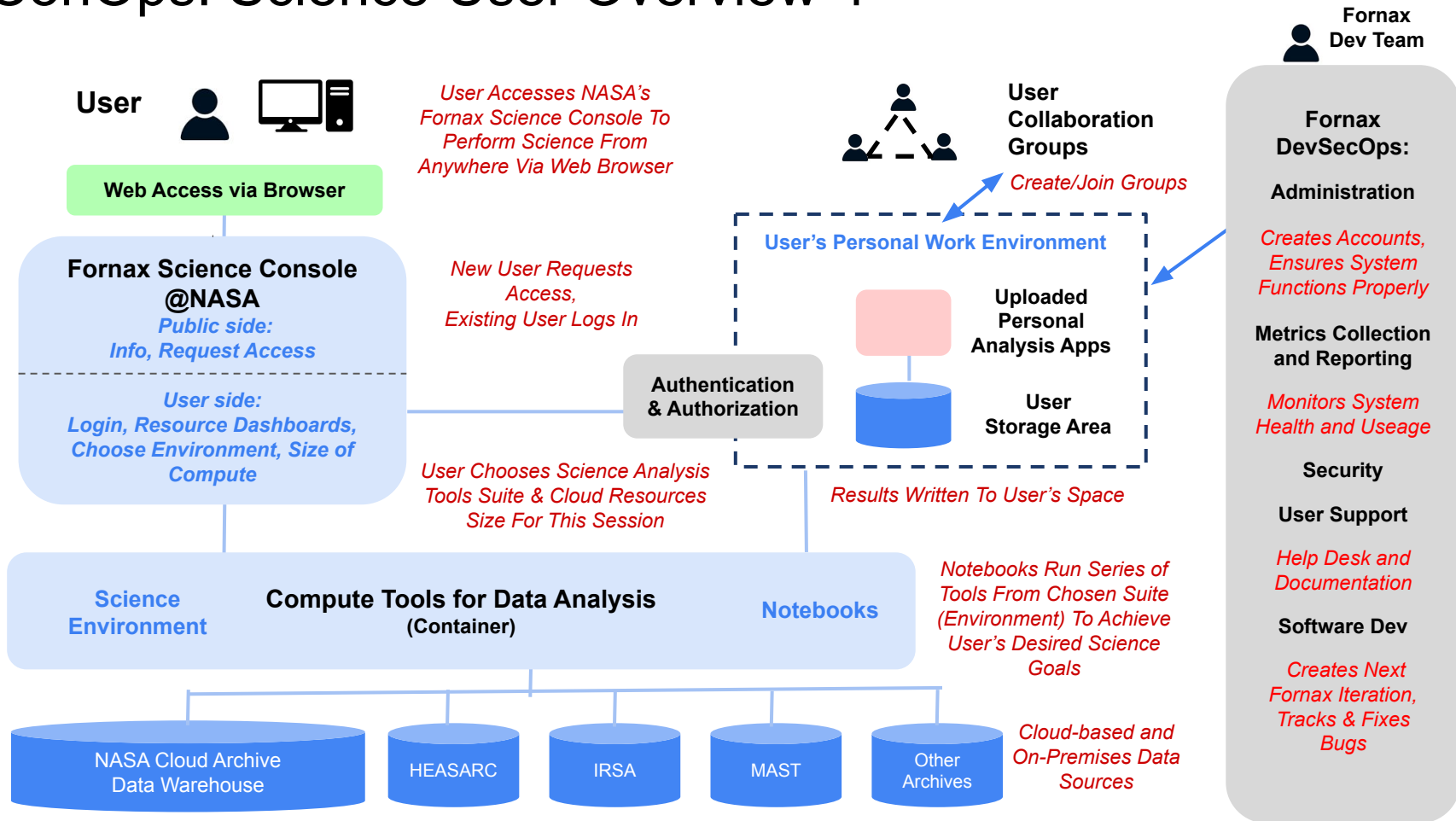




# Org chart: who's who

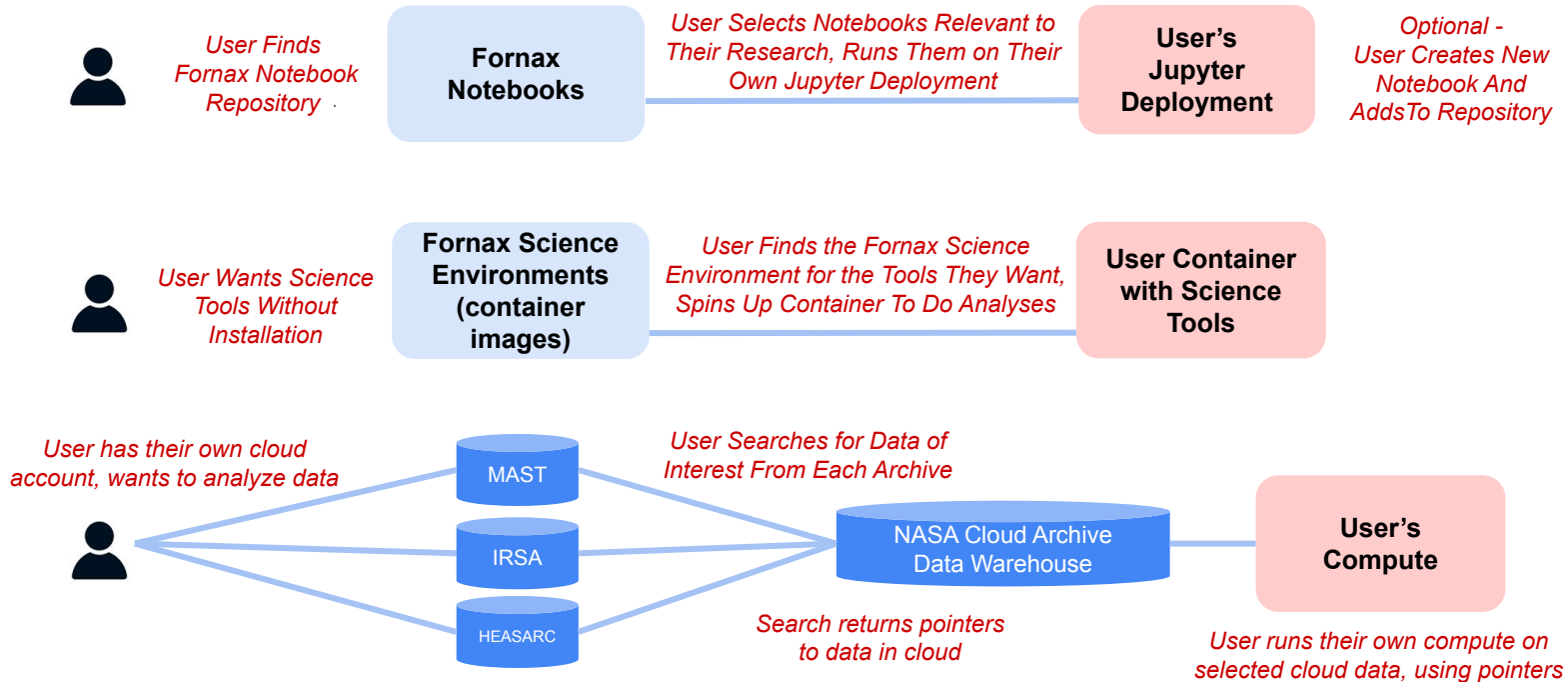


# ConOps: Science User Overview-1



# ConOps: Science User Overview-2

Many Fornax elements are publically available, and can be accessed without logging into the NASA Fornax Science Console. For example:



Or, user can access [AWS Marketplace: AllWISE Data | Wide-field Infrared Survey Explorer \(WISE\)](#)

# ConOps: Science User Overview-3

The Fornax Science Console is open source code. It can be modified and deployed by a user at their home institution, in cloud space provided by that institution, to do science analysis with their resources.

