

ACCESS Pegasus

A hosted scientific workflow system
part of ACCESS Support

Mats Rynge, USC/ISI

ACCESS Support Strategy



**Powerful Tools
& Workflows**



**Dynamic Knowledge
Base**



**Community Expertise
and Experience**

Powerful Tools & Workflows

OnDemand

■ INTEGRATED WEB-BASED INTERFACES

Schedule jobs, manage files, create remote visualizations and use a host of other valuable services.

Pegasus

■ AUTOMATED WORKFLOWS

Simplify complex data workflows on distributed computing resources, such as clusters, grids, and clouds.

Knowledge Base

Self-service resources reduce the learning curve



DOCS

RP guides, code
and best practices



LINKS

Provided and vetted
by the community



TICKETS

Answers build the
Knowledge Base



ASK.CI

Community
Q&A forum

Computational Science & Support Network

CSSN

Utilizing community expertise



MATCH Engagements

Assistance is available from CSSN experts matched to Researcher needs.

$$M_+ = R + S M$$

MATCHPlus

Researchers assisted by a Student and their Mentor

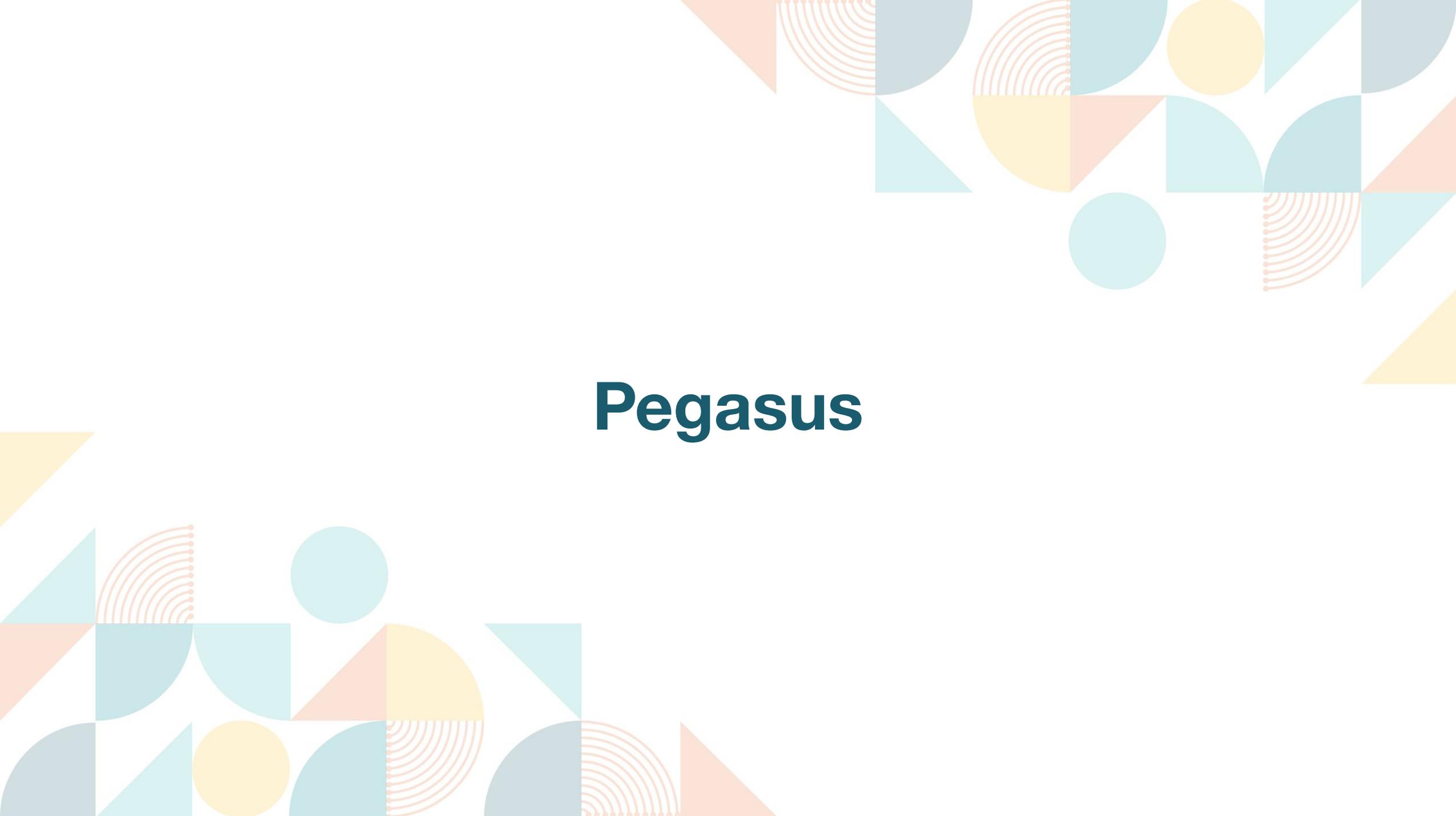
■ < 6 MONTHS

$$M_{++} = R + C$$

MATCHPremier

Researchers assisted by an expert Consultant

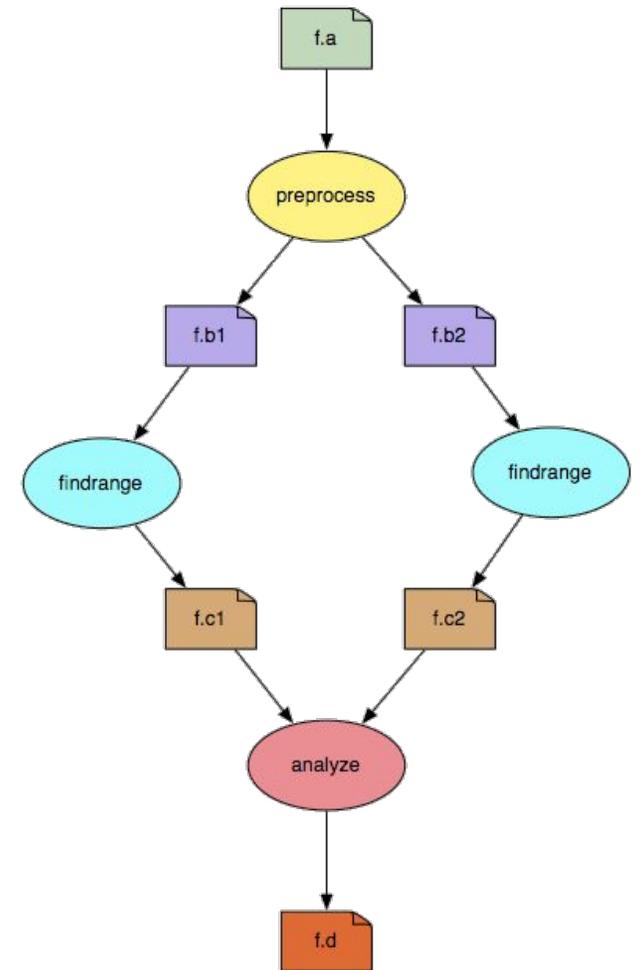
■ 6+ MONTHS

The image features decorative geometric patterns in the corners. The top-right and bottom-left corners contain clusters of shapes including triangles, circles, semi-circles, and concentric arcs in shades of teal, orange, and yellow. The word "Pegasus" is centered in a dark teal font.

Pegasus

Scientific Workflows

- An abstraction to express ensemble of complex computational operations
 - *Eg: retrieving data from remote storage services, executing applications, and transferring data products to designated storage sites*
- A workflow is represented as a directed acyclic graph (DAG)
 - *Nodes: tasks or jobs to be executed*
 - *Edges: depend between the tasks*
- Have a monolithic application/experiment?
 - *Find the inherent DAG structure in your application to convert into a workflow*



Key Pegasus Concepts

▲ Pegasus WMS == Pegasus planner (mapper) + DAGMan workflow engine + HTCondor scheduler/broker

- Pegasus maps workflows to infrastructure
- DAGMan manages dependencies and reliability
- HTCondor is used as a broker to interface with different schedulers

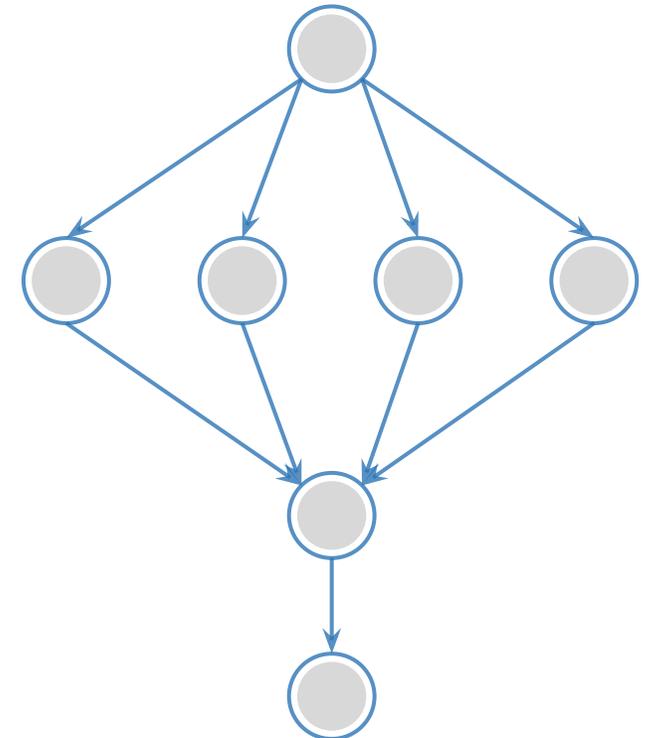
▲ Workflows are DAGs

- Nodes: jobs, edges: dependencies
- No while loops, no conditional branches
- Jobs are standalone executables

▲ Planning occurs ahead of execution

▲ Planning converts an abstract workflow into a concrete, executable workflow

- Planner is like a compiler

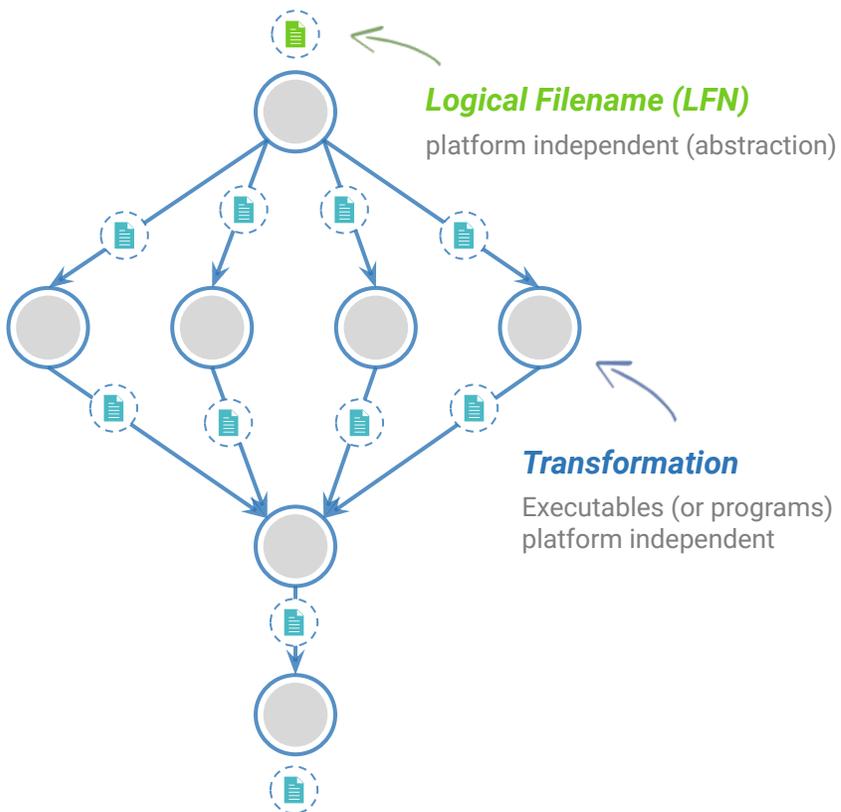


Input Workflow Specification **YAML formatted**

Portable Description

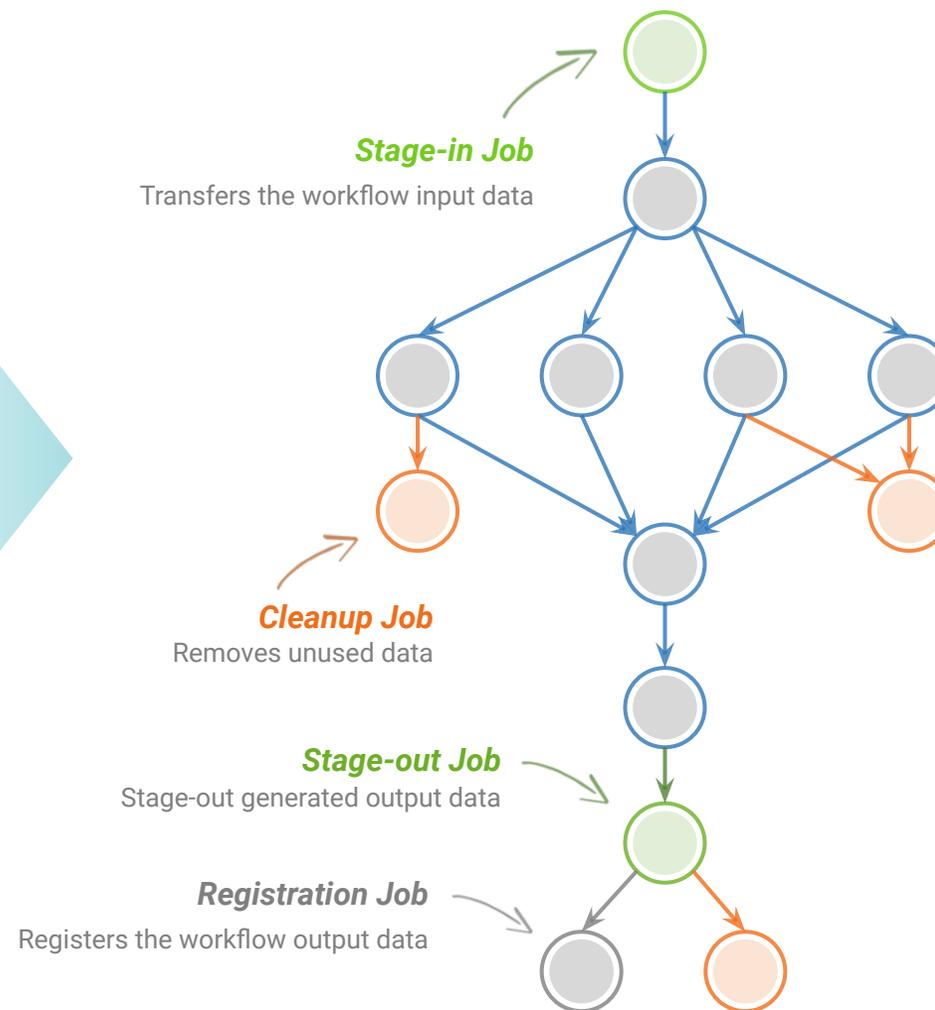
Users do not worry about low level execution details

ABSTRACT WORKFLOW



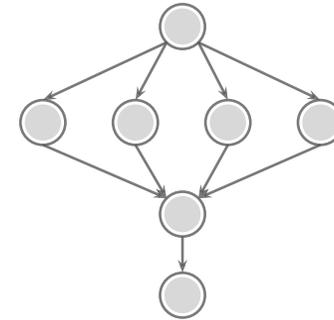
directed-acyclic graphs

Output Workflow

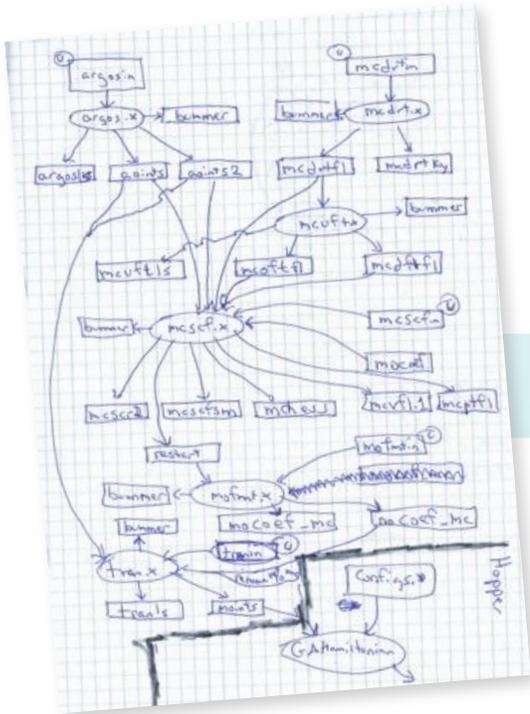


EXECUTABLE WORKFLOW

Pegasus provides tools to generate the Abstract Workflow



Abstract Workflow



```

#!/usr/bin/env python3

import os
import logging
from pathlib import Path
from argparse import ArgumentParser

logging.basicConfig(level=logging.DEBUG)

# --- Import Pegasus API -----
from Pegasus.api import *

# --- Create Abstract Workflow -----
wf = Workflow("pipeline")

webpage = File("pegasus.html")

# --- Create Parent Job -----
curl_job = (
    Job("curl")
    .add_args("-o", webpage, "http://pegasus.isi.edu")
    .add_outputs(webpage, stage_out=False, register_replica=False)
)

count = File("count.txt")

# --- Create Dependent Job -----
wc_job = (
    Job("wc")
    .add_args("-l", webpage)
    .add_inputs(webpage)
    .set_stdout(count, stage_out=True, register_replica=True)
)

# --- Add jobs to the Abstract Workflow -----
wf.add_jobs(curl_job, wc_job)

# --- Add control flow dependency -----
wf.add_dependency(wc_job, parents=[curl_job])

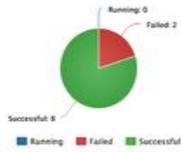
# --- Write out the Abstract Workflow -----
wf.write()
  
```



```

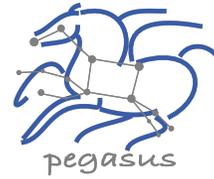
x-pegasus:
  apiLang: python
  createdBy: vahi
  createdOn: 11-19-20T14:57:58Z
  pegasus: '5.0'
  name: pipeline
  jobs:
  - type: job
    name: curl
    id: ID0000001
    arguments:
    - -o
    - pegasus.html
    - http://pegasus.isi.edu
    uses:
    - lfn: pegasus.html
      type: output
      stageOut: false
      registerReplica: false
  - type: job
    name: wc
    id: ID0000002
    stdout: count.txt
    arguments:
    - -l
    - pegasus.html
    uses:
    - lfn: count.txt
      type: output
      stageOut: true
      registerReplica: true
    - lfn: pegasus.html
      type: input
  jobDependencies:
  - id: ID0000001
    children:
    - ID0000002
  
```

YAML Formatted



Show results for all

Workflow Label	Submit Host	Submit Directory	State	Submitted On
split	workflow.isi.edu	/ifs/ccg3/ccg/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0006	Running	Fri, 23 Oct 2015 16:04:00
split	workflow.isi.edu	/ifs/ccg3/ccg/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0004	Failed	Fri, 23 Oct 2015 15:56:01
diamond	workflow.isi.edu	/ifs/ccg3/ccg/home/pegtrain01/examples/diamond/pegtrain01/pegasus/diamond/run0002	Successful	Fri, 23 Oct 2015 15:50:17
split	workflow.isi.edu	/ifs/ccg3/ccg/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0003	Failed	Fri, 23 Oct 2015 15:41:15
split	workflow.isi.edu	/ifs/ccg3/ccg/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0002	Successful	Fri, 23 Oct 2015 15:04:44
process	workflow.isi.edu	/ifs/ccg3/ccg/home/pegtrain01/examples/process/pegtrain01/pegasus/process/run0001	Successful	Fri, 23 Oct 2015 15:00:36
pipeline	workflow.isi.edu	/ifs/ccg3/ccg/home/pegtrain01/examples/pipeline/pegtrain01/pegasus/pipeline/run0001	Successful	Fri, 23 Oct 2015 15:00:26
merge	workflow.isi.edu	/ifs/ccg3/ccg/home/pegtrain01/examples/merge/pegtrain01/pegasus/merge/run0001	Successful	Fri, 23 Oct 2015 15:00:15



PEGASUS DASHBOARD

web interface for monitoring and debugging workflows

Statistics

Workflow Wall Time	12 mins 23 secs
Workflow Cumulative Job Wall Time	9 mins 34 secs
Cumulative Job Walltime as seen from Submit Side	9 mins 35 secs
Workflow Cumulative Badput Time	9 mins 23 secs
Cumulative Job Badput Walltime as seen from Submit Side	9 mins 20 secs
Workflow Retries	1

Workflow Statistics

Type	Succeeded	Failed	Incomplete	Total	Retries	Total + Retries
Tasks	5	0	0	5	0	5
Jobs	16	0	0	16	2	18
Sub Workflows	0	0	0	0	0	0

Type	Succeeded	Failed	Incomplete	Total	Retries	Total + Retries
Tasks	5	0	0	5	0	5
Jobs	16	0	0	16	2	18
Sub Workflows	0	0	0	0	0	0

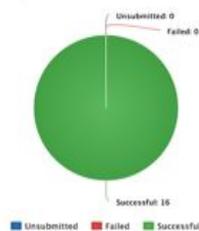
Real-time **monitoring** of workflow executions. It shows the **status** of the workflows and jobs, job **characteristics, statistics** and **performance** metrics.

Provenance data is stored into a relational database.

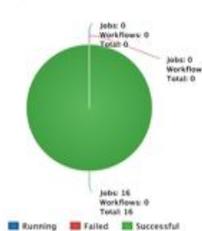
Workflow Details

Label	split
Type	root-wf
Progress	Successful
Submit Host	workflow.isi.edu
User	pegtrain01
Submit Directory	/ifs/ccg3/ccg/home/pegtrain01/examples/split/split/run0002
DAGMan Out File	split-0.dag.dagman.out
Wall Time	12 mins 23 secs
Cumulative Wall Time	9 mins 34 secs

Job Status (Entire Workflow)



Job Status (Per Workflow)



Real-time Monitoring

Reporting

Debugging

Troubleshooting

RESTful API

command-line...

```
$ pegasus-status pegasus/examples/split/run0001
STAT IN_STATE JOB
Run 00:39 split-0 (/home/pegasus/examples/split/run0001)
Idle 00:03 └─split_ID0000001
Summary: 2 Condor jobs total (I:1 R:1)

UNRDY READY PRE IN_Q POST DONE FAIL %DONE STATE DAGNAME
14      0      0      1      0      2      0      11.8 Running *split-0.dag
```

```
$ pegasus-analyzer pegasus/examples/split/run0001
pegasus-analyzer: initializing...

*****Summary*****

Total jobs : 7 (100.00%)
# jobs succeeded : 7 (100.00%)
# jobs failed : 0 (0.00%)
# jobs unsubmitted : 0 (0.00%)
```

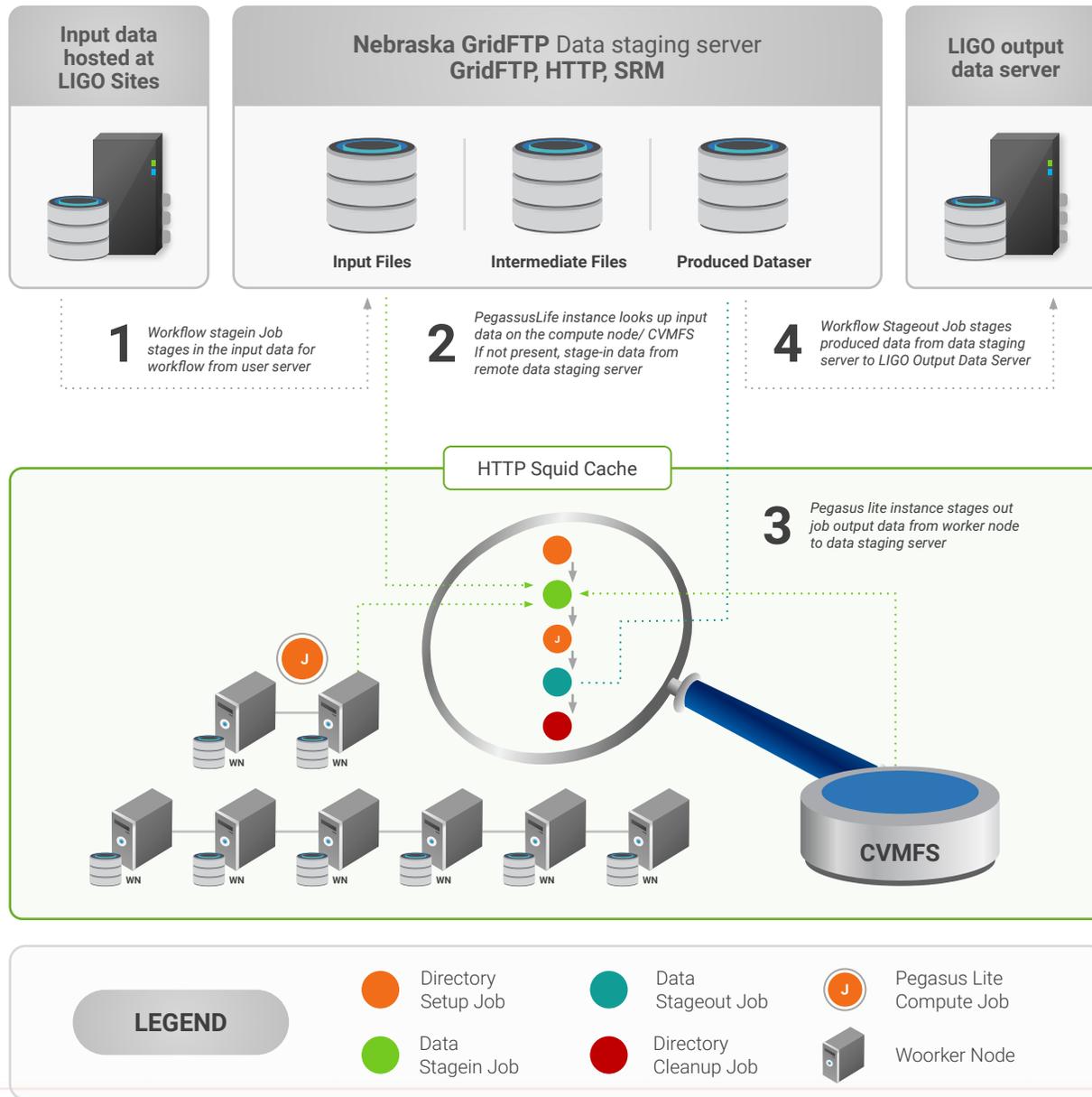
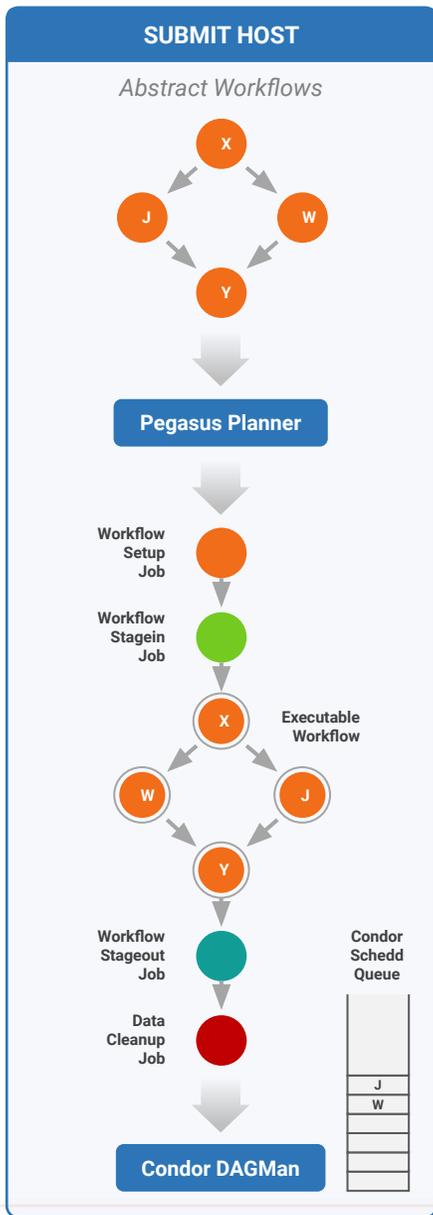
```
$ pegasus-statistics -s all pegasus/examples/split/run0001
-----
Type           Succeeded Failed Incomplete Total Retries Total+Retries
Tasks           5         0         0         5         0         5
Jobs            17        0         0        17         0        17
Sub-Workflows   0         0         0         0         0         0
-----

Workflow wall time : 2 mins, 6 secs
Workflow cumulative job wall time : 38 secs
Cumulative job wall time as seen from submit side : 42 secs
Workflow cumulative job badput wall time :
Cumulative job badput wall time as seen from submit side :
```

**Provenance Data
can be Summarized
Pegasus-Statistics
or
Used for Debugging
Pegasus-Analyzer**

The page features decorative geometric patterns in the corners. The top-right and bottom-left corners contain clusters of shapes including triangles, circles, semi-circles, and concentric arcs in shades of teal, orange, and yellow. The text 'Success Stories' is centered in a dark teal font.

Success Stories



Data Flow for LIGO Pegasus Workflows in OSG

Advanced LIGO Laser Interferometer Gravitational Wave Observatory



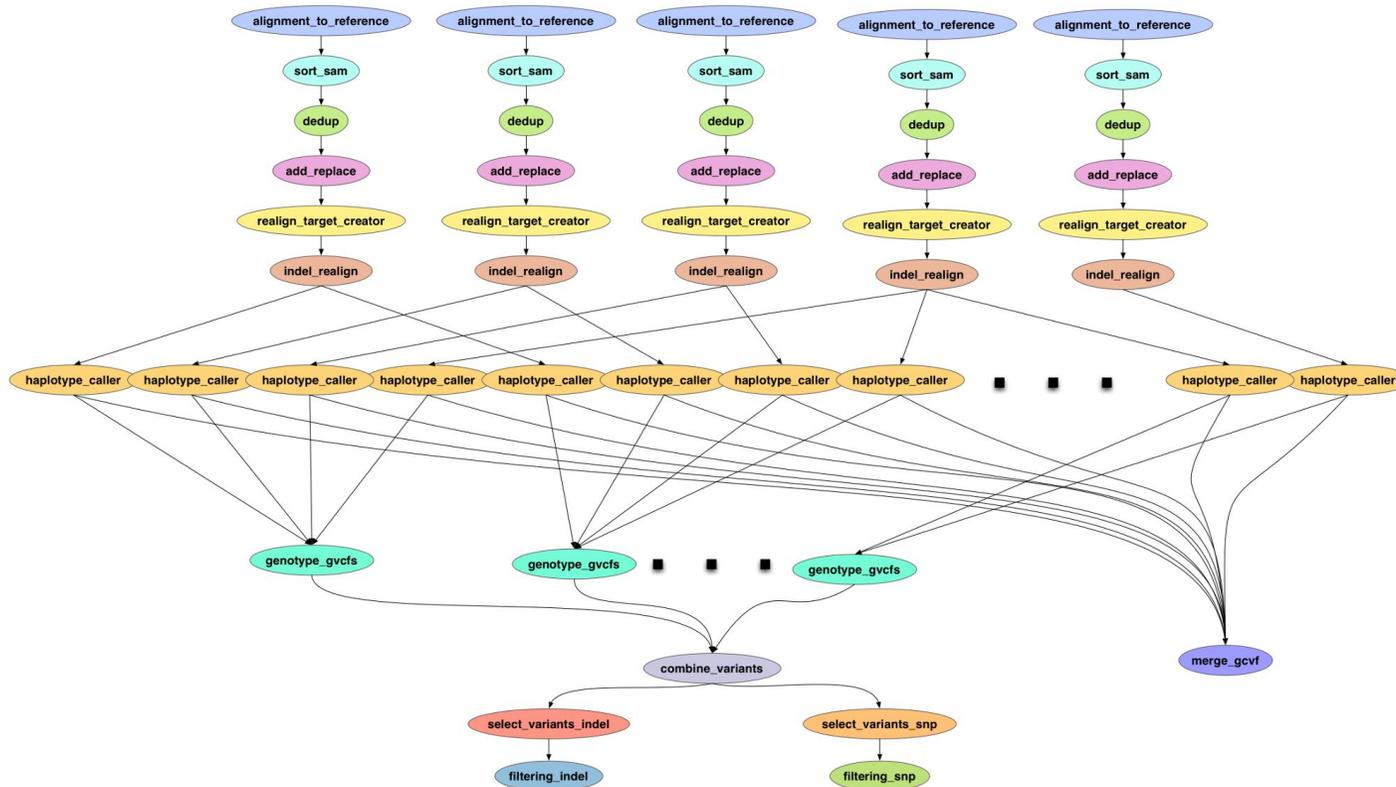
60,000 Compute Tasks
Input Data: 5000 files (10GB total)
Output Data: 60,000 files (60GB total)
 Processed Data: 725 GB

Executed on **LIGO Data Grid, EGI, Open Science Grid and XSEDE**

SoyKB

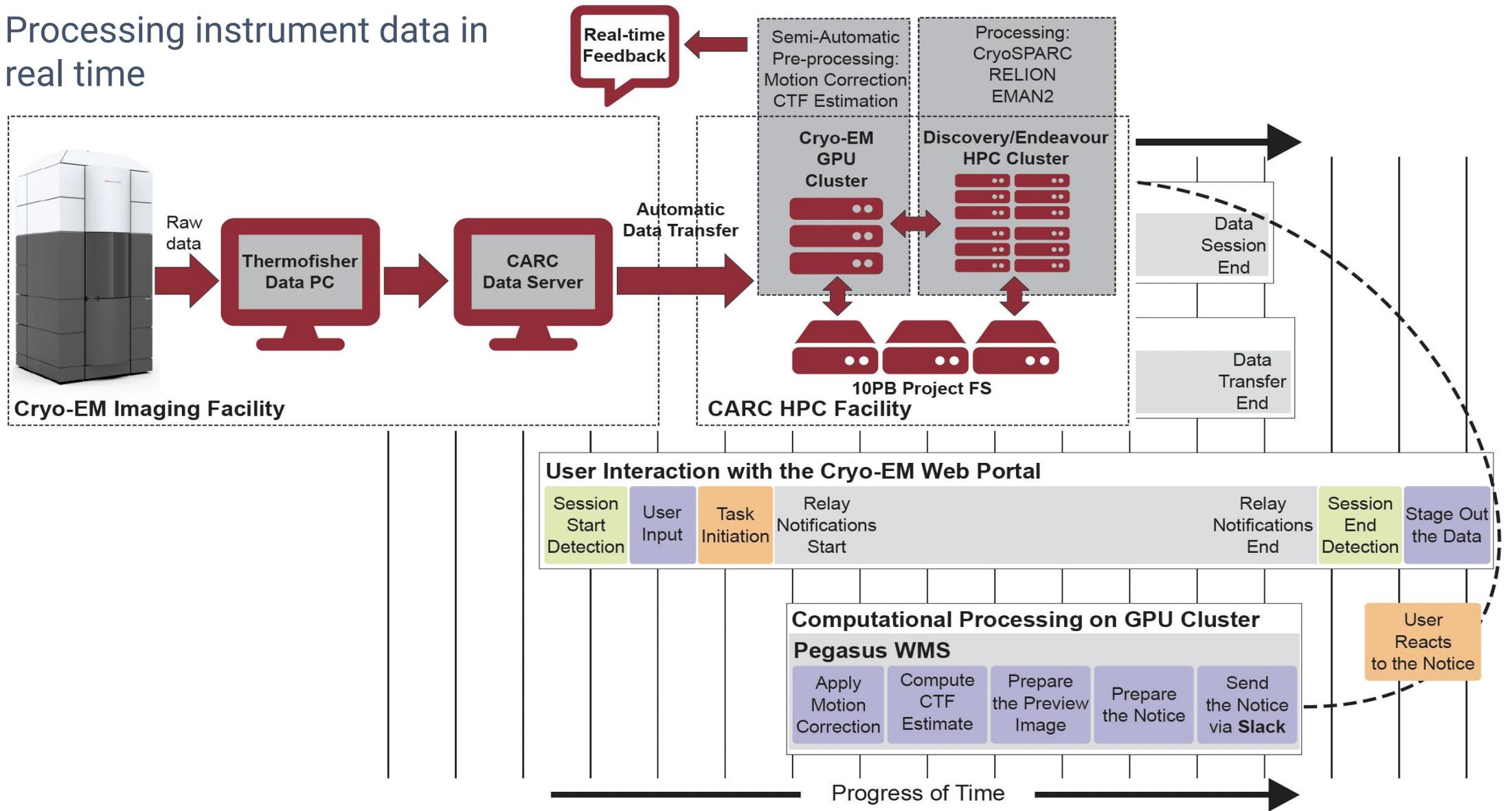
PI: Dong Xu

Trupti Joshi, Saad Kahn, Yang Liu, Juexin Wang, Badu Valliyodan, Jiaojiao Wang



Task	Base Code	Cores (Threads)	Memory (GB)
Alignment_to_reference	BWA	7	8
Sort_sam	Picard	1	21
Dedup	Picard	1	21
Add_replace	Picard	1	21
Realign_target_creator	GATK	15	10
Indel_realign	GATK	1	10
Haplotype_caller	GATK	1	3
Genotype_gvcfs	GATK	1	10
Merge_gvcf	GATK	10	20
Combine_variants	GATK	1	10
Select_variants	GATK	14	10
Filtering	GATK	1	10

Processing instrument data in real time



The page features decorative geometric patterns in the corners. The top-right and bottom-left corners contain clusters of shapes including triangles, circles, and semi-circles in shades of teal, yellow, and orange. Some shapes are filled with concentric lines. The text is centered in the white space between these patterns.

ACCESS Pegasus

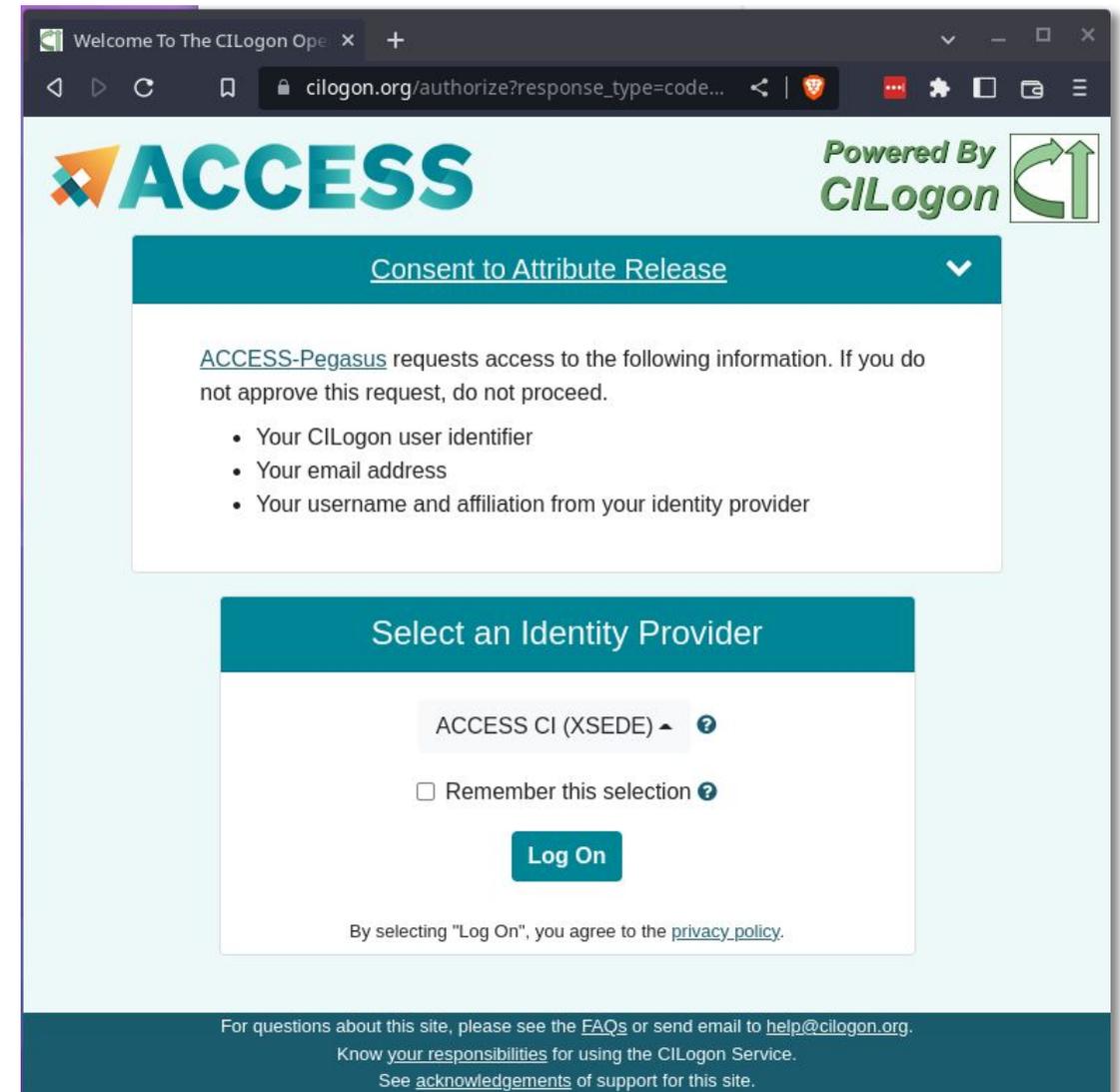
<https://support.access-ci.org/pegasus>

Prepare Logging In

CILogin with your ACCESS ID and institutional login

- <https://access.pegasus.isi.edu>

All registered ACCESS users with an active allocation automatically have access

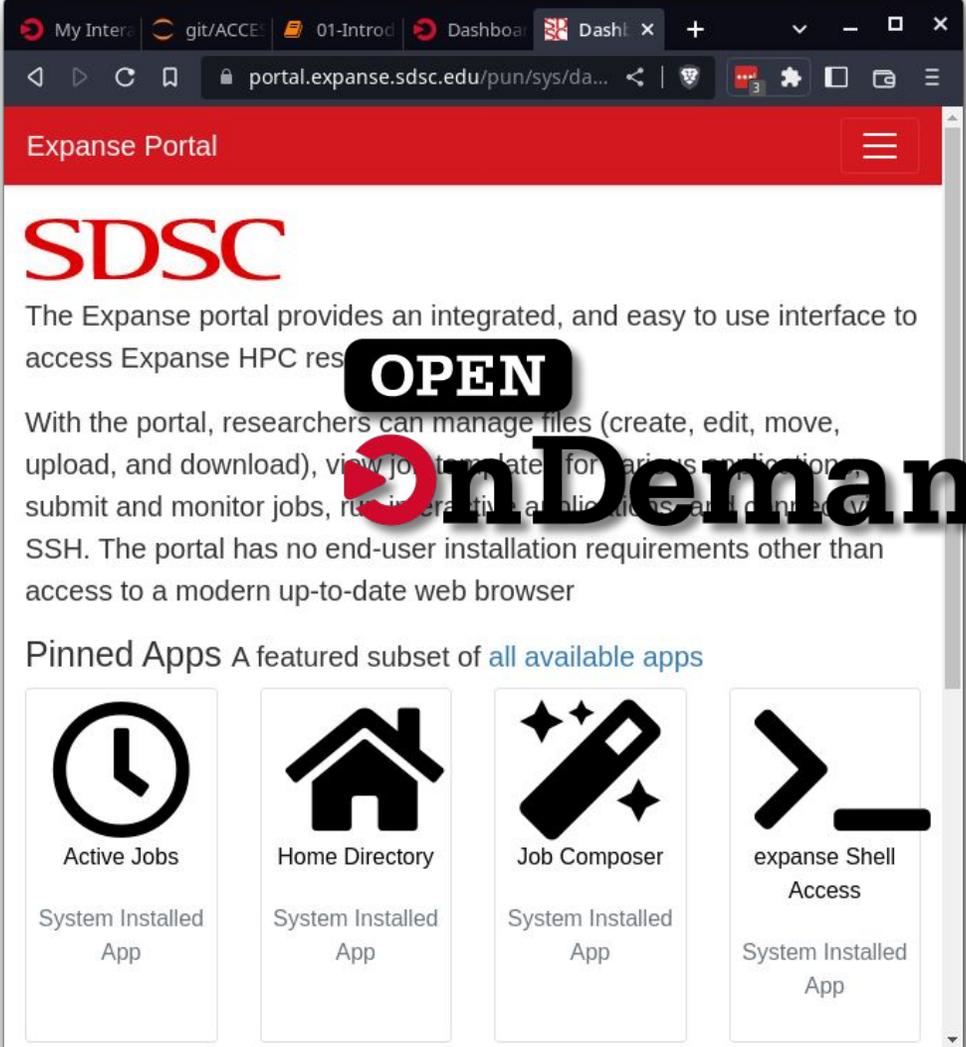


The screenshot shows a web browser window with the URL `cilogon.org/authorize?response_type=code...`. The page features the ACCESS logo and the text "Powered By CILogon". A teal header bar contains the text "Consent to Attribute Release". Below this, a white box contains the following text: "ACCESS-Pegasus requests access to the following information. If you do not approve this request, do not proceed." followed by a bulleted list: "Your CILogon user identifier", "Your email address", and "Your username and affiliation from your identity provider". Below the consent box is another teal header bar that says "Select an Identity Provider". Underneath, there is a dropdown menu showing "ACCESS CI (XSEDE)", a checkbox for "Remember this selection", and a teal "Log On" button. At the bottom of the white box, it says "By selecting 'Log On', you agree to the [privacy policy](#)." The footer of the page contains links for "FAQs", "help@cilogon.org", "responsibilities", and "acknowledgements".

Prepare Setting Up Resources

One time setup

Use [Open OnDemand instances](#)
at resource providers to install
ssh keys, and determine local
allocation id



The screenshot shows a web browser window displaying the SDSC Expanse Portal. The browser's address bar shows the URL `portal.expance.sdsc.edu/pun/sys/da...`. The page header is red with the text "Expanse Portal" and a menu icon. Below the header, the "SDSC" logo is prominently displayed in red. A paragraph of text describes the portal's purpose: "The Expanse portal provides an integrated, and easy to use interface to access Expanse HPC res...". A large, semi-transparent "OPEN OnDemand" watermark is overlaid on the text. Below this, another paragraph explains the portal's features: "With the portal, researchers can manage files (create, edit, move, upload, and download), view job templates for various applications, submit and monitor jobs, run interactively, applications, and connect via SSH. The portal has no end-user installation requirements other than access to a modern up-to-date web browser". At the bottom, a section titled "Pinned Apps" lists four applications: "Active Jobs", "Home Directory", "Job Composer", and "expanse Shell Access". Each application card includes an icon, the name, and the status "System Installed App".

My Interactive Sessions - git/ACCESS-Pegasus-Exa 01-Introduction-API

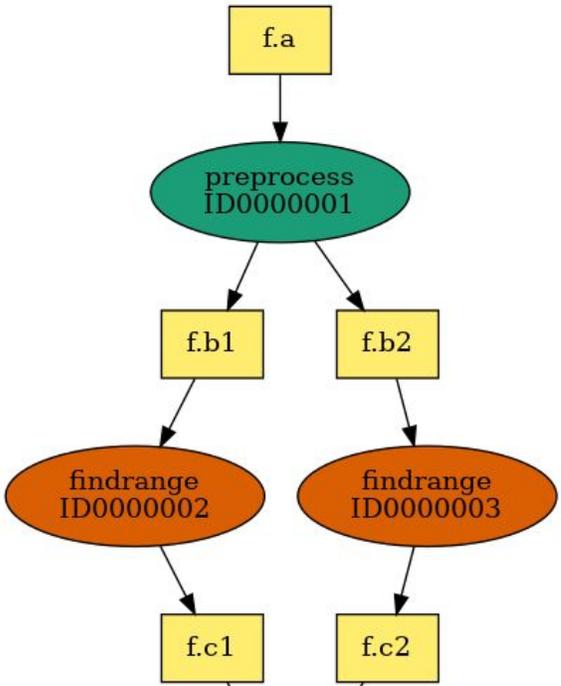
access.pegasus.isi.edu/node/match.js2local...

jupyter 01-Introduction-API (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

```
In [14]: # view rendered workflow
from IPython.display import Image
Image(filename='graph.png')
```

Out[14]:



```
graph TD
    fa[f.a] --> preprocess(preprocess ID0000001)
    preprocess --> fb1[f.b1]
    preprocess --> fb2[f.b2]
    fb1 --> findrange2(findrange ID0000002)
    fb2 --> findrange3(findrange ID0000003)
    findrange2 --> fc1[f.c1]
    findrange3 --> fc2[f.c2]
```

Step 1

Designing Workflow

Pegasus API in Jupyter Notebook

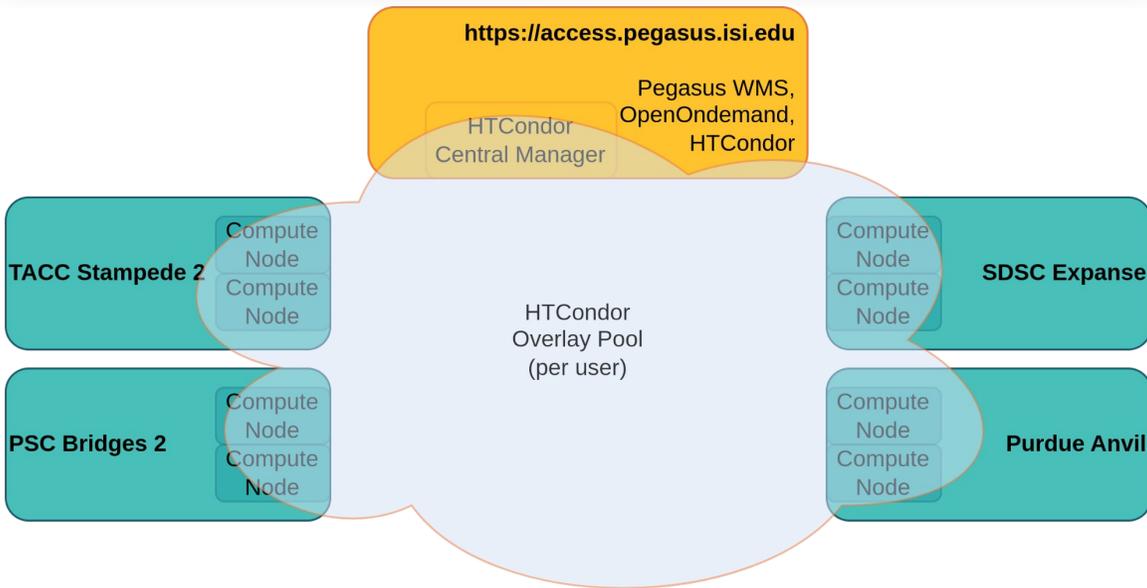
Fully hosted environment, based on Open OnDemand

```
Dashboard ryngex git/ACCE 01-Intro Dashboard +
access.pegasus.isi.edu/pun/sys/she...
Host: localhost Themes: Default
[ryngex@access ~]$ htcondor annex annex
--nodes 1 --lifetime 86400 --project
ddm160003 $USER standard@anvil
```

Step 2

Provision Resources

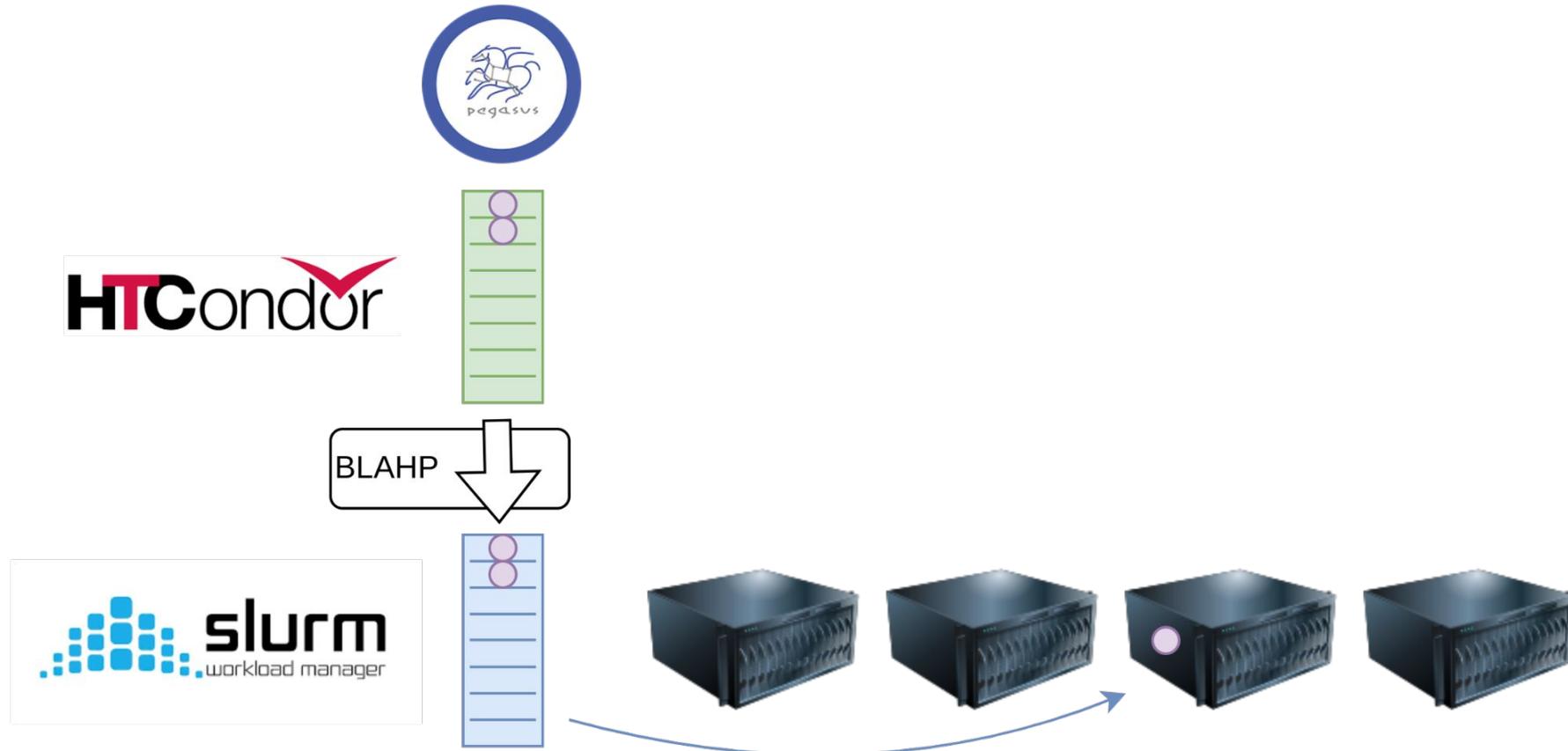
Use the **HTCondor Annex** tool to dynamically bring in compute nodes from one or more resource providers



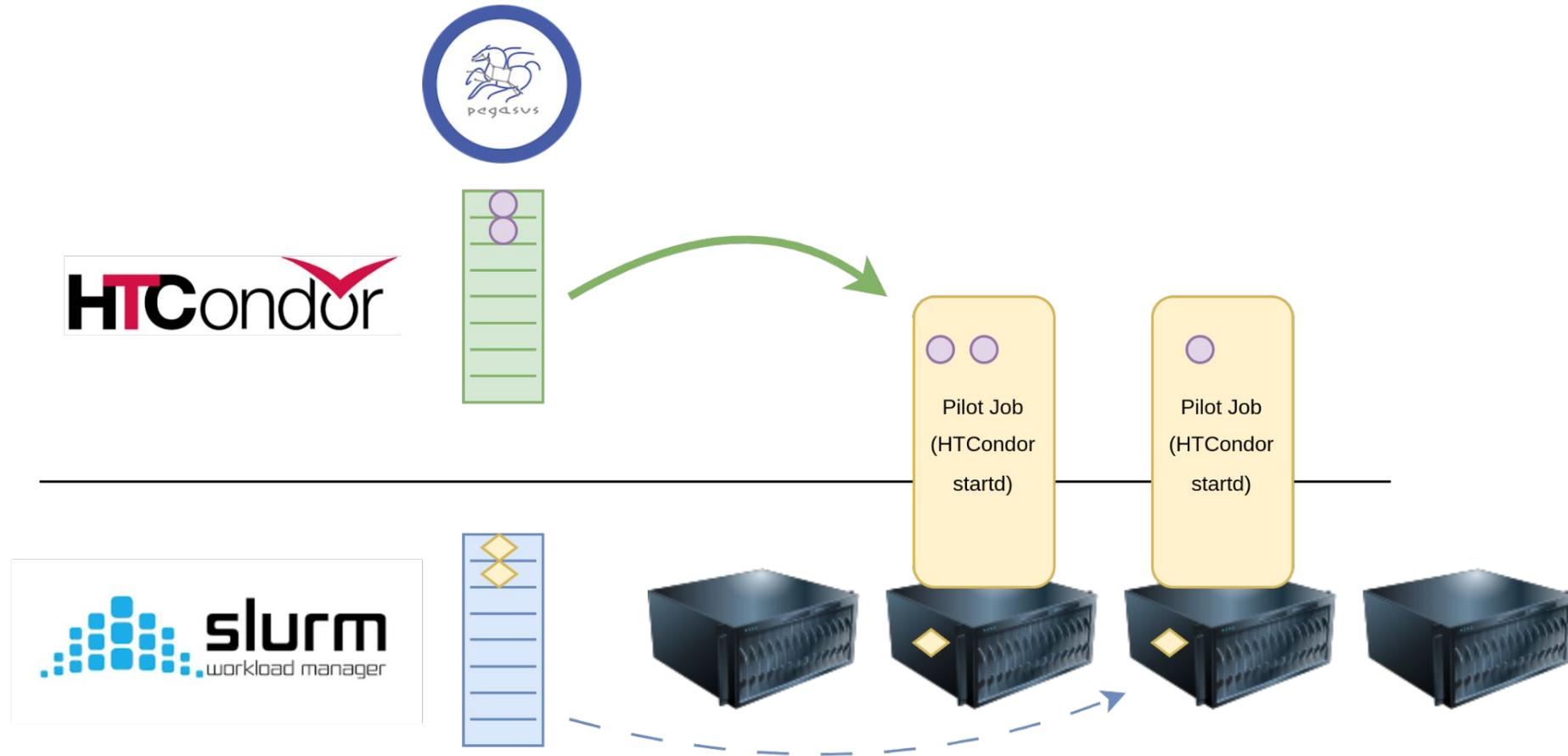
HTCondor Annex / Pilot Jobs

- **A pilot can run multiple user jobs** - it stays active until no more user jobs are available or until end of life has been reached, whichever comes first.
- **A pilot is partitionable** - job slots will dynamically be created based on the resource requirements in the user jobs. This means you can fit multiple user jobs on a compute node at the same time.
- **A pilot will only run jobs for the user who started it.**

HTCondor with BLAHP translation layer



HTCondor Pilot Jobs

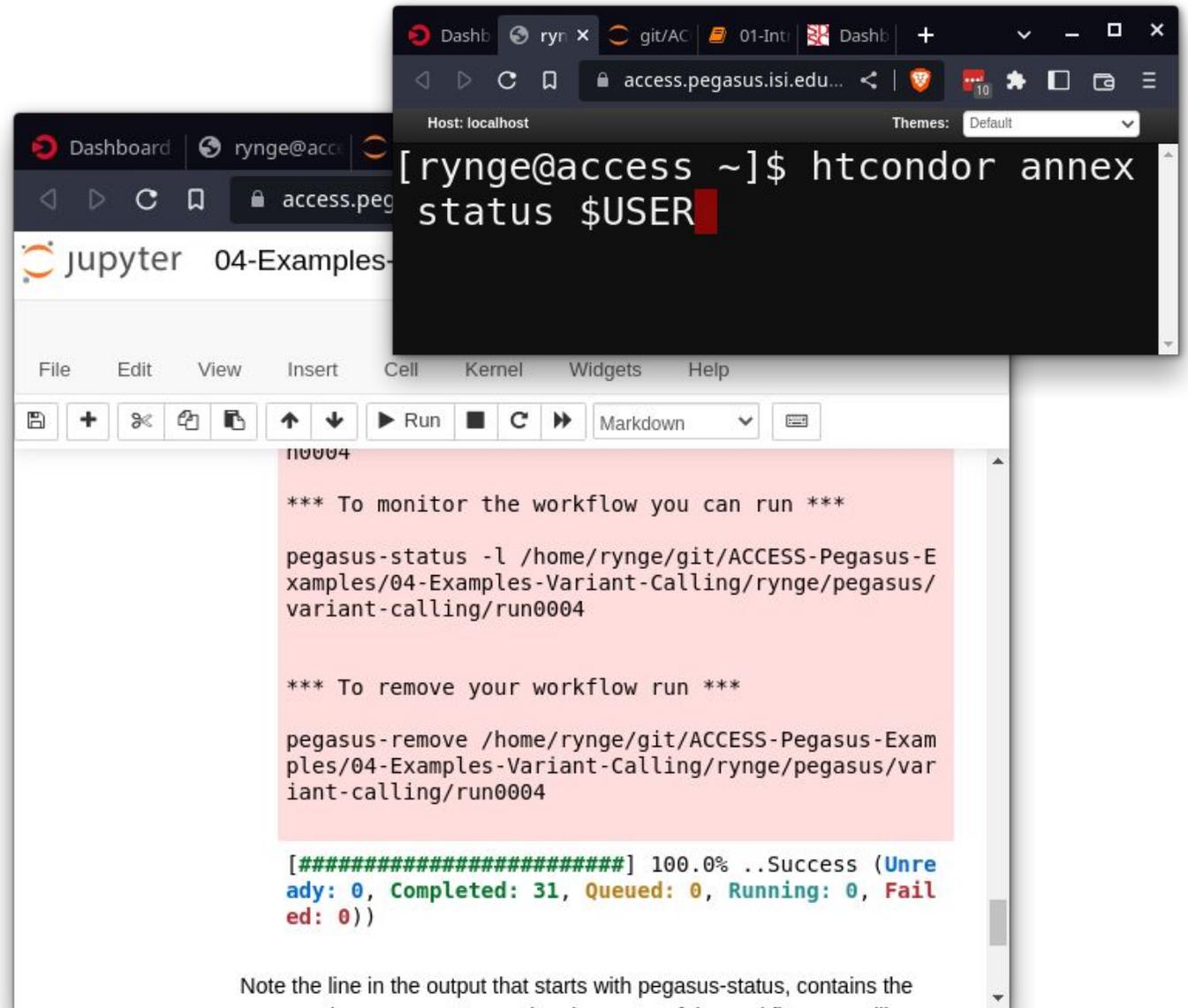


Step 3

Monitoring Workflow and Resources

Workflows can be monitored from within the Jupyter notebook, or via command line

HTCondor Annex can be monitored on the command line



The image shows a Jupyter notebook interface with a terminal window overlaid. The terminal window displays the following commands and output:

```
[rynge@access ~]$ htcondor annex status $USER
```

The Jupyter notebook cell contains the following text:

```
run0004
```

```
*** To monitor the workflow you can run ***
```

```
pegasus-status -l /home/rynge/git/ACCESS-Pegasus-Examples/04-Examples-Variant-Calling/rynge/pegasus/variant-calling/run0004
```

```
*** To remove your workflow run ***
```

```
pegasus-remove /home/rynge/git/ACCESS-Pegasus-Examples/04-Examples-Variant-Calling/rynge/pegasus/variant-calling/run0004
```

```
[#####] 100.0% ..Success (Unready: 0, Completed: 31, Queued: 0, Running: 0, Failed: 0))
```

Note the line in the output that starts with pegasus-status, contains the

Try it out!

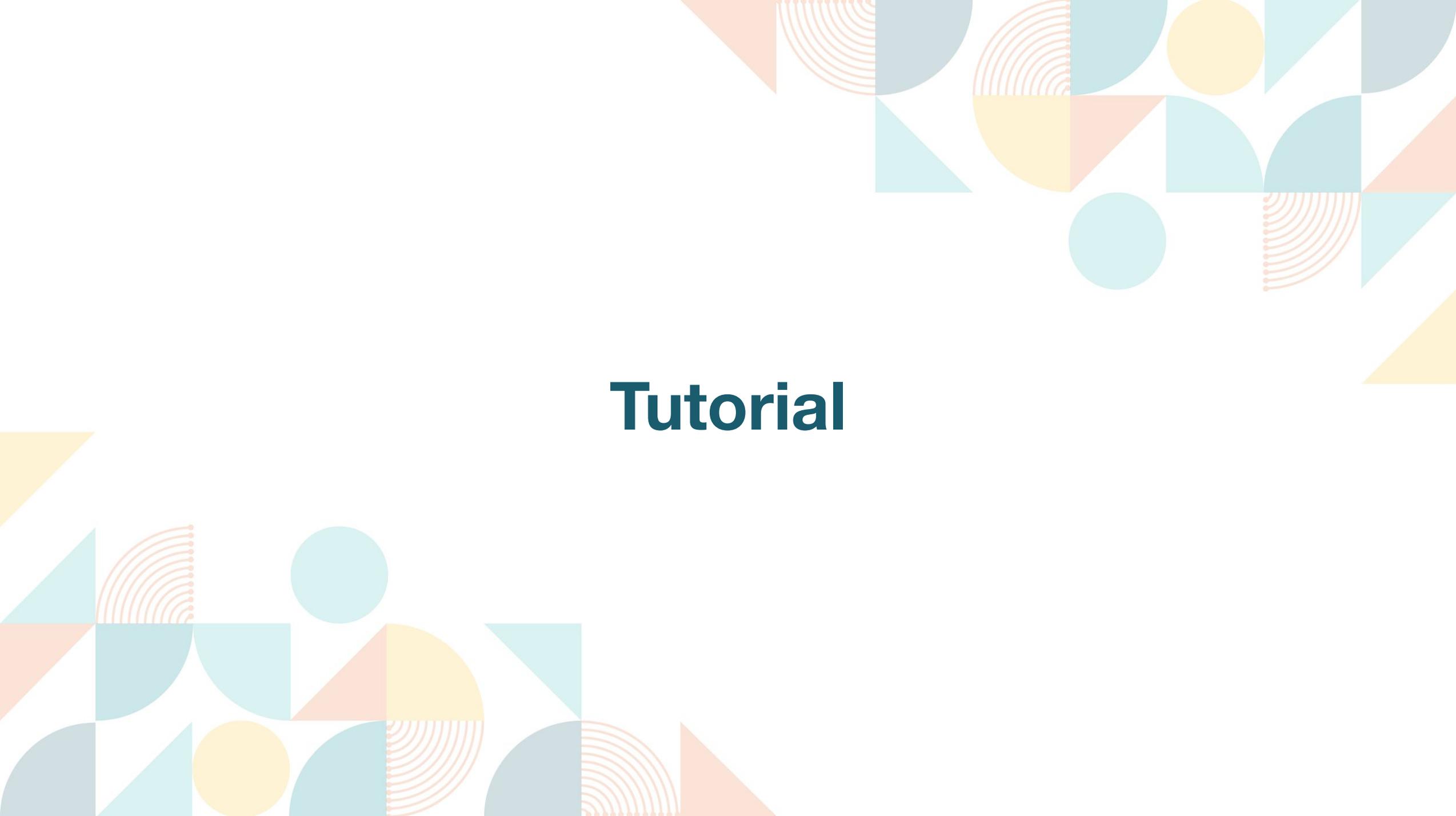
Documentation:

<https://support.access-ci.org/pegasus>

Open a ticket:

<https://support.access-ci.org/open-a-ticket>

Questions?

The image features decorative geometric patterns in the corners. The top-right and bottom-left corners contain clusters of shapes including triangles, circles, semi-circles, and concentric arcs in shades of teal, orange, and yellow. The central area is white and contains the word "Tutorial" in a dark teal font.

Tutorial

Tutorial

This is **not** using ACCESS resources - jobs are staying local in the container.

If we are not finishing here today, feel free to keep exploring on your own

In-person: handout

Remote: <https://tinyurl.com/pegasus-ern>