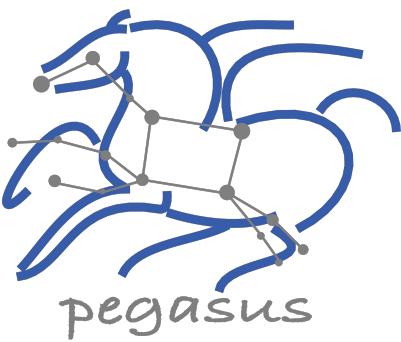


Unlock the Power of Bioinformatics Pipelines with the Pegasus Workflow Management System Workshop



Mats Rynge¹, Karan Vahi¹, Tomasz Osinski²

¹Information Sciences Institute, University of Southern California

²USC Center for Advanced Research Computing (CARC)

rynge@isi.edu , vahi@isi.edu , osinski@usc.edu



Advanced Research Computing
Enabling scientific breakthroughs at scale



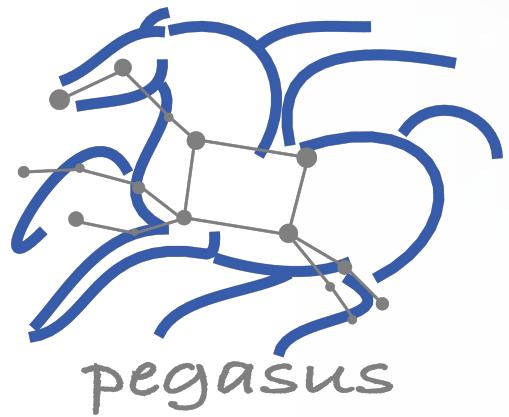
Advancing
Innovation



Preparations and Outline

- You need to be on USC Network (USC Secure Wifi or USC VPN) and need to use your USC credentials to log in
 - Use a web browser and log on to USC OnDemand Instance at <https://ondemand.carc.usc.edu>
-

- SlideDeck:
<https://pegasus.isi.edu/tutorial/keck23/USC-HSC-2023-Pegasus.pdf>
- Slides - Introduction to Pegasus
- Hands on (API, Debugging, Command Line)
- Variant Calling Example



1. Introduction

Workflow Systems and USC CARC / HPC?

- We will talk about:
 - Multiple job workloads
 - Relationship between jobs
 - Automatic data management
 - ... and more
- HPC is not just parallel jobs
 - High throughput computing (HTC)

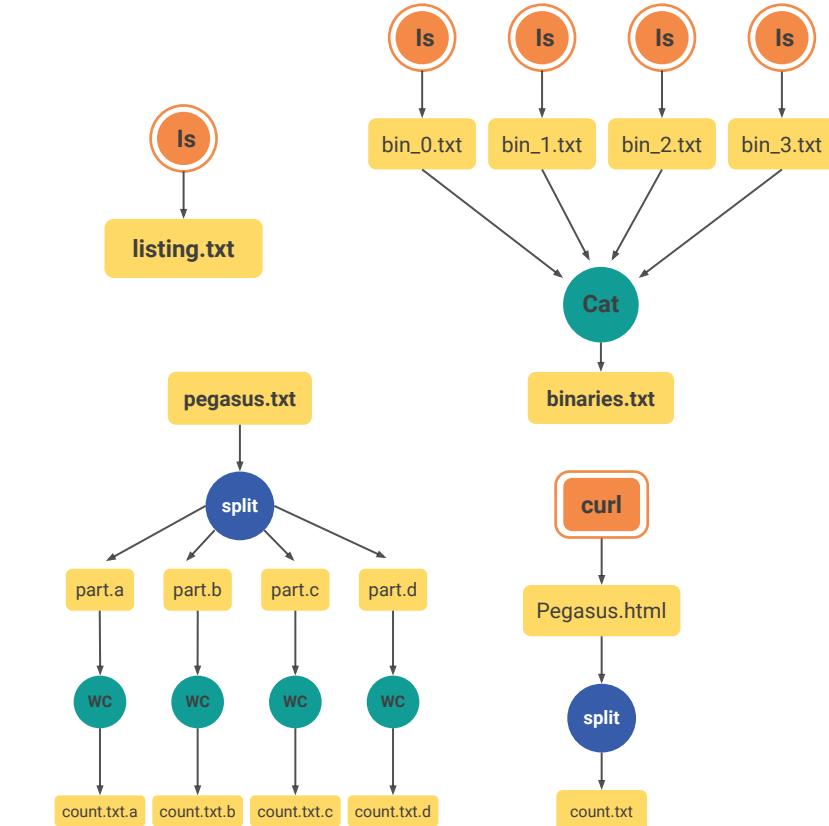


What are Scientific Workflows



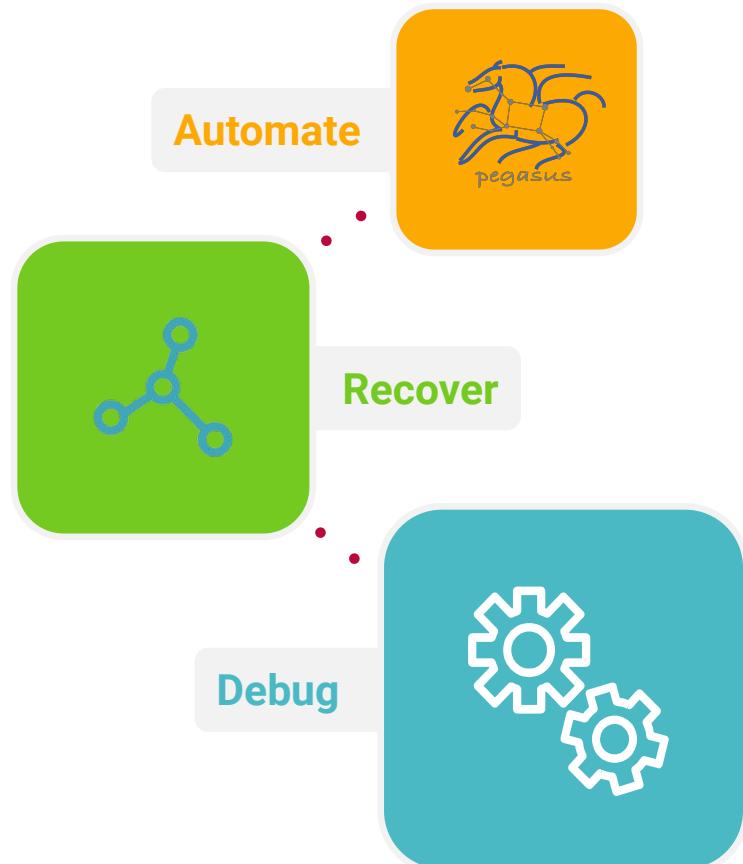
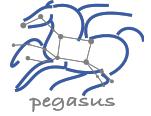
- ▶ **Conducts a series of computational tasks.**
 - Resources distributed across Internet.
- ▶ **Chaining (outputs become inputs) replaces manual hand-offs.**
 - Accelerated creation of products.
- ▶ **Ease of use - gives non-developers access to sophisticated codes.**
 - Resources distributed across Internet.
- ▶ **Provides framework to host or assemble community set of applications.**
 - Honors original codes. Allows for heterogeneous coding styles.
- ▶ **Framework to define common formats or standards when useful.**
 - Promotes exchange of data, products, codes. Community metadata.
- ▶ **Multi-disciplinary workflows can promote even broader collaborations.**
 - E.g., ground motions fed into simulation of building shaking.
- ▶ **Certain rules or guidelines make it easier to add a code into a workflow.**

Workflow Building Blocks

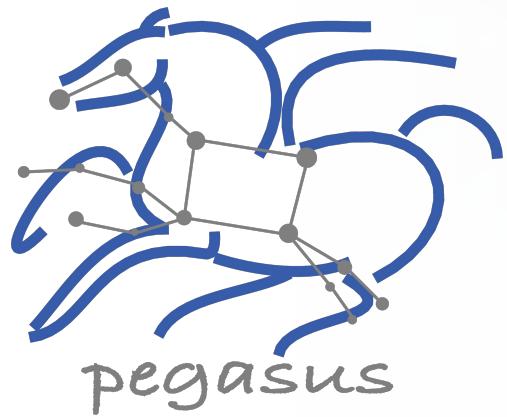


Slide Content Courtesy of David Okaya, SCEC, USC

Why Pegasus?

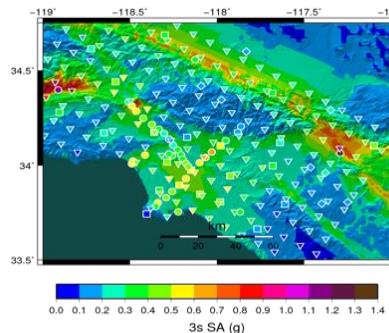


- ▶ **Automates Complex**, Multi-stage Processing Pipelines
- ▶ Enables Parallel, **Distributed Computations**
- ▶ **Automatically Executes** Data Transfers
- ▶ Reusable, Aids **Reproducibility**
- ▶ Records How Data was Produced (**Provenance**)
- ▶ Handles **Failures** with to Provide Reliability
- ▶ Keeps Track of Data and **Files**
- ▶ Ensures **Data Integrity** during workflow execution



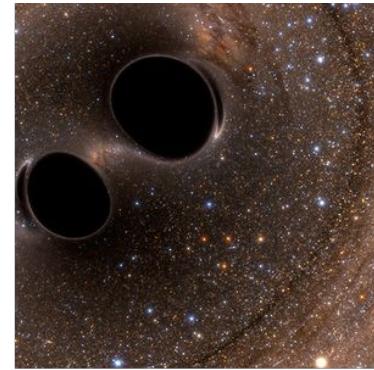
Some of The Success Stories...

Southern California Earthquake Center's CyberShake



First Physics-Based "Shake map" of Southern California

Laser Interferometer Gravitational-Wave Observatory (LIGO)



First direct detection of a gravitational wave (colliding black holes)

XENONnT - Dark Matter Search



Mix of MPI and single-core jobs, mix of CPU, GPU codes.
Large data sets (10s of TBs), ~300 workflows with
420,000 tasks each
Supported since 2005: changing CI, x-platform execution

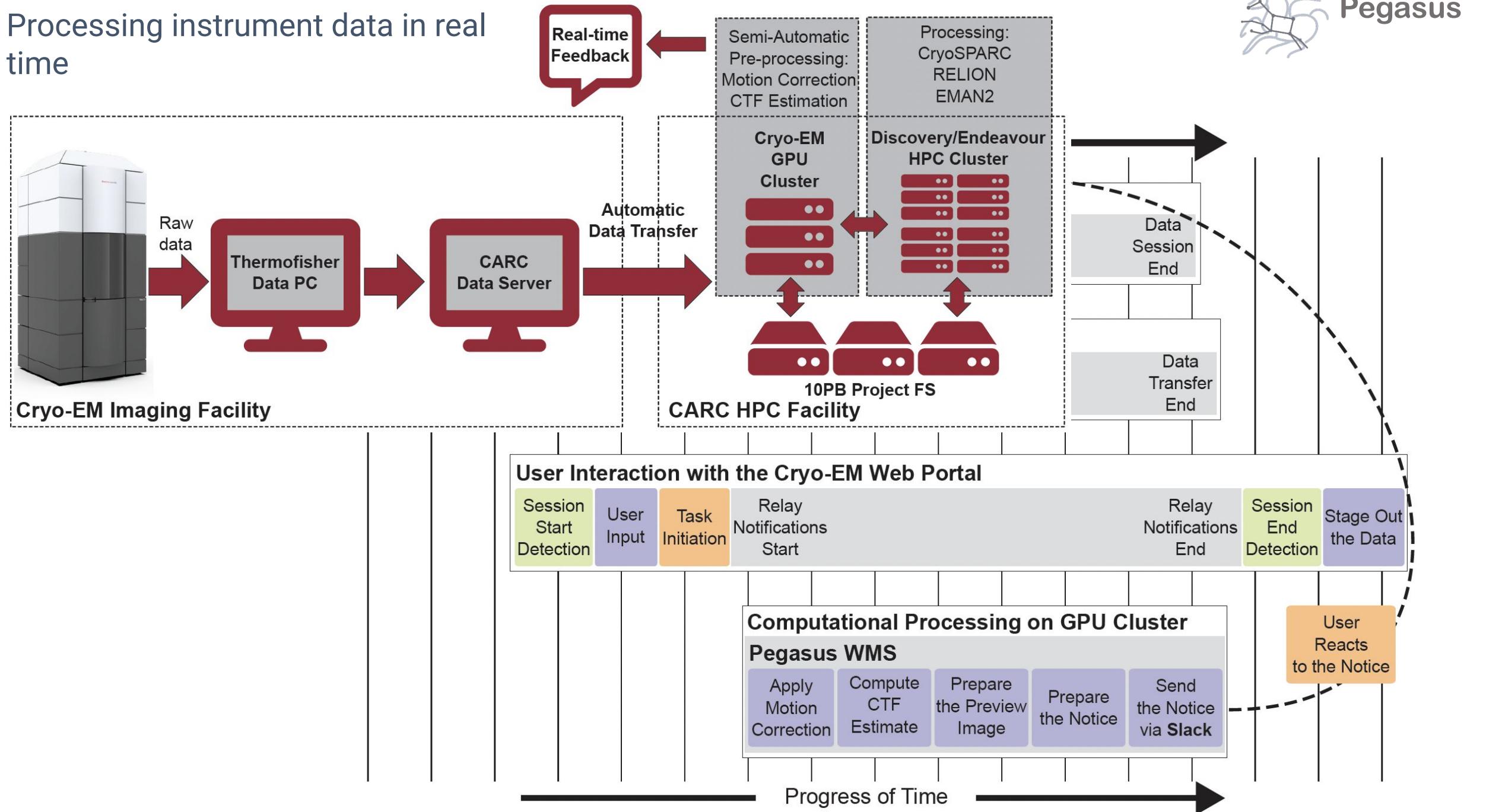
High-throughput computing workload, access to HPC resources, ~ 21K Pegasus workflows, ~ 107M tasks

Supported since 2001, distributed data, opportunistic computing resources

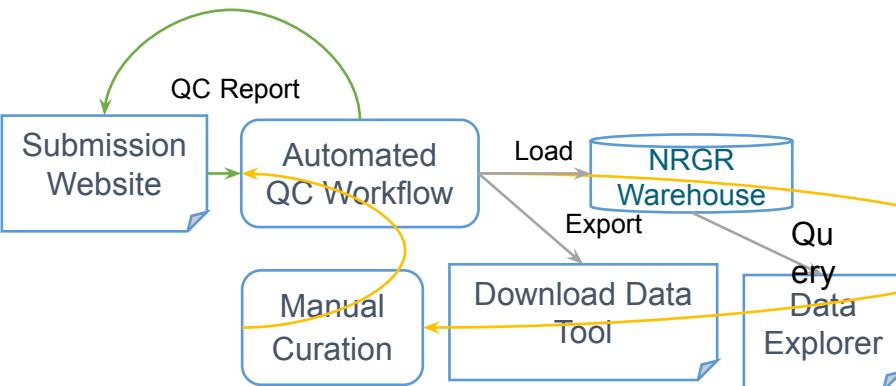
- Custom data management
- Rucio for data management
- MongoDB instance to track science runs and data products.

Monte Carlo simulations and the main processing pipeline.

Processing instrument data in real time



The NIMH Center for Collaborative Genomic Studies on Mental Disorders, now known as the NIMH Repository and Genomics Resource (NRGR), maintains biomaterials, demographic, and phenotypic data from over 200,000 well-characterized individuals with a range of psychiatric illnesses, their family members, and unaffected controls.



Validate with AutoQC

Previous Validations | Help

OVERVIEW HOW TO VALIDATE AND SUBMIT DATA ▾ SUBMISSION REQUIREMENTS ▾ VALIDATE WITH AUTOQC

Validate your data for sanity checks and quality control.

Choose File Browse

What data are you submitting?
-- Choose a Disorder --

Study Id

Email Notification

Validate

Flowchart of validation steps:

```

graph TD
    id_validation[id_validation] --> extended_diagnosis_validation[extended_diagnosis_validation]
    race_ethnicity_validation[race_ethnicity_validation] --> extended_diagnosis_validation
    extended_diagnosis_validation --> advanced_qc[advanced_qc]
    phenotypic_validation[phenotypic_validation] --> advanced_qc
    submission_validation[submission_validation] --> advanced_qc
    pedigree_validation[pedigree_validation] --> advanced_qc
    
```

- **Easy to Use Web-Based Interface**
 - Simple Submission
 - Real-time Monitoring and Error Reports
 - After automated QC, submit corrected files for expert curation
- **Scalable**
 - Workflow based architecture using Pegasus WMS
- **Extensible Design**
 - Easily add new QC steps, and checks
- **Enables Complex checks**
 - Pedigree Checks
 - QC Checks validating data with external sources
 - QC Checks can correlate data across multiple files and across multiple fields within files
- Ensures high-quality uniform data deposited at NRGR
- Better resource utilization: solve most QC problems automatically, use expert curation for hard cases

<https://pegasus.isi.edu>

Auto QC Status

◀ Back to Previous Validations

Successful: 100%

Summary

UID	5e6a6ddd95f6e
Disorder	Depression
Study Id	149
File	shaptest7.zip
User	JaclynVitanza
Email	jv607@dls.rutgers.edu
Started On	Mar 12, 2020 10:14 AM
Workflow Directory	/web/data/qc/runs/5e6a6ddd95f6e

Sanity Check Status

Download All Files		
File	Submission Validation	Pedigree Validation
study_149_sub.csv	Standardized File Download	Log Download
ID Validation		
study_149_id.csv	Standardized File Download	Log Download
Phenotypic Validation		
shaps01_phen.csv	Standardized File Download	Log Download
Advanced QC		
study_149_sub.canon.csv	Corrected Submission File Download	
study_149_id.canon.csv	Corrected ID File Download	
Corrections Log	Corrections Log Download	
Advanced QC Report	Advanced QC Report Download	

SOYBEAN KNOWLEDGE BASE (SoyKB)
A web resource for Soybean Translational Genomics

SoyKB Home

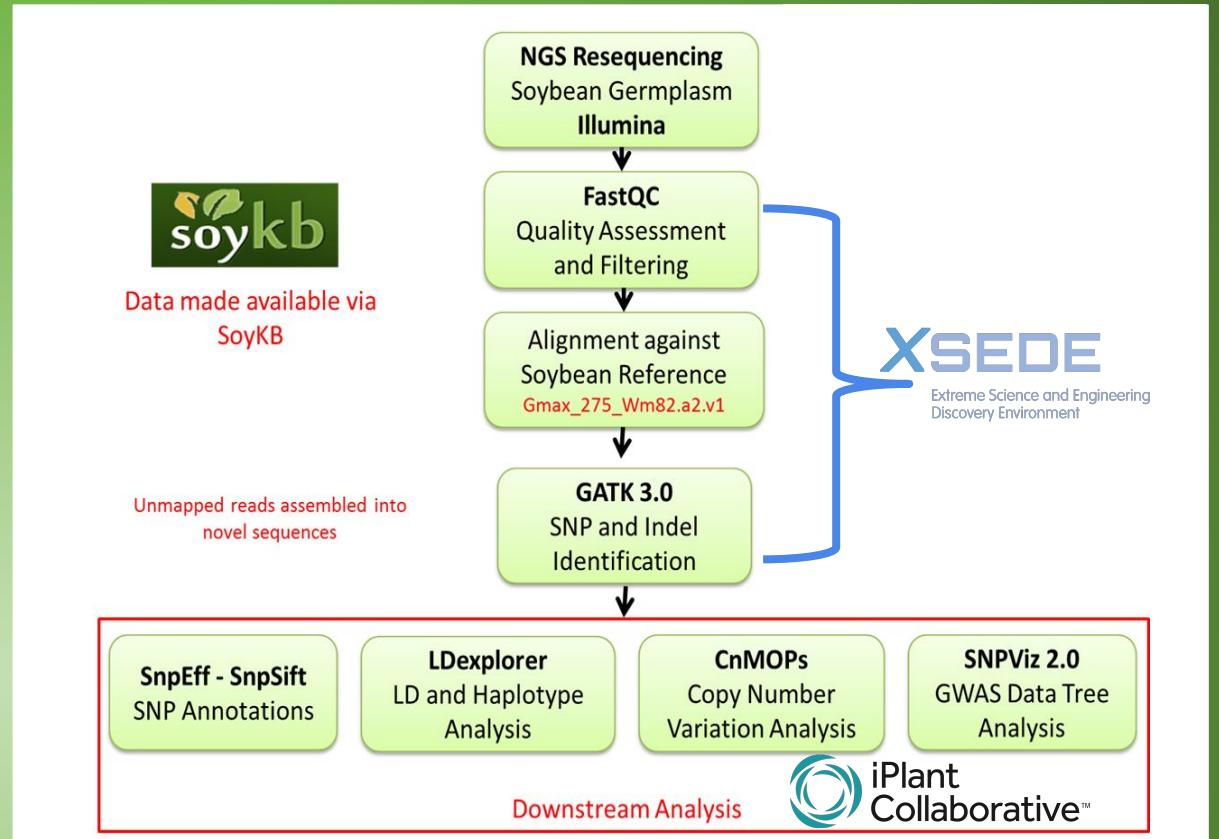
A hallmark of modern biology is tremendous amounts of complex omics data, which require large-scale data management, comprehensive computational analyses, and efficient integration, for better understanding of the data and hypothesis generation. For soybean with a newly sequenced genome, there is an increasing need from the soybean community to have a one-stop interactive, web-based portal to browse, access and share knowledge about soybean.

Towards this, we developed the Soybean Knowledge Base (SoyKB), a comprehensive all-inclusive web resource for soybean. SoyKB is designed to handle the storage and integration of the gene, genomics, EST, microarray, transcriptomics, proteomics, metabolomics, pathway and phenotype data.

SoyKB provides an informatics-based social network system to build connections among soybean researchers, producers and consumers.

Latest News

- SoyKB: a powerful tool at the junction of plant biology and computer science
- University of Missouri Leads Soybean Sequencing Effort
- SoyKB: Leading the convergence of wet and dry science in the era of Big Data

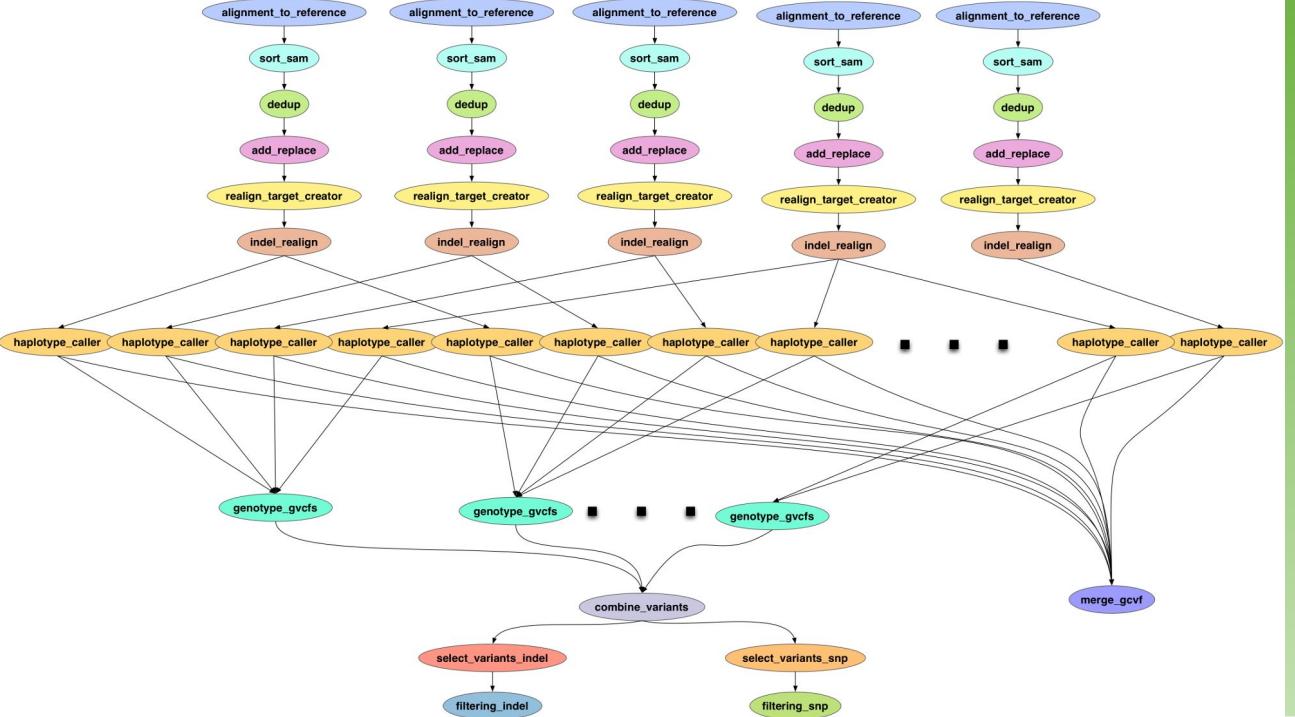


<http://soykb.org>

XSEDE Allocation

PI: Dong Xu

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



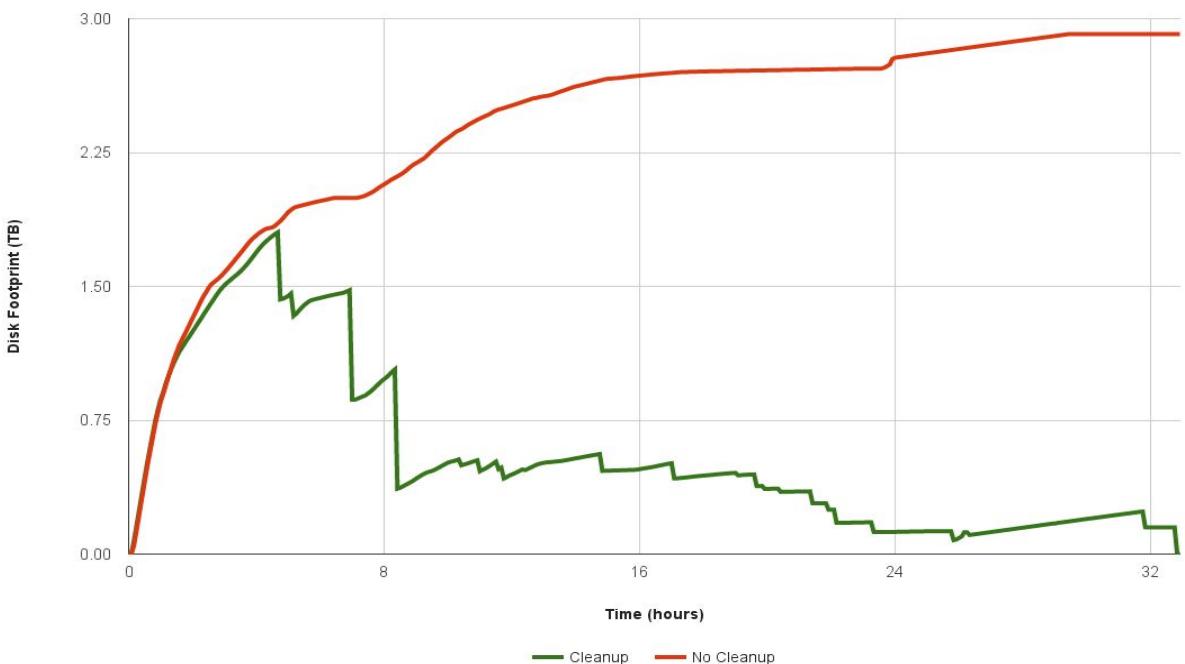
TACC Wrangler as Execution Environment

Flash Based Shared Storage

Switched to glideins (pilot jobs) - Brings in remote compute nodes and joins them to the HTCondor pool on in the submit host - Workflow runs at a finer granularity

Works well on Wrangler due to more cores and memory per node (48 cores, 128 GB RAM)

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





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

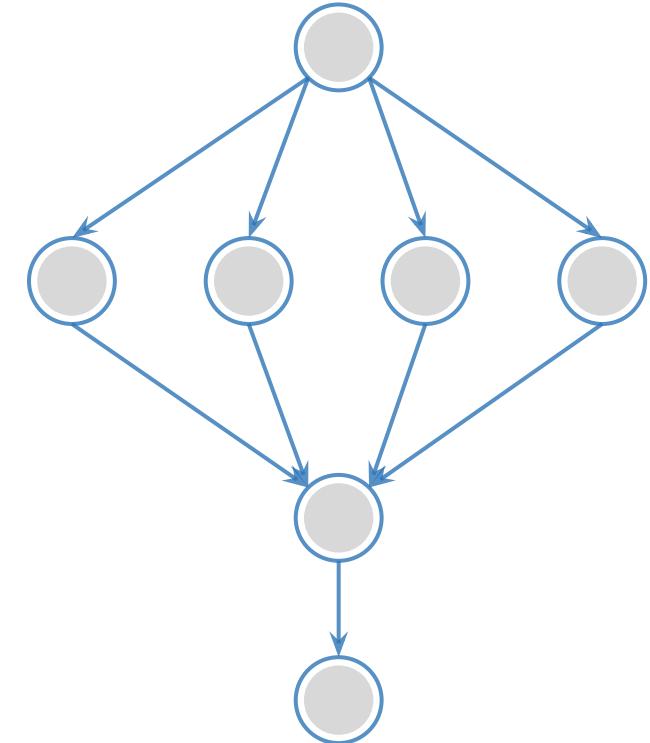
▲ Workflows are DAGs

- 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

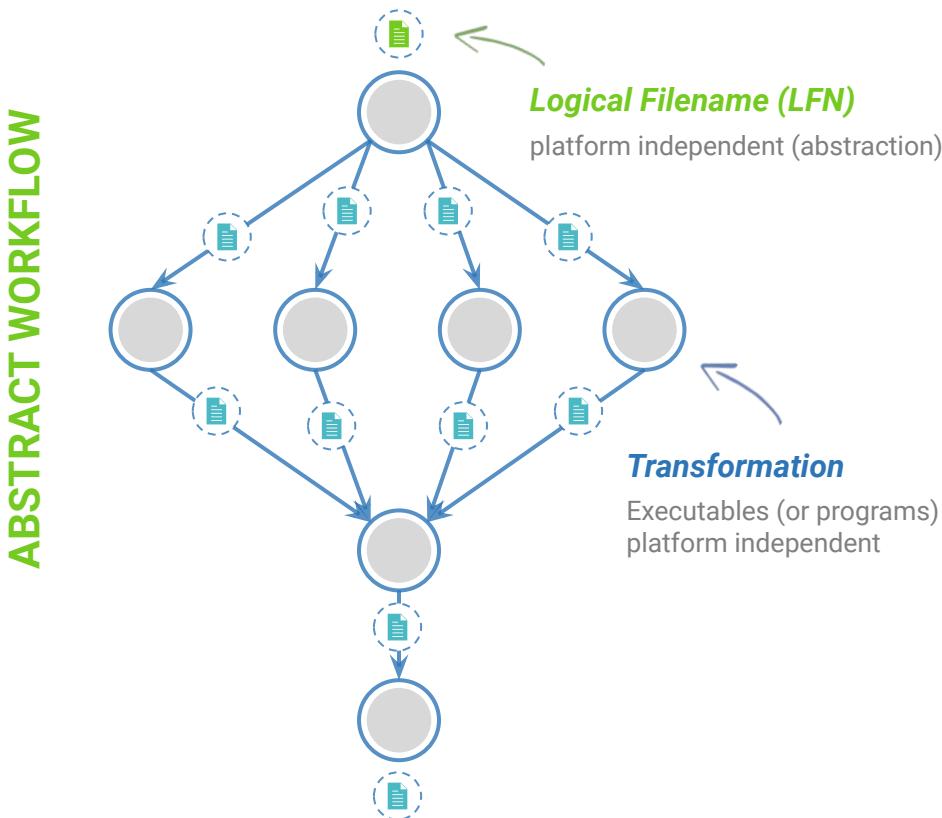
- Planner is like a compiler



Input Workflow Specification YAML formatted

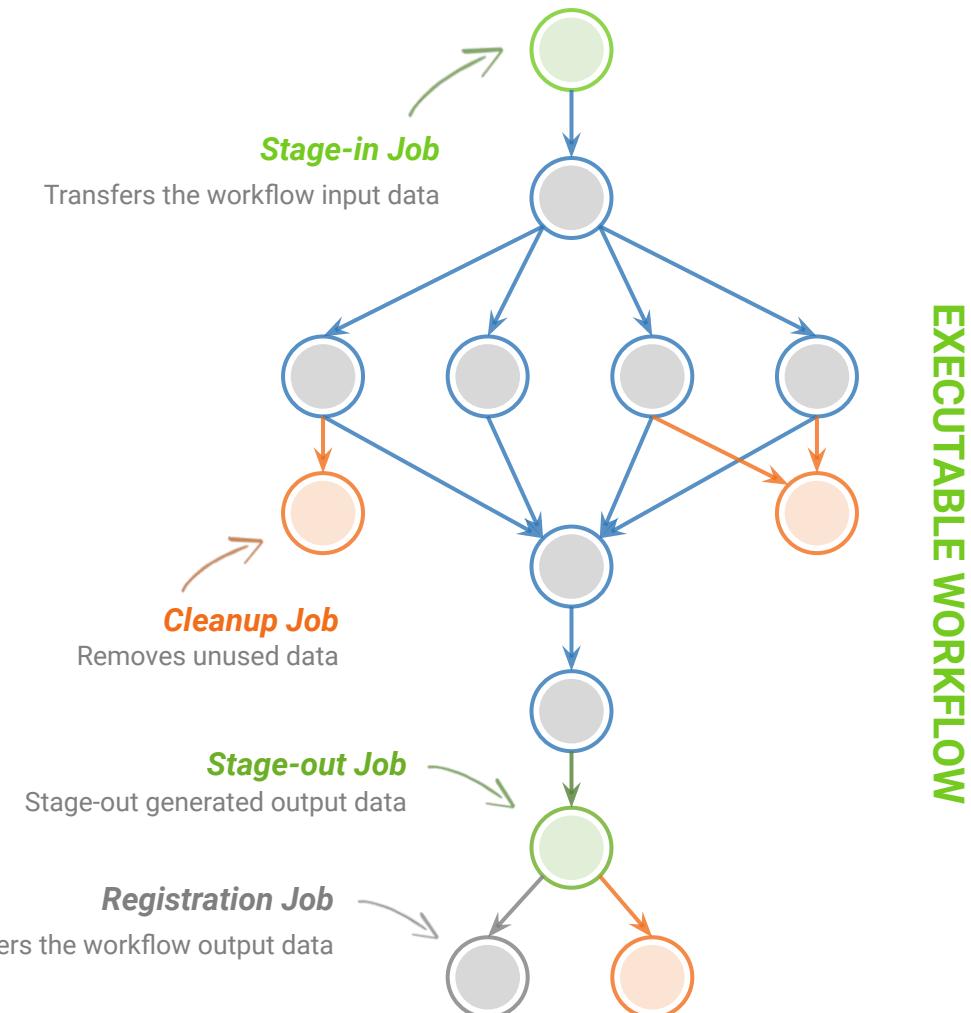
Portable Description

Users do not worry about low level execution details



directed-acyclic graphs

Output Workflow



Pegasus Deployment



Workflow Submit Node

- Pegasus WMS
- HTCondor

One or more Compute Sites

- Compute Clusters
- Cloud
- OSG

Input Sites

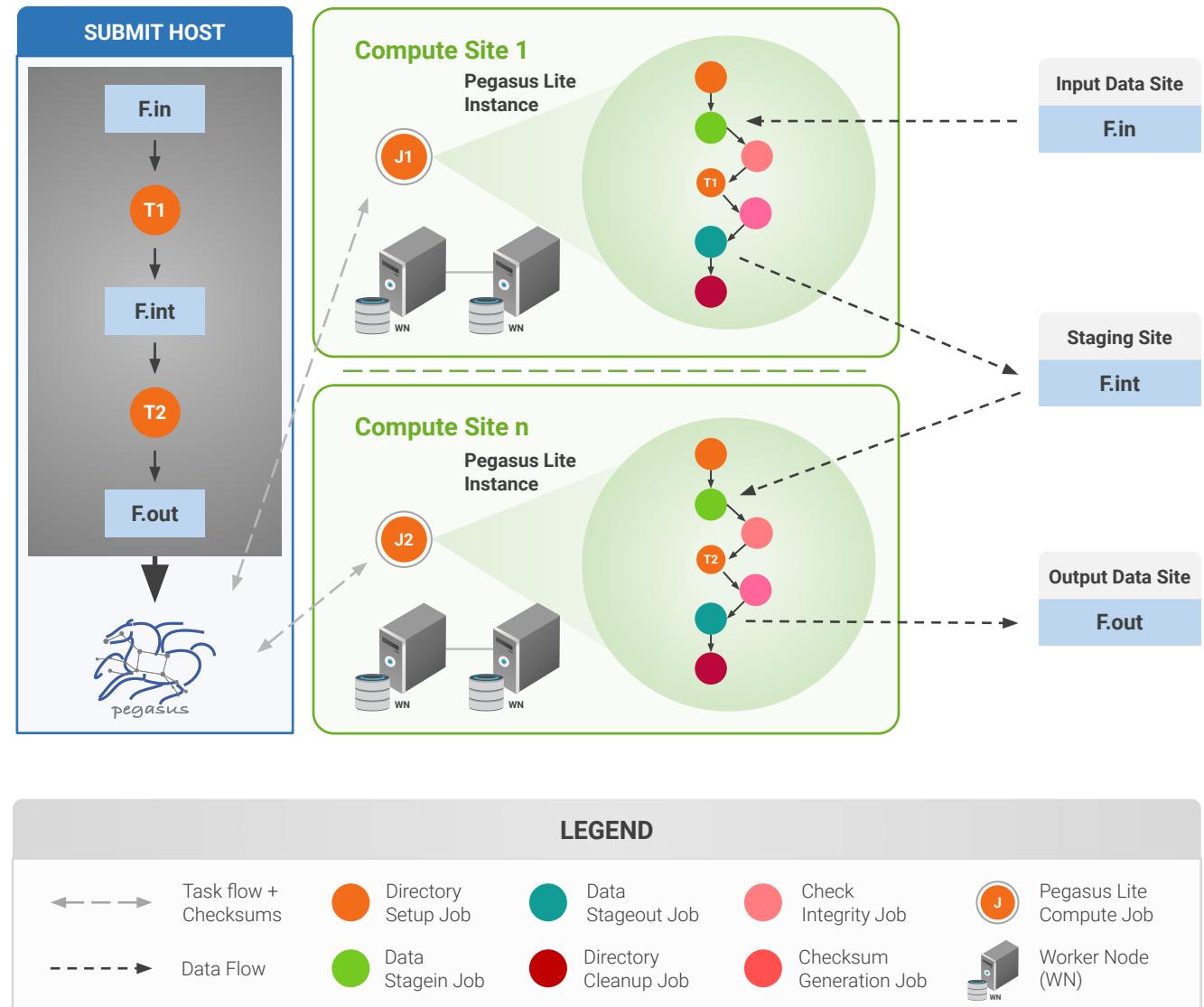
- Host Input Data

Data Staging Site

- Coordinate data movement for workflow

Output Site

- Where output data is placed





Pegasus-transfer

Pegasus' internal data transfer tool with support for a number of different protocols

● Directory creation, file removal

- If protocol can support it, also used for cleanup

● Two stage transfers

- e.g., GridFTP to S3 = GridFTP to local file, local file to S3

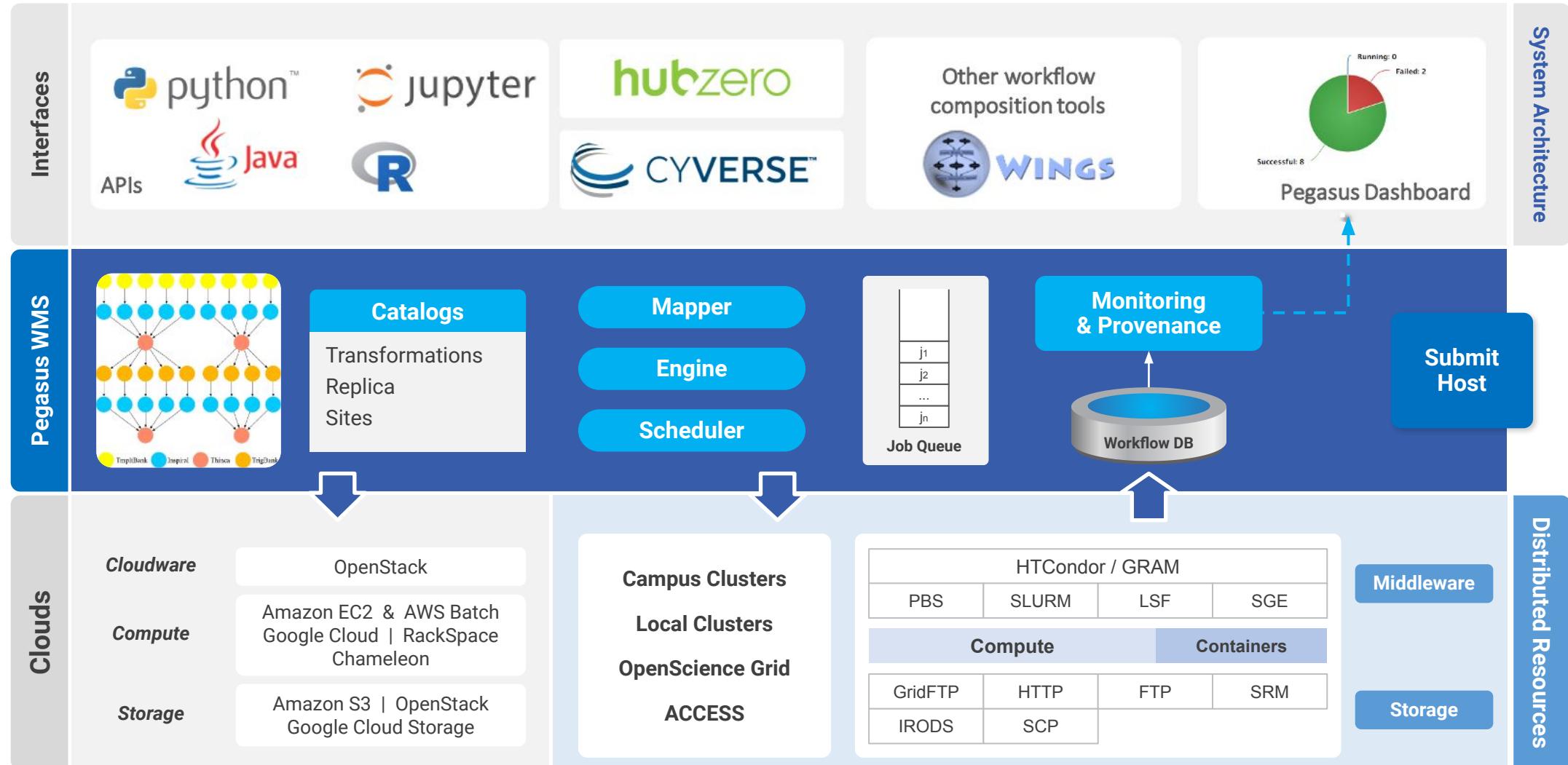
● Parallel transfers

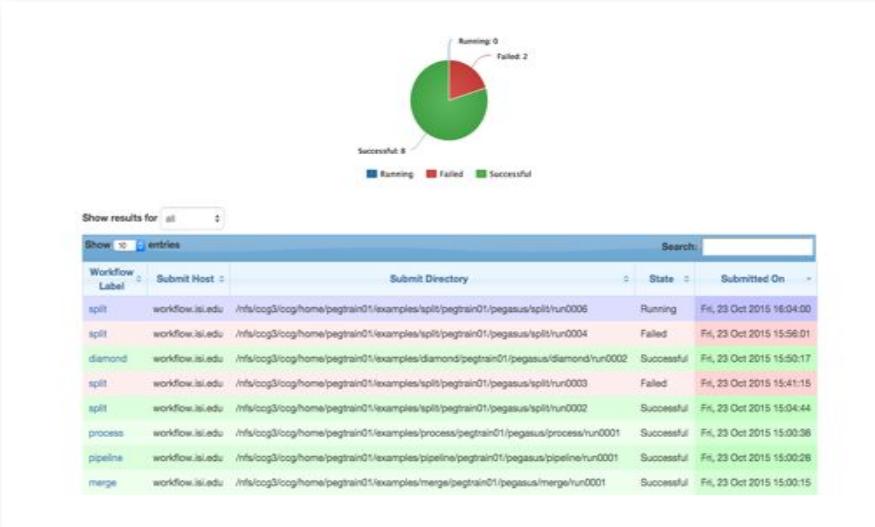
● Automatic retries

● Credential management

- Uses the appropriate credential for each site and each protocol (even 3rd party transfers)

HTTP
SCP
GridFTP
Globus
Online
iRods
Amazon S3
Google Storage
SRM
FDT
Stashcp
Rucio
cp
ln -s





Show results for: all

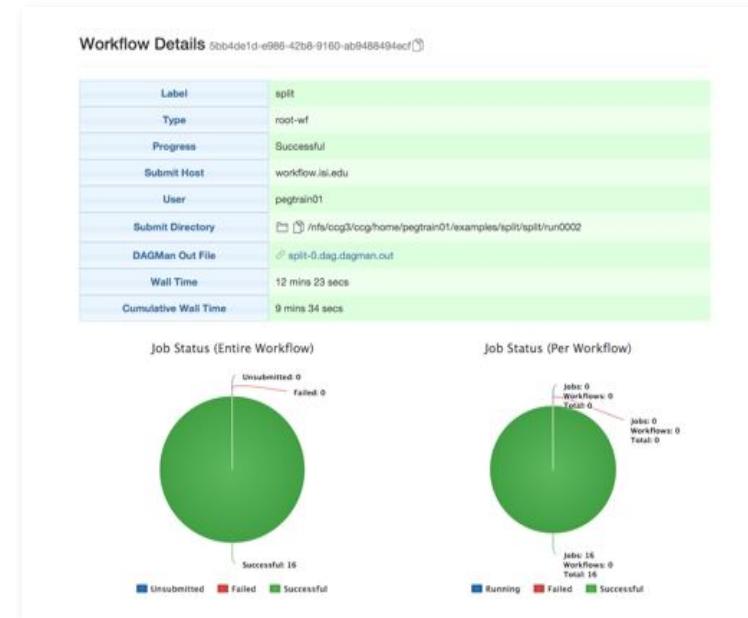
Show: 50 entries

Search:

Workflow Label	Submit Host	Submit Directory	State	Submitted On
split	workflow.isi.edu	/nfs/cog3/cog/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0006	Running	Fri, 23 Oct 2015 16:04:00
split	workflow.isi.edu	/nfs/cog3/cog/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0004	Failed	Fri, 23 Oct 2015 15:56:01
diamond	workflow.isi.edu	/nfs/cog3/cog/home/pegtrain01/examples/diamond/pegtrain01/pegasus/diamond/run0002	Successful	Fri, 23 Oct 2015 15:50:17
split	workflow.isi.edu	/nfs/cog3/cog/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0003	Failed	Fri, 23 Oct 2015 15:41:15
split	workflow.isi.edu	/nfs/cog3/cog/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0002	Successful	Fri, 23 Oct 2015 15:04:44
process	workflow.isi.edu	/nfs/cog3/cog/home/pegtrain01/examples/process/pegtrain01/pegasus/process/run0001	Successful	Fri, 23 Oct 2015 15:00:38
pipeline	workflow.isi.edu	/nfs/cog3/cog/home/pegtrain01/examples/pipeline/pegtrain01/pegasus/pipeline/run0001	Successful	Fri, 23 Oct 2015 15:00:26
merge	workflow.isi.edu	/nfs/cog3/cog/home/pegtrain01/examples/merge/pegtrain01/pegasus/merge/run0001	Successful	Fri, 23 Oct 2015 15:00:15

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

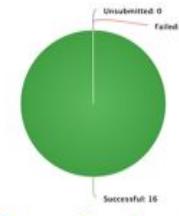
Provenance data is stored into a relational database.



Workflow Details 5bb4de1d-e986-42b8-9160-ab9486494ecf

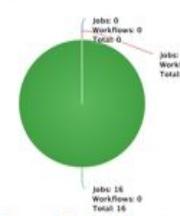
Label	split
Type	root-wf
Progress	Successful
Submit Host	workflow.isi.edu
User	pegtrain01
Submit Directory	/nfs/cog3/cog/home/pegtrain01/examples/split/split/run0000
DAGMan Out File	split-0.dag.dagman.out
Wall Time	12 mins 23 secs
Cumulative Wall Time	9 mins 34 secs

Job Status (Entire Workflow)



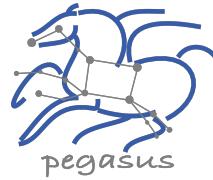
Successful: 16
Unsubmitted: 0
Failed: 0

Job Status (Per Workflow)



Jobs: 16
Workflows: 0
Failed: 0
Unsubmitted: 0
Total: 16

<https://pegasus.isi.edu>



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 Waittime as seen from Submit Side	9 mins 35 secs
Workflow Cumulative Badput Time	9 mins 23 secs
Cumulative Job Badput Waittime as seen from Submit Side	9 mins 20 secs
Workflow Retries	1

Workflow Statistics

This Workflow						
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
Entire Workflow						
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

Job Breakdown Statistics

Job Statistics

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

And if a job fails?



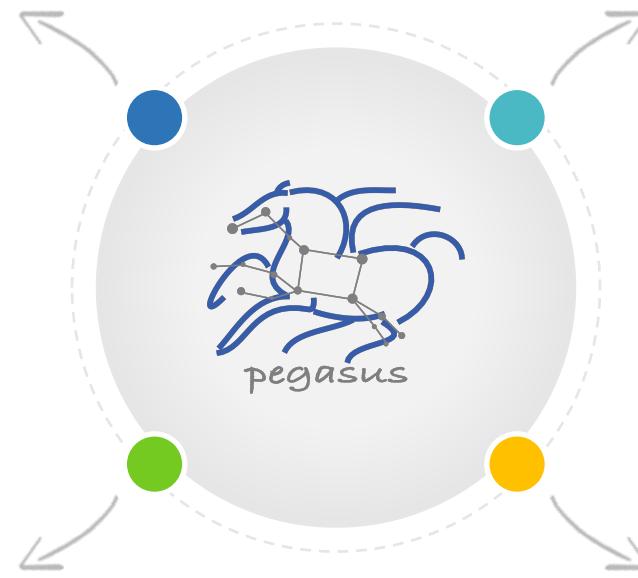
Postscript

detects non-zero exit code output
parsing for success or failure
message exceeded timeout do not
produced expected output files



Checkpoint Files

job generates checkpoint files
staging of checkpoint files is
automatic on restarts



Job Retry



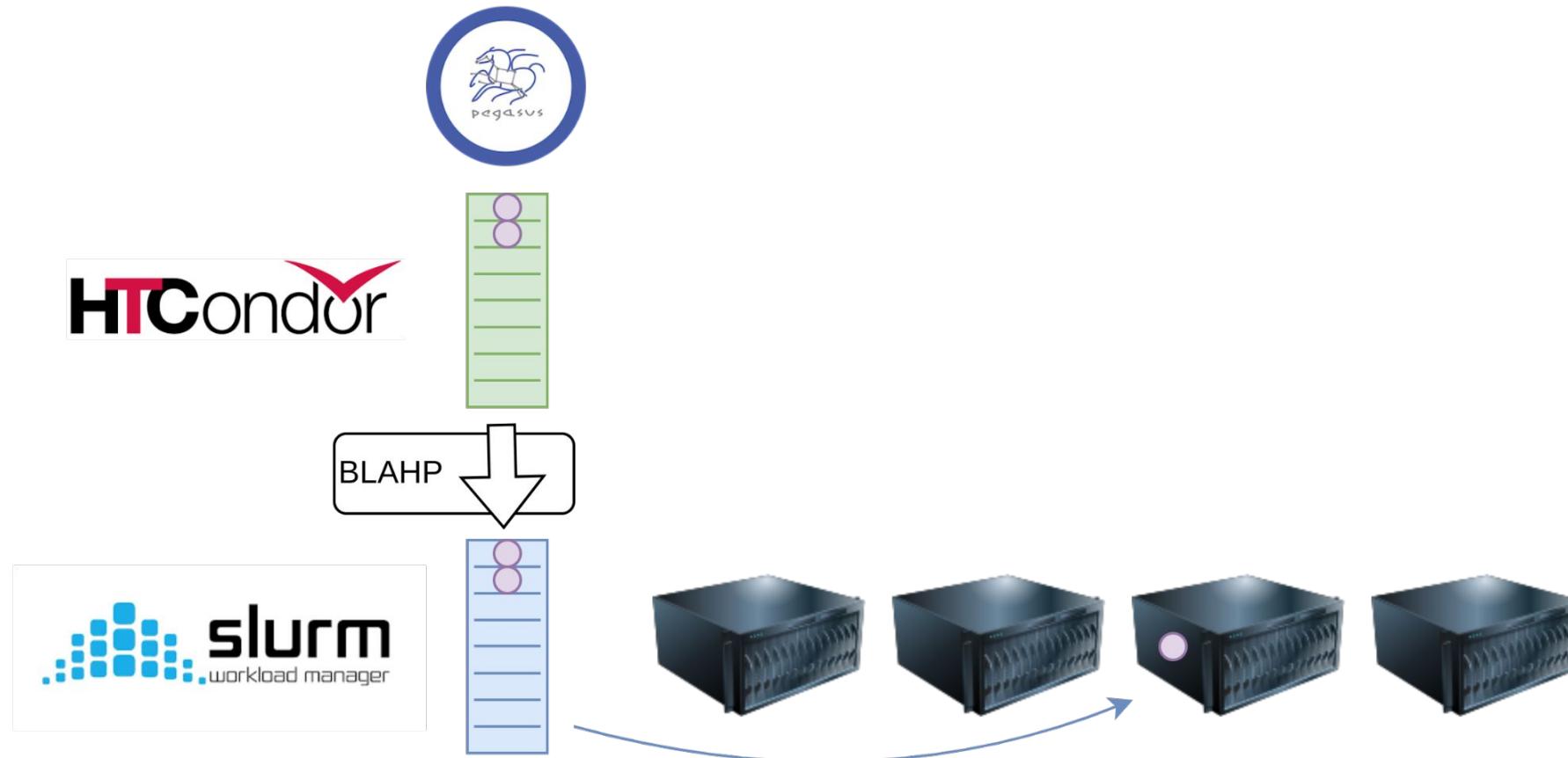
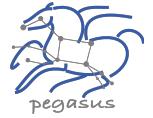
helps with transient failures
set number of retries per job
and run



Rescue DAGs

workflow can be restarted from
checkpoint file recover from
failures with minimal loss

HTCondor with BLAHP translation layer



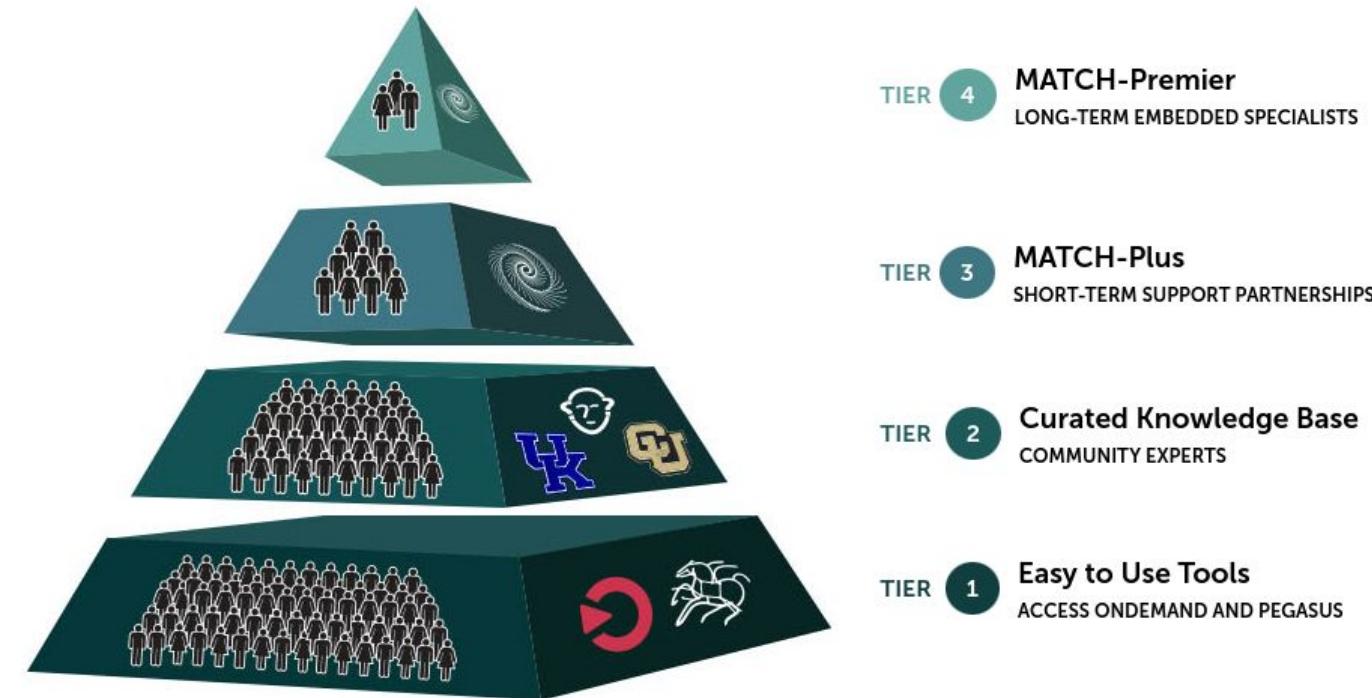
Pegasus is part of the ACCESS support strategy

Pegasus is be used as a tier 1 tool

Central Open OnDemand instance with Pegasus, HTCondor and Jupyter

It is be easy to run HTC workflows across ACCESS sites

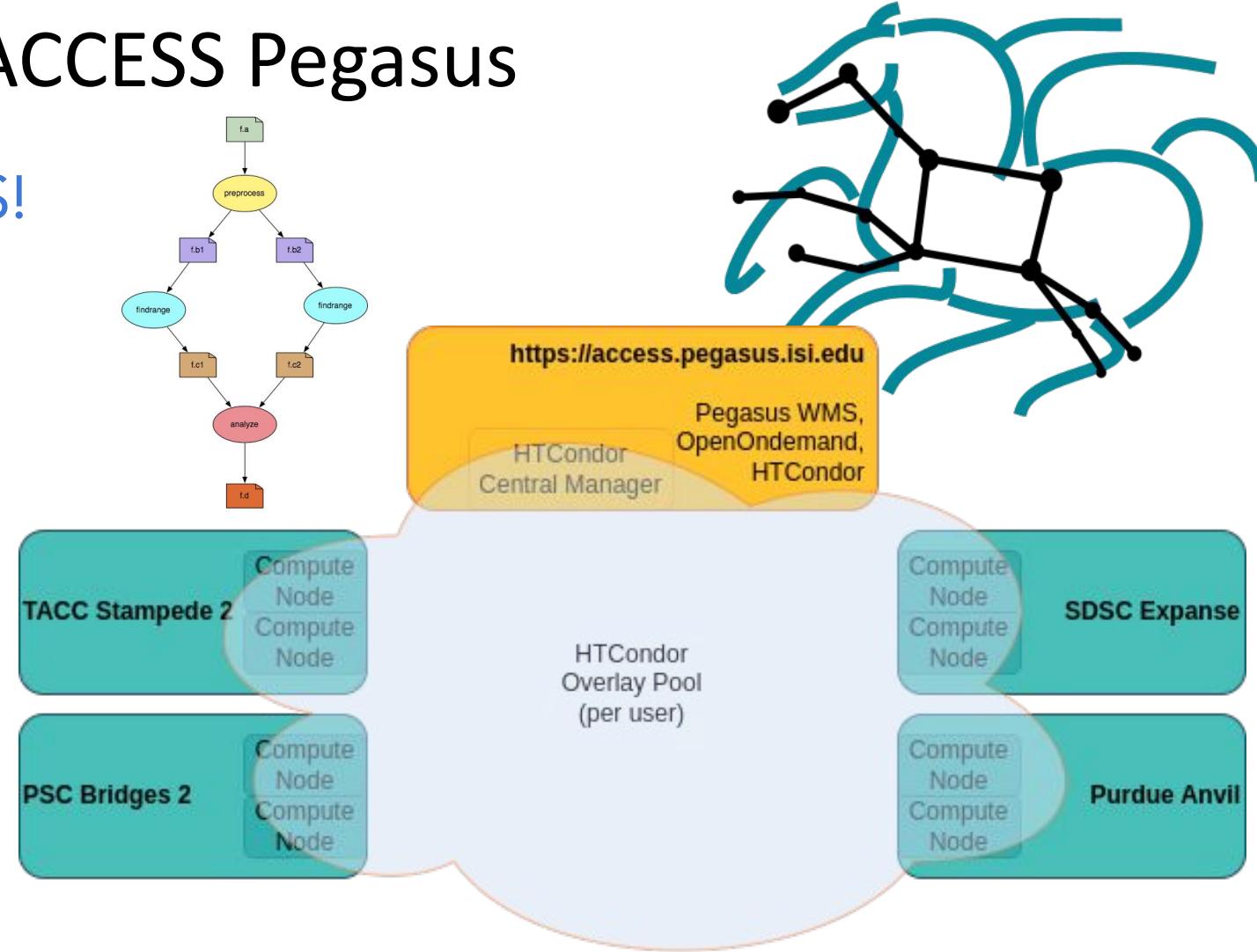
Tiered Support Strategy



ACCESS Pegasus

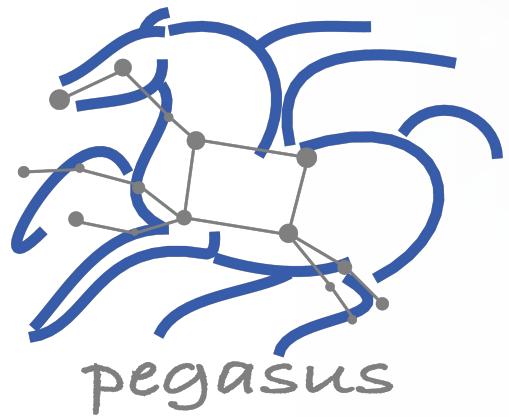
Bring your workflows to ACCESS!

- Execute scientific workflows across ACCESS resources
- OpenOnDemand Portal: has all you need: Jupyter Notebooks, ACCESS authentication, Pegasus workflow management, and HTCondor job management
- Bring your own ACCESS capacity: HTCondor Annex - pilot jobs automatically create a virtual HTCondor pool



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

More at: support.access-ci.org/pegasus



2. Hands on Exercises

Hands on Tutorial Exercises: Login to Open OnDemand

- You need to be on USC Network (USC Secure Wifi or USC VPN) and need to use your USC credentials to log in
- Use a web browser and log on to USC OnDemand Instance at <https://ondemand.carc.usc.edu> .

Hands on Tutorial Exercises: Start a Jupyter Server

- Start a Jupyter notebook server, Click on Interactive Apps and then select JupyterLab

The screenshot shows a web browser window for the URL `ondemand.carc.usc.edu`. The page header includes standard OS X-style window controls (red, yellow, green) and a title bar with the URL. Below the title bar is a navigation bar with links for 'CARC OnDemand', 'Files', 'Jobs', 'Clusters', and 'Interactive Apps'. The 'Interactive Apps' link is highlighted with a yellow box. A dropdown menu is open under 'Interactive Apps', titled 'Interactive Apps'. It contains two sections: 'GUIs' and 'Servers'. Under 'GUIs', there are links for COMSOL, IGV, MATLAB, Mathematica, Stata/MP, Terminal, and VS Code. Under 'Servers', there are links for JupyterLab and RStudio Server. The 'JupyterLab' link is also highlighted with a yellow box. The main content area of the page features the USC logo and the text 'Advanced Research Computing' and 'Enabling scientific breakthroughs'. It also mentions 'OnDemand provides an integrated, single access point to our HPC resources.'

powered by
OPEN **OnDemand**
Open "https://ondemand.carc.usc.edu/pun/sys/dashboard/batch_connect/sys/jupyter/session_contexts/new" in a new tab

OnDemand version: 2.0.29

Hands on Tutorial Exercises: Jupyter Lab Configuration

- When launching the Jupyter Lab, it is important to select the following
- For Cluster: specify [Discovery](#)
- For Account: specify the account [osinski_982](#) *
- For Partition specify [htcondor](#)

*The account is workshop specific and will be active for 7 days after the workshop

The screenshot shows the CARC OnDemand web interface. At the top, there is a navigation bar with links for 'CARC OnDemand', 'Files', 'Jobs', 'Clusters', 'Interactive Apps', and a user icon. Below the navigation bar, the URL 'Home / My Interactive Sessions / JupyterLab' is displayed. The main content area is titled 'JupyterLab' and contains the following fields:

- Cluster:** discovery
- Modules to load (optional):** (empty input field)
- Account:** ttrojan_123
- Partition:** htcondor
- Number of CPUs:** 1
- Memory (GB):** 1
- GPU Type (optional):** (empty input field)
- Number of GPUs (optional):** (empty input field)

The left sidebar lists various interactive applications under 'Interactive Apps', including COMSOL, IGV, MATLAB, Mathematica, Stata/MP, Terminal, VS Code, and JupyterLab, which is highlighted with a blue background.

Hands on Tutorial Exercises: Connect to JupyterLab

The screenshot shows the CARC OnDemand web interface. The top navigation bar includes links for CARC OnDemand, Files, Jobs, Clusters, Interactive Apps, and a user profile. A green notification bar at the top center says "Session was successfully created." Below the navigation is a breadcrumb trail: Home / My Interactive Sessions.

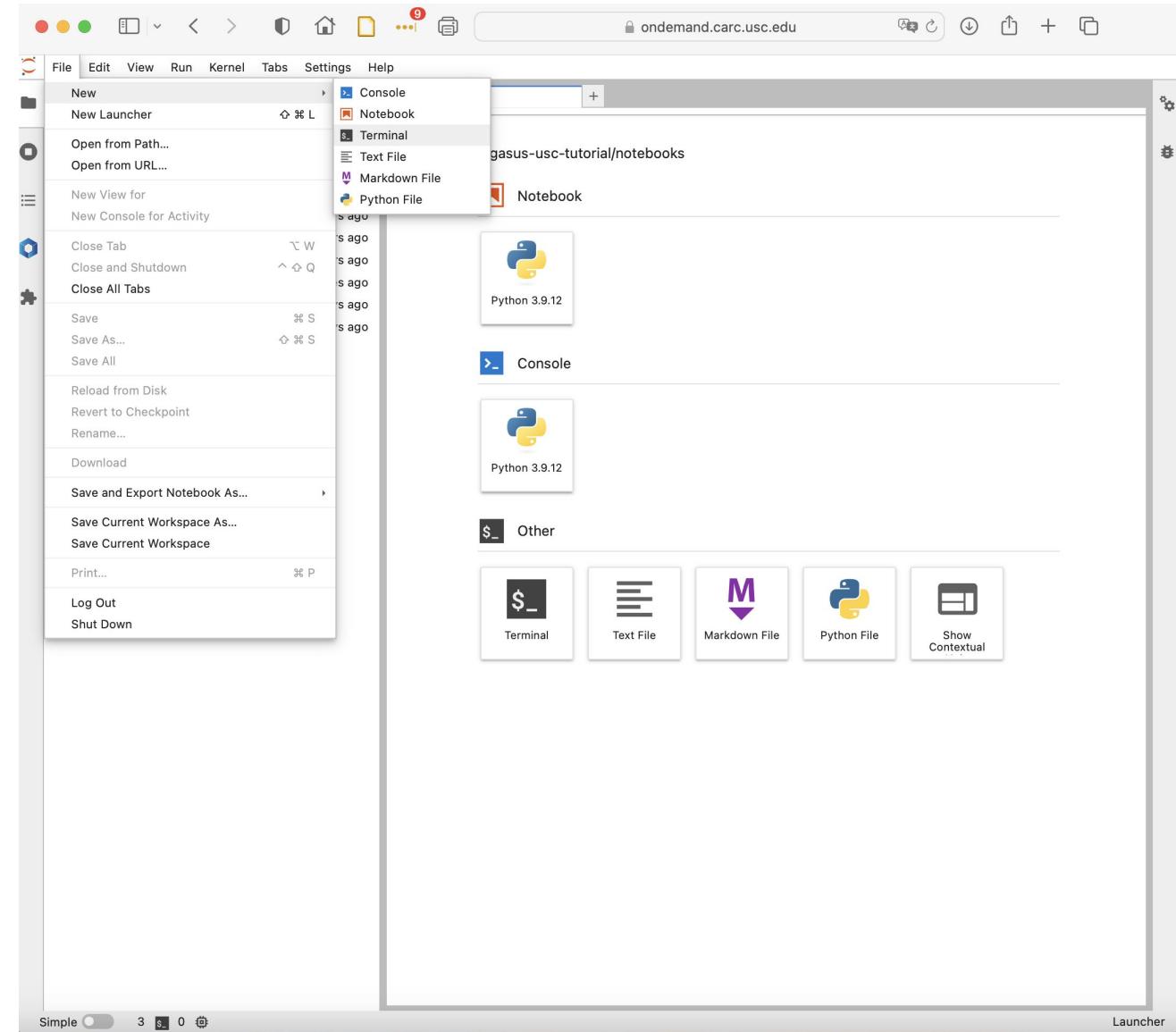
The main content area displays three interactive session cards:

- JupyterLab (14184588)** 1 node | 1 core | Running
Host: >[_e10-12.hpc.usc.edu](#) Delete
Created at: 2023-03-27 17:09:57 PDT
Time Remaining: 5 hours and 58 minutes
Session ID: [8e5389bc-46d3-44c7-8d12-d836d6c81419](#)
[Connect to JupyterLab](#)
- JupyterLab (14170685)** Completed
Created at: 2023-03-27 10:48:35 PDT Delete
Session ID: [cbf7d166-ee81-4764-9f8a-9a98f108887d](#)
For debugging purposes, this card will be retained for 6 more days
- JupyterLab (14083145)** Completed
Created at: 2023-03-20 15:45:18 PDT Delete
Session ID: [da637e53-df1a-4866-8f9a-7f6f6ef16ed1](#)
For debugging purposes, this card will be retained for 6 more days

A sidebar on the left lists available interactive apps under "Interactive Apps":
GUIs:
• COMSOL
• IGV
• MATLAB
• Mathematica
• Stata/MP
• Terminal
• VS Code
Servers:
• JupyterLab
• RStudio Server

Hands on Tutorial Exercises: Start a Terminal

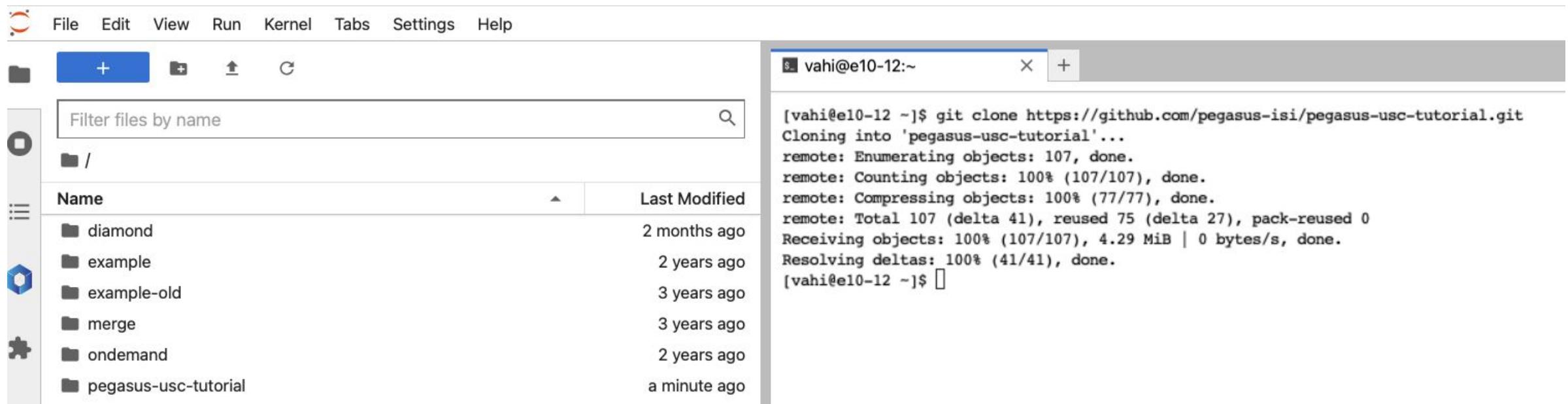
- In JupyterLab, Click on File -> New and then click on Terminal to get the terminal



Hands on Tutorial Exercises: Clone Repository

- Clone Tutorial Repository in the terminal

```
git clone https://github.com/pegasus-isi/pegasus-usc-tutorial.git
```



The screenshot shows a desktop interface with a file manager on the left and a terminal window on the right.

File Manager: The sidebar icons include a circular arrow, File, Edit, View, Run, Kernel, Tabs, Settings, Help, a plus sign, a folder, an upward arrow, and a circular arrow. The main area has a search bar with "Filter files by name" and a list of items under a directory icon. The list includes:

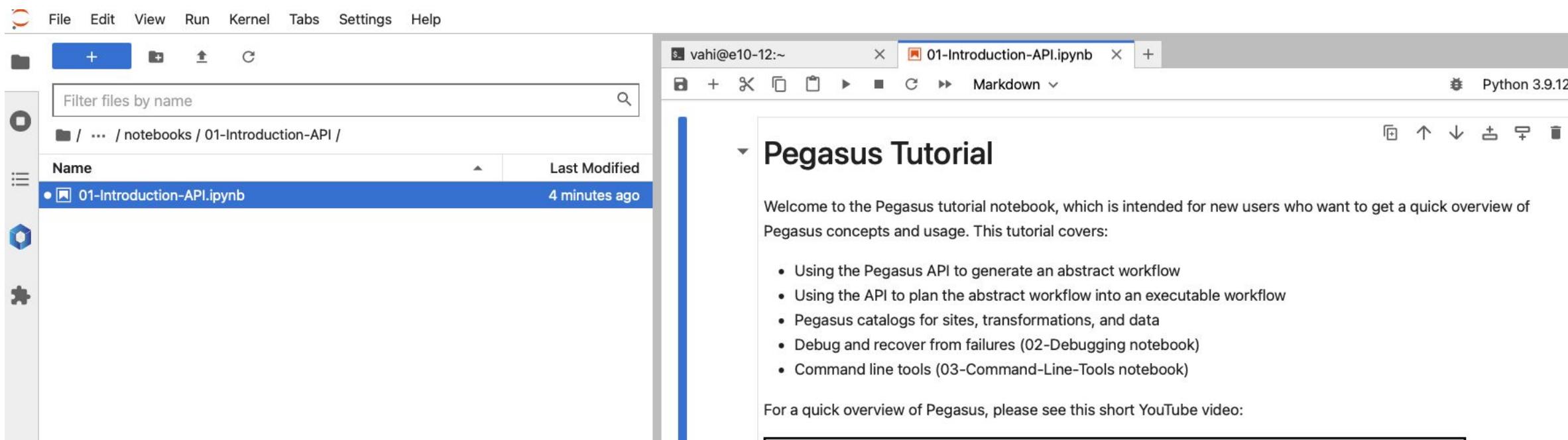
Name	Last Modified
diamond	2 months ago
example	2 years ago
example-old	3 years ago
merge	3 years ago
ondemand	2 years ago
pegasus-usc-tutorial	a minute ago

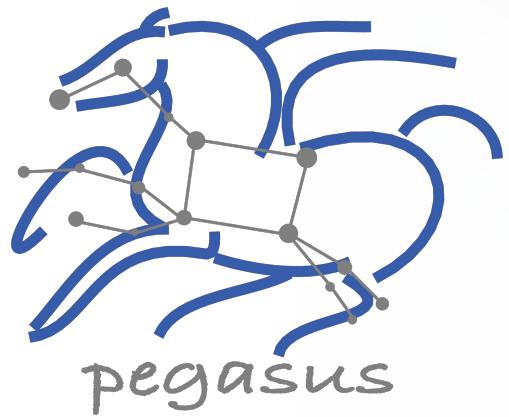
Terminal: The title bar says "vahi@e10-12:~". The terminal window displays the output of a git clone command:

```
[vahi@e10-12 ~]$ git clone https://github.com/pegasus-isi/pegasus-usc-tutorial.git
Cloning into 'pegasus-usc-tutorial'...
remote: Enumerating objects: 107, done.
remote: Counting objects: 100% (107/107), done.
remote: Compressing objects: 100% (77/77), done.
remote: Total 107 (delta 41), reused 75 (delta 27), pack-reused 0
Receiving objects: 100% (107/107), 4.29 MiB | 0 bytes/s, done.
Resolving deltas: 100% (41/41), done.
[vahi@e10-12 ~]$
```

Hands on Tutorial Exercises: Navigate to Notebooks

- In Jupyter, navigate to the example you are interested in, and step through the notebook.
- For first time users, we highly recommend to do the notebooks in order, as they build up on concepts in the previous notebooks.





2.1 API



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

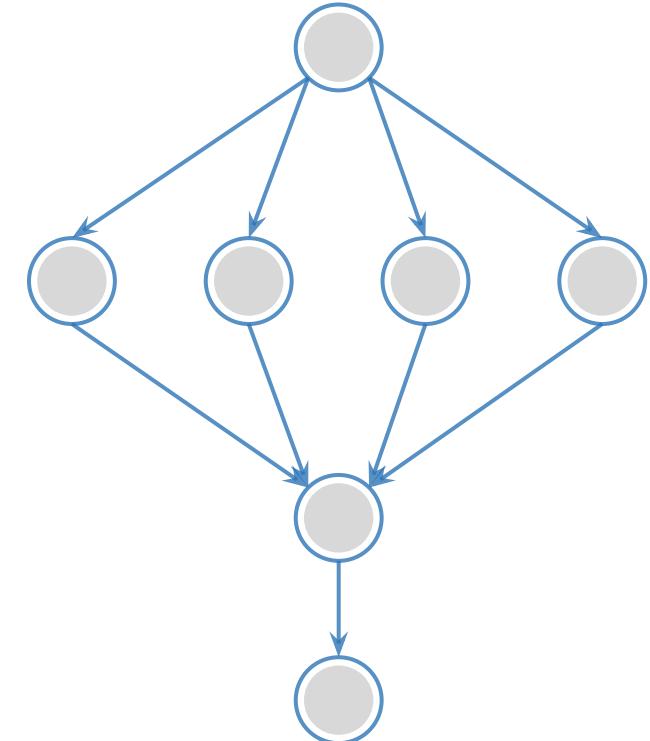
► Workflows are DAGs

- 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



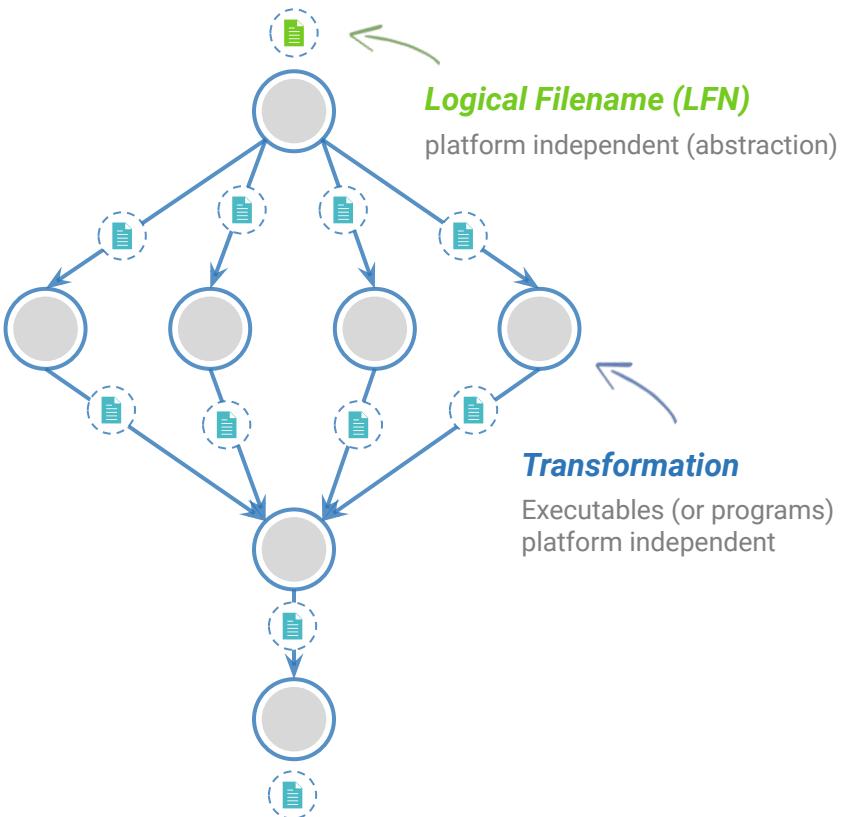


Input Workflow Specification YAML formatted

Portable Description

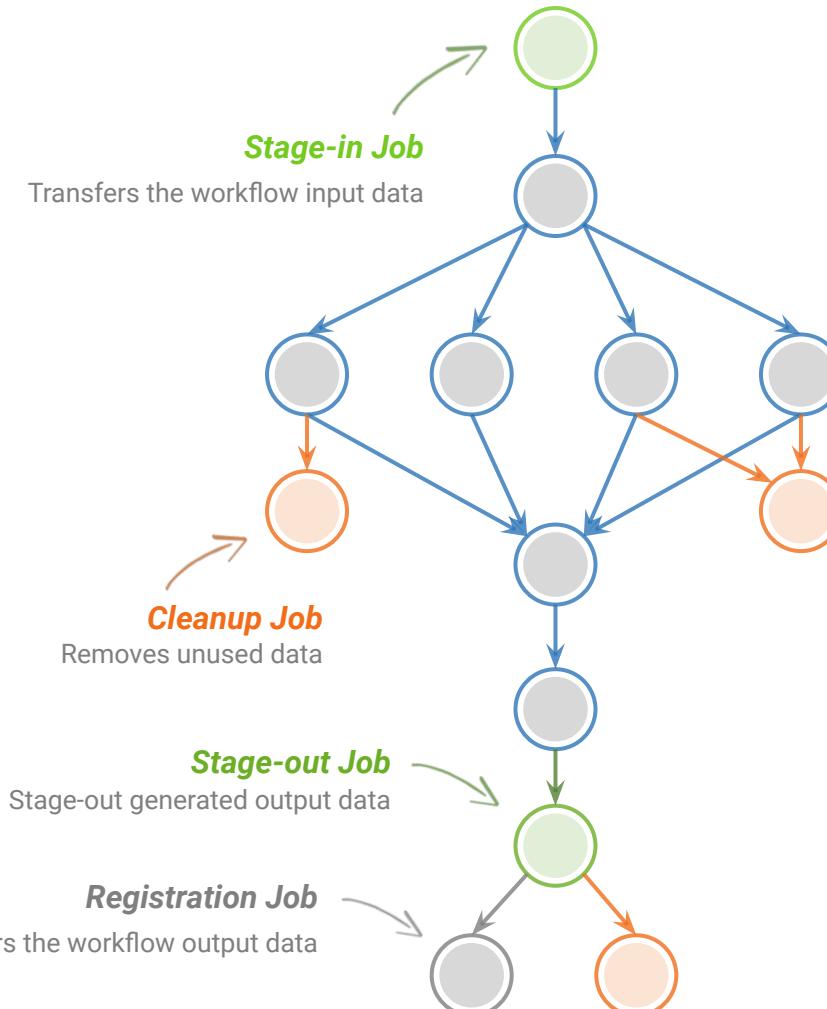
Users do not worry about low level execution details

ABSTRACT WORKFLOW

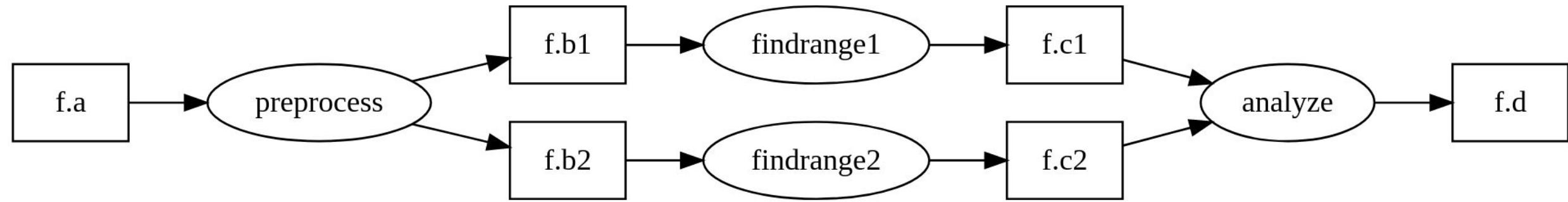


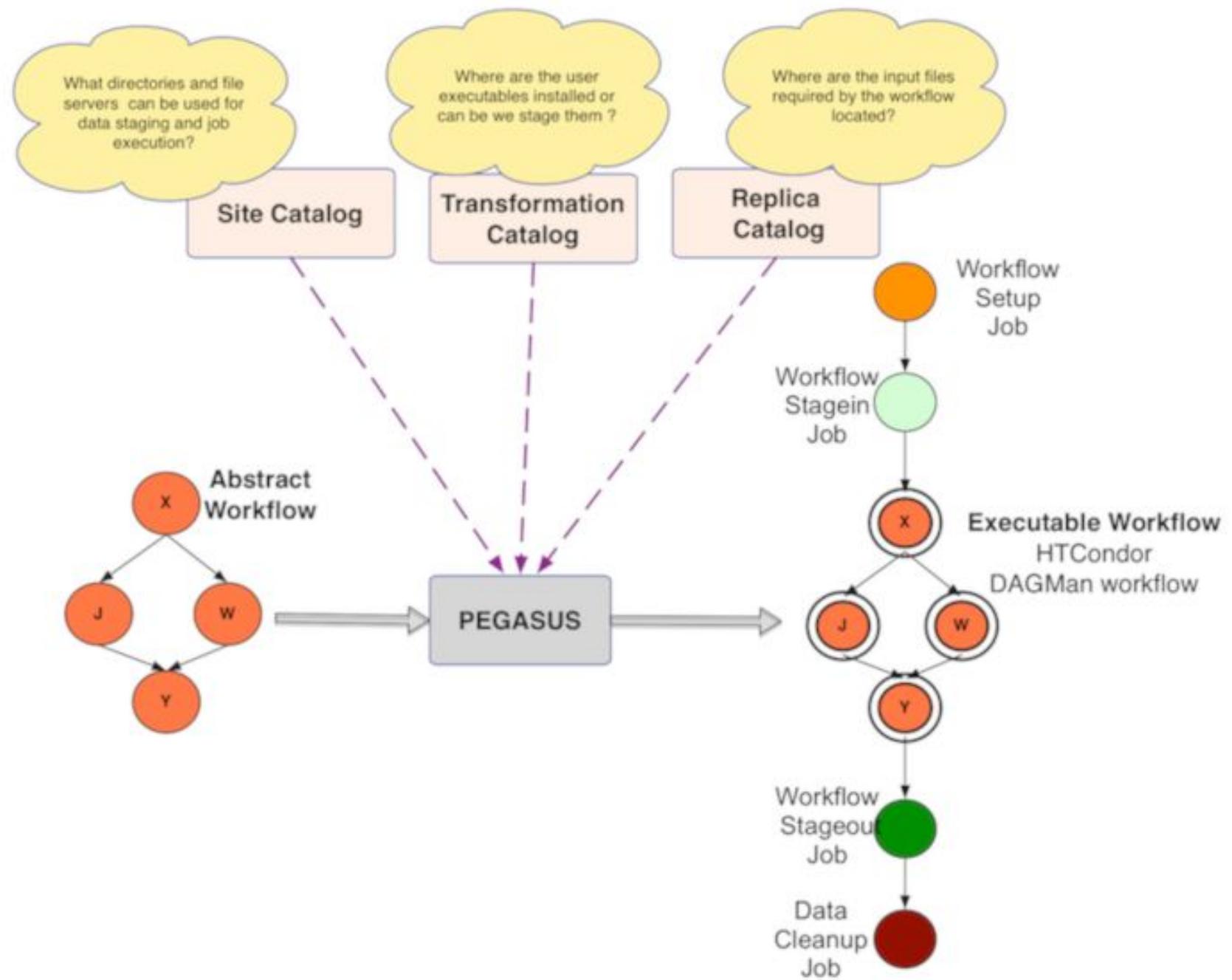
directed-acyclic graphs

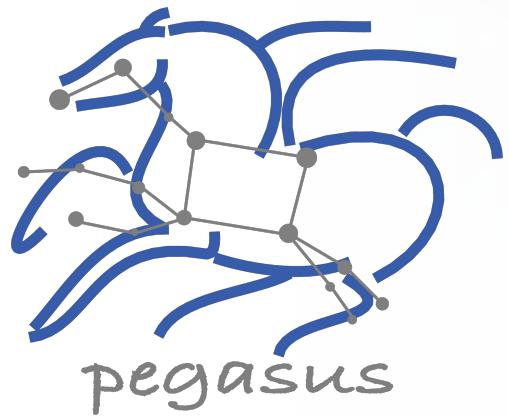
Output Workflow



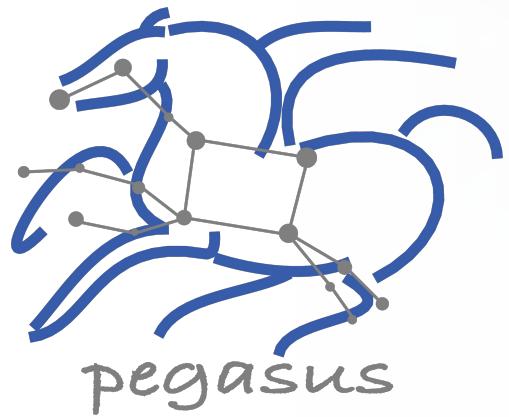
EXECUTABLE WORKFLOW







2.2 Debugging



2.3 Command Line Tools

Pegasus Container Support



Users can refer to **containers** in the **Transformation Catalog** with their executable preinstalled



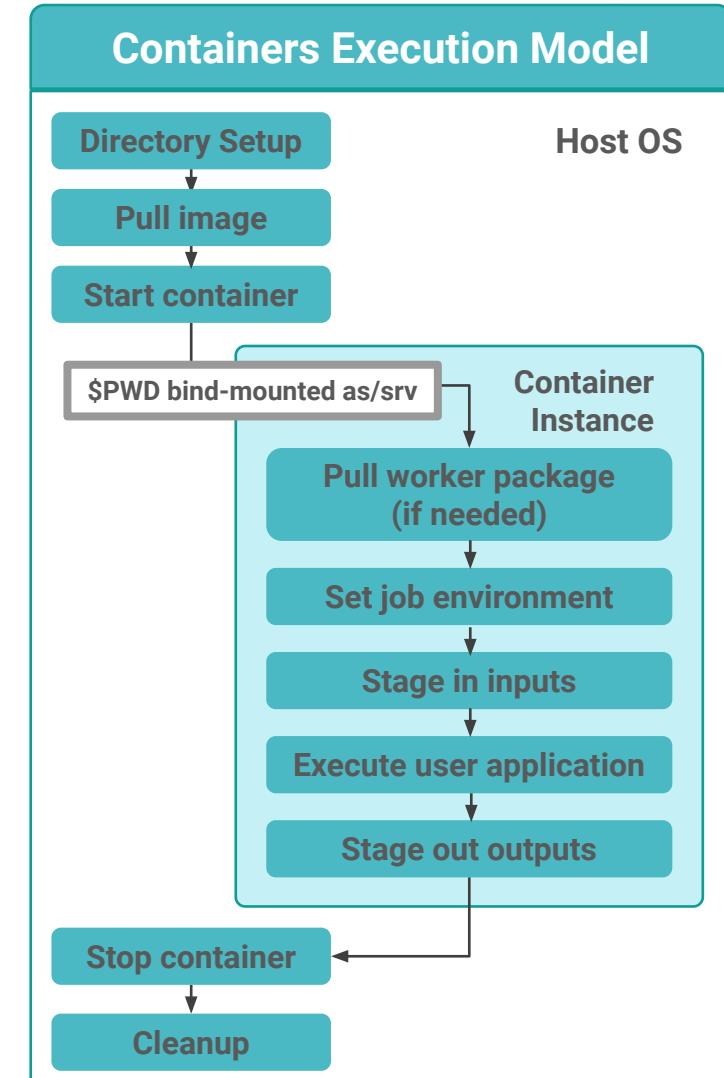
Users can **refer** to a **container** they want to **use – Pegasus stages** their executables and containers to the node

- Useful if you want to use a site recommended/standard container image.
- Users are using generic image with executable staging.



Future Plans

- Users can **specify an image buildfile** for their jobs.
- *Pegasus will build the Docker image as separate jobs in the executable workflow, export them as a tar file and ship them around*



Data Management for Containers



Containers are data too!

Pegasus treats containers as input data dependency

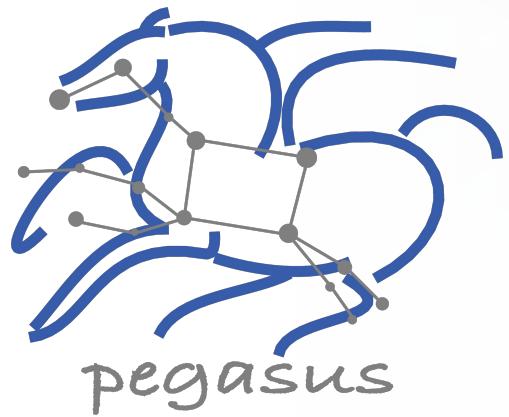
- Staged to compute node if not present
- Docker or Singularity Hub URL's
- Docker Image exported as a TAR file and available at a server, just like any other input dataset

Scaling up for larger workflows

- The image is pulled down as a tar file as part of data stage-in jobs in the workflow
- The exported tar file is then shipped with the workflow and made available to the jobs
- Pricing considerations. You are now charged if you exceed a certain rate of pulls from Hubs

Other Optimizations

- Symlink against existing images on shared file system such as CVMFS
- The exported tar file is then shipped with the workflow and made available to the jobs



2.4 Variant Calling Workflow



Genomics Analysis on HPC cluster

Not only GUI applications

Three reasons why a lot of genomics analysis is done using command-line tools:

- a large number of files - the command-line allows you to automate repetitive tasks,
- more compute power than is available on your local computer - remote computers require a command-line interface
- custom analyses - command-line tools often enable more customization than the corresponding GUI tools (if in fact a GUI tool even exists)

The tools used in example:

- Burrows-Wheeler Aligner (BWA) 0.7.17
- SamTools 1.15.1
- Bcftools 1.15.1
- HTSlib 1.15.1
- SRA Tools 3.0.0



Genomics Analysis on HPC cluster

Example: Variant Calling

Navigate in the file manager on the left and double-click the notebook to start:

The screenshot shows a Jupyter Notebook interface. On the left, there is a file manager pane displaying a directory structure under "/.../notebooks/05-VariantCalling/". The contents include a folder "ref_genome" (modified 2 days ago), a folder "tools" (modified 2 days ago), and a notebook "05-Variant-Calling.ipynb" (modified 27 minutes ago). The main pane on the right shows a tab titled "osinski@e10-12:~" with the notebook "05-Variant-Calling.ipynb" open. The notebook content starts with a section titled "Variant Calling Pegasus Workflow". The text in this section explains the purpose of the workflow, which is to create a Pegasus Workflow corresponding to the Au variant calling example. It mentions that the workflow downloads and aligns SRA data to the E. coli REL606 reference genome to see how the population changed over time. A note indicates that one major change from other examples is that this workflow uses CARC preprocessed data limited to 100GB. Below this, there is a section titled "Container" with a note stating that all tools required to execute the jobs in the container are included in a specific directory structure under "pegasus/variant-calling". The runtime used can easily be changed in the workflow definition.

Name

Last Modified

File Edit View Run Kernel Tabs Settings Help

Filter files by name

osinski@e10-12:~ 05-Variant-Calling.ipynb

Variant Calling Pegasus Workflow

In this notebook, we will create Pegasus Workflow corresponding to the Au variant calling to see how the population changed over time.

This workflow downloads and aligns SRA data to the E. coli REL606 reference genome.

One major change from other examples is that this workflow uses CARC preprocessed data limited to 100GB.

Container

All tools required to execute the jobs in the container are all included in a specific directory structure under `pegasus/variant-calling`. The workflow is setup up to use that container. The runtime used can easily be changed in the workflow definition.



Genomics Analysis on HPC cluster

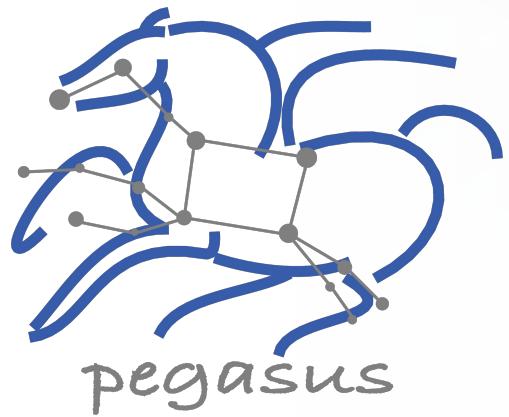
Example: Variant Calling

Reference genome: Escherichia coli B strains REL606 (ecoli_rel606)

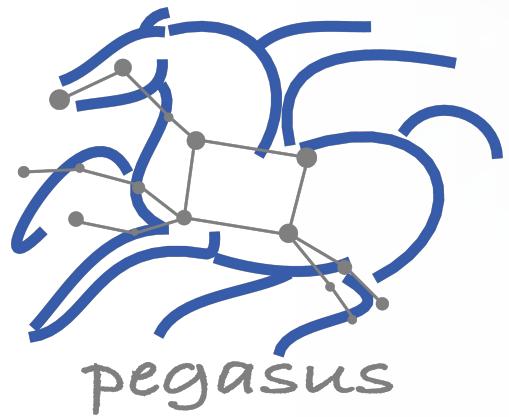
SRA data: SRR2584866, SRR2589044

Tasks performed:

```
fasterq-dump -split-files sra_id  
bwa mem genome fastq_1 fastq_2 > sam_file  
samtools view -S -b sam_file > bam_file  
samtools sort -o sorted_bam bam_file  
samtools index sorted_bam  
bcftools mpileup -O b -o raw_bcf -f genome sorted_bam  
bcftools call --ploidy 1 -m -v -o variants raw_bcf  
vcfutils.pl varFilter variants > final_variants
```



2.5 Summary



3. Advanced Topics



Data Staging Configurations

HTCondor I/O (HTCondor pools, OSG, ...)

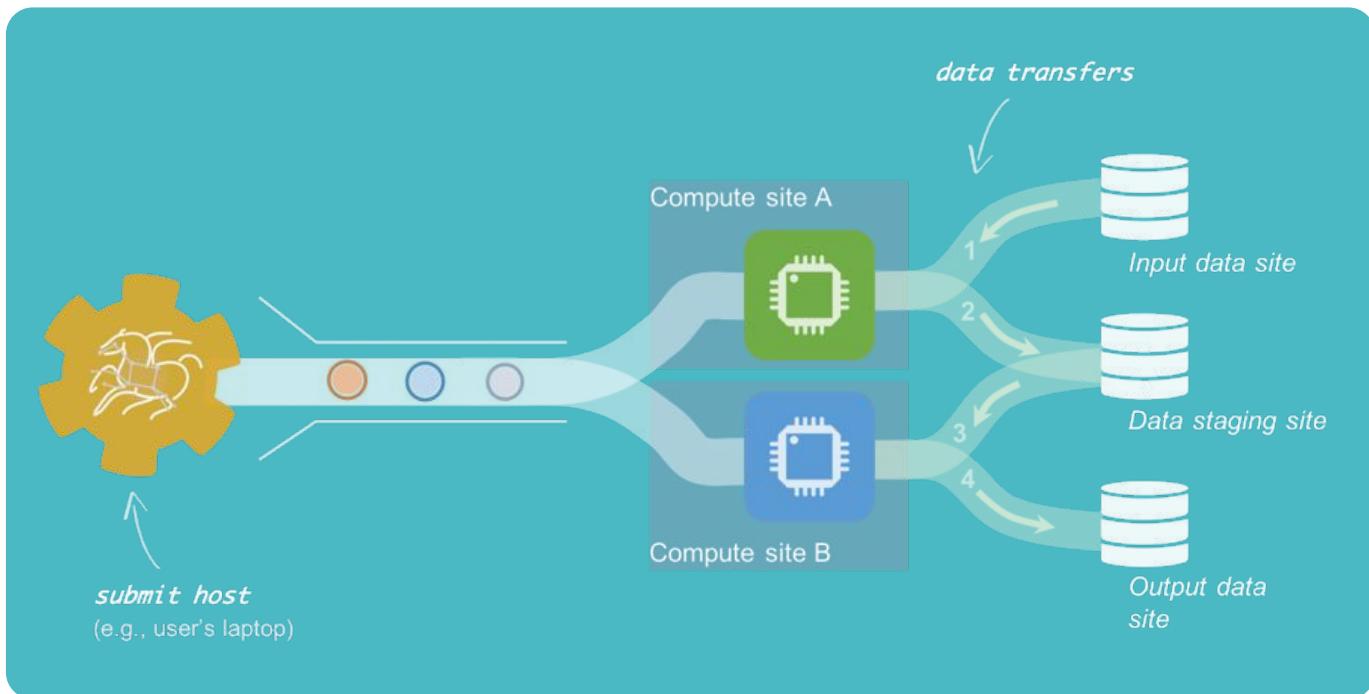
- Worker nodes do not share a file system
- Data is pulled from / pushed to the submit host via HTCondor file transfers
- Staging site is the submit host

Non-shared File System (clouds, OSG, ...)

- Worker nodes do not share a file system
- Data is pulled / pushed from a staging site, possibly not co-located with the computation

Shared File System (HPC sites, XSEDE, Campus clusters, ...)

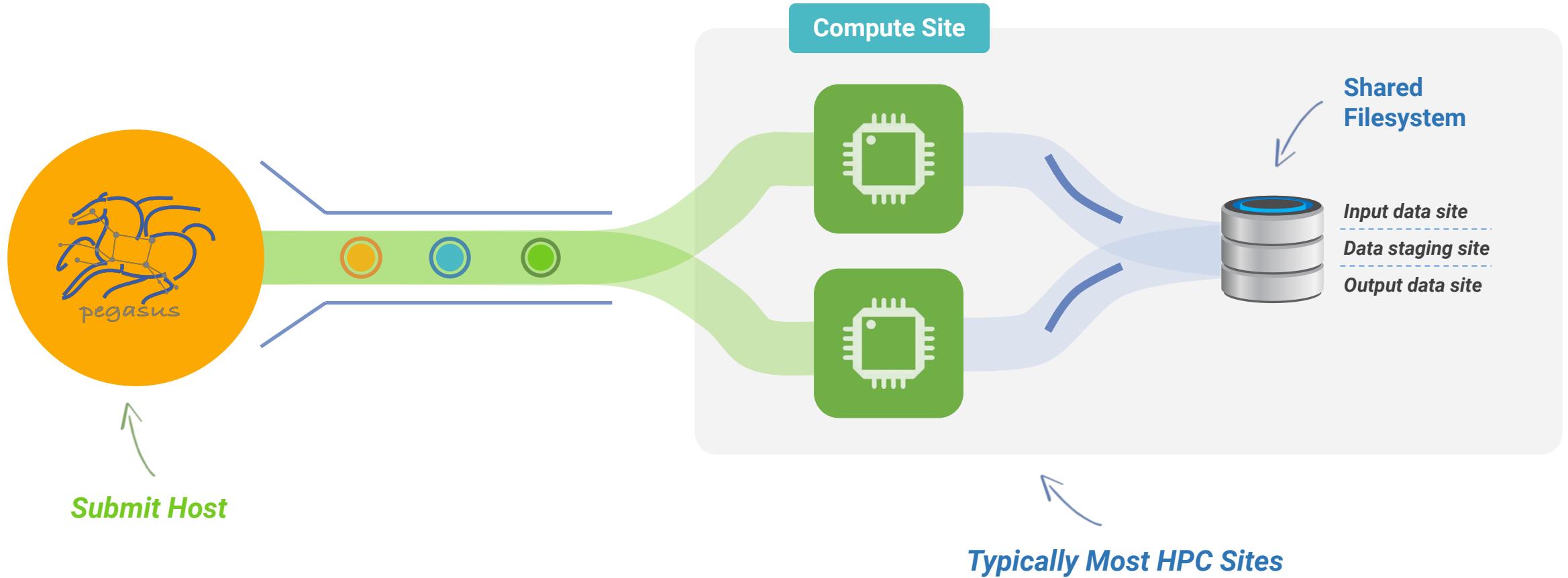
- I/O is directly against the shared file system





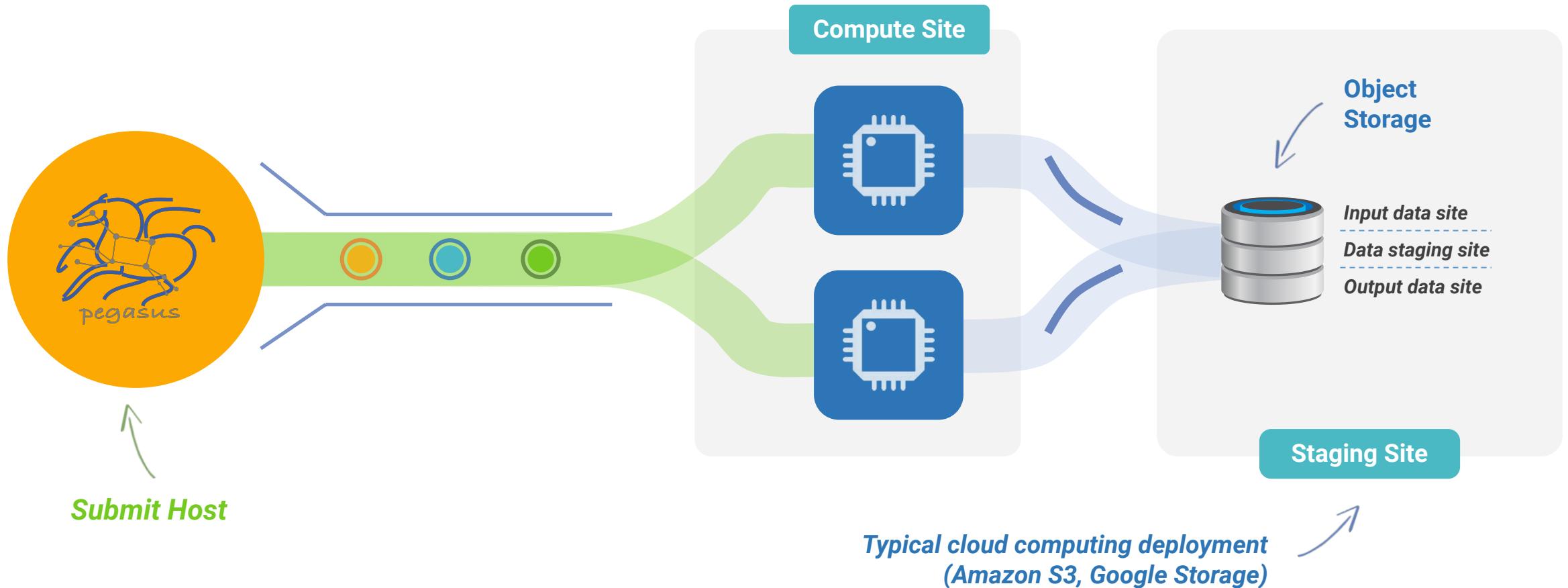
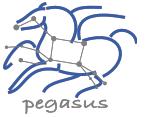
High Performance Computing

There are several possible configurations...



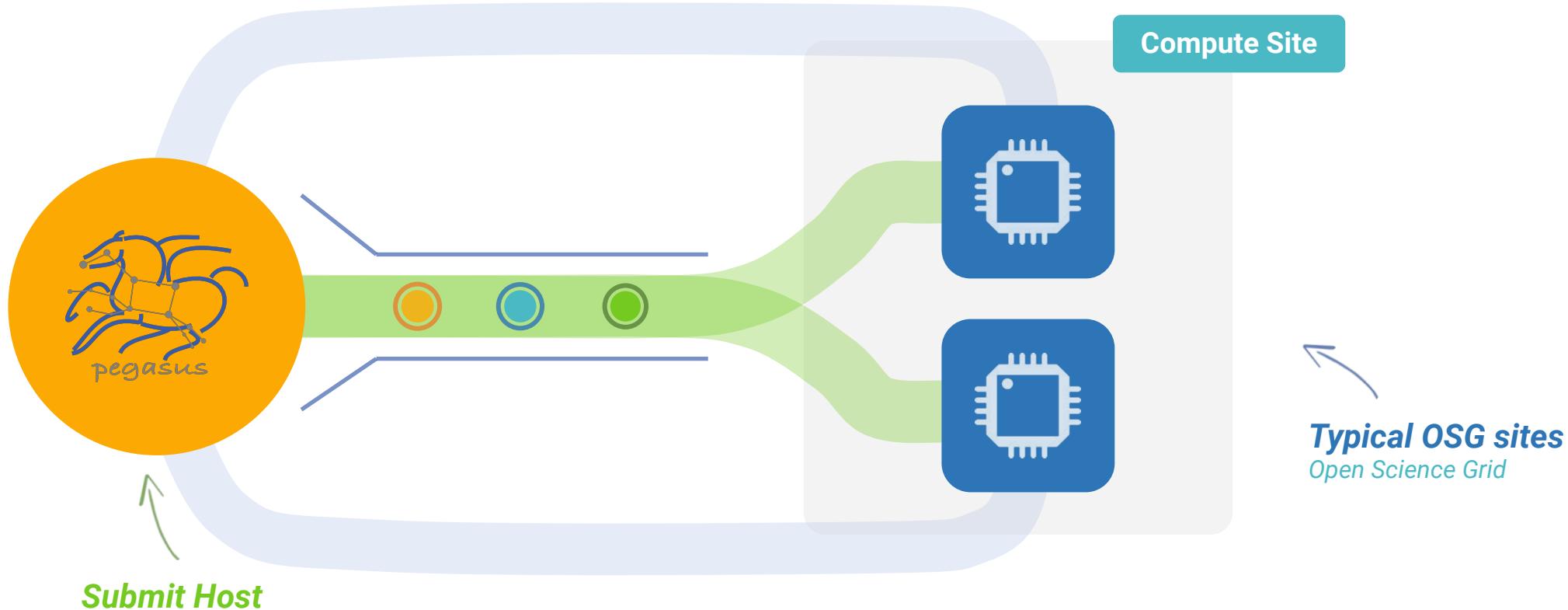
Cloud Computing

High-scalable object storages

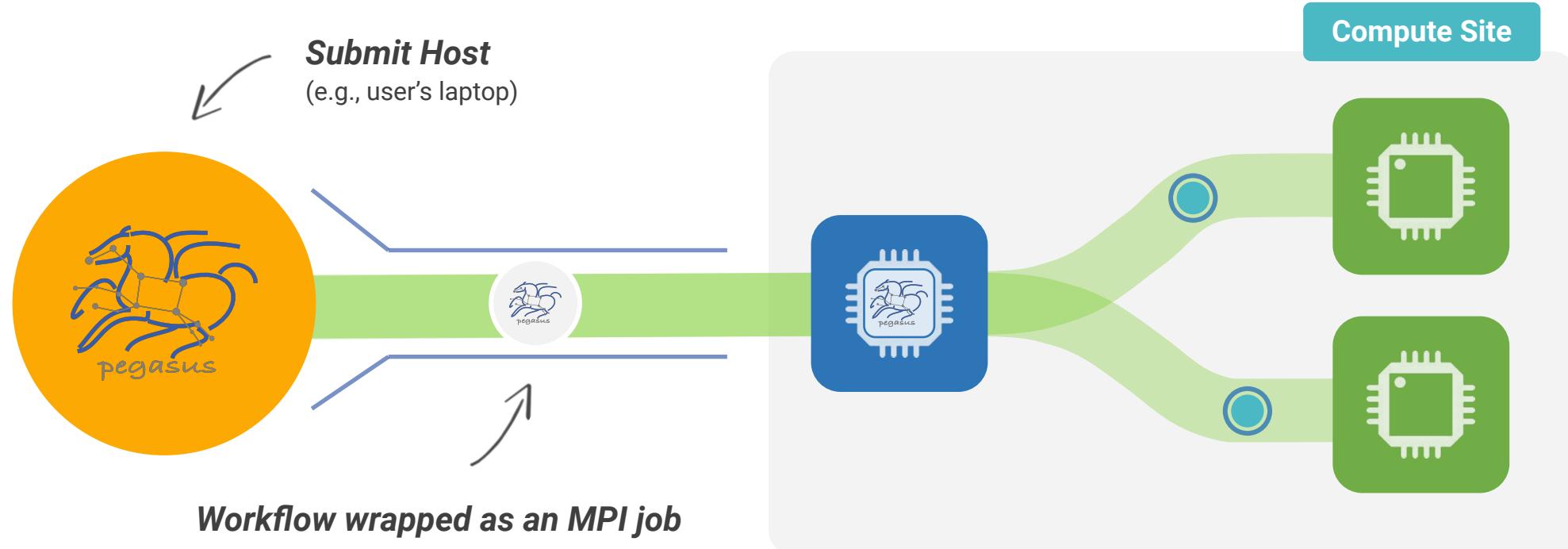


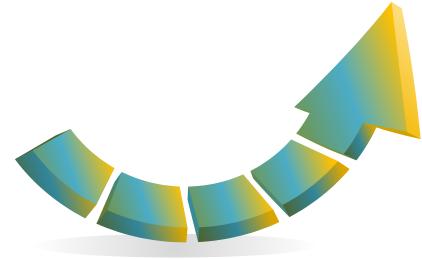
Grid Computing

Local data management



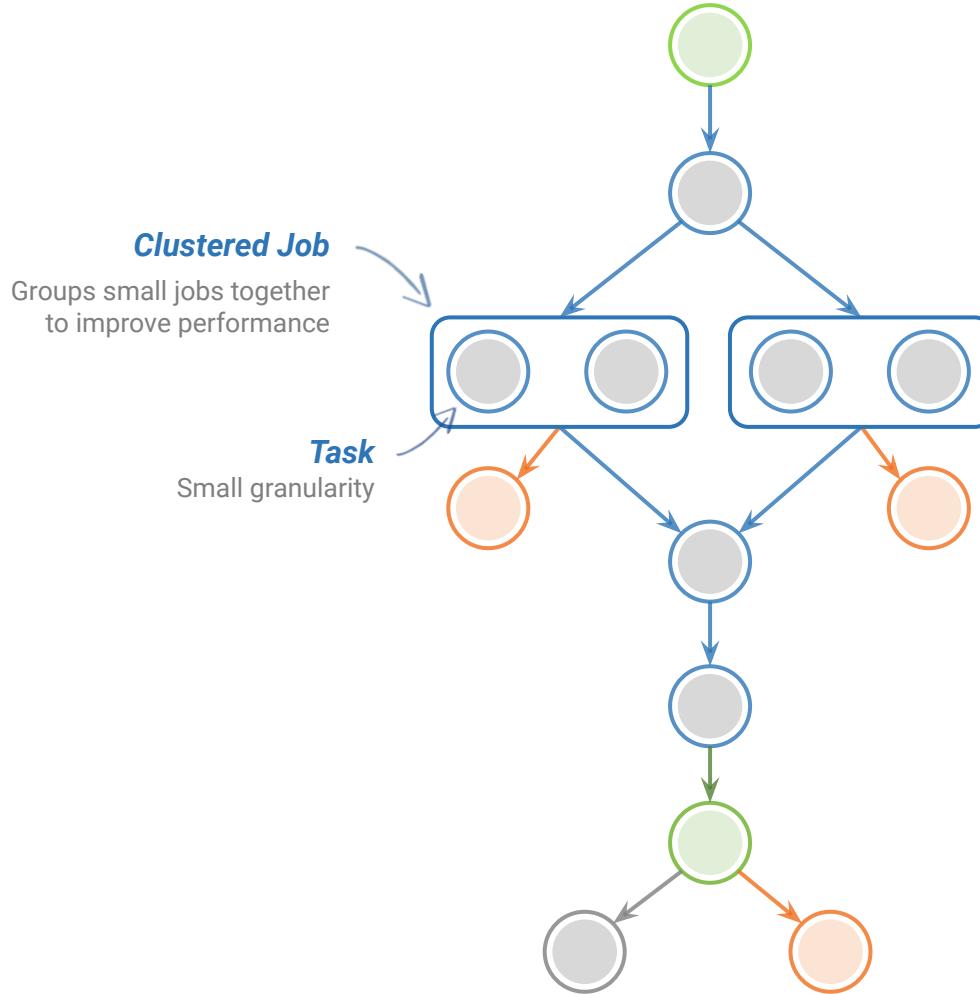
Running fine-grained workflows on HPC systems...





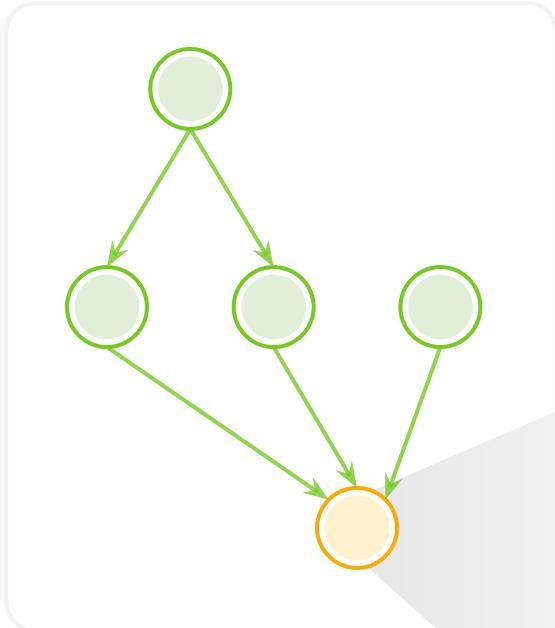
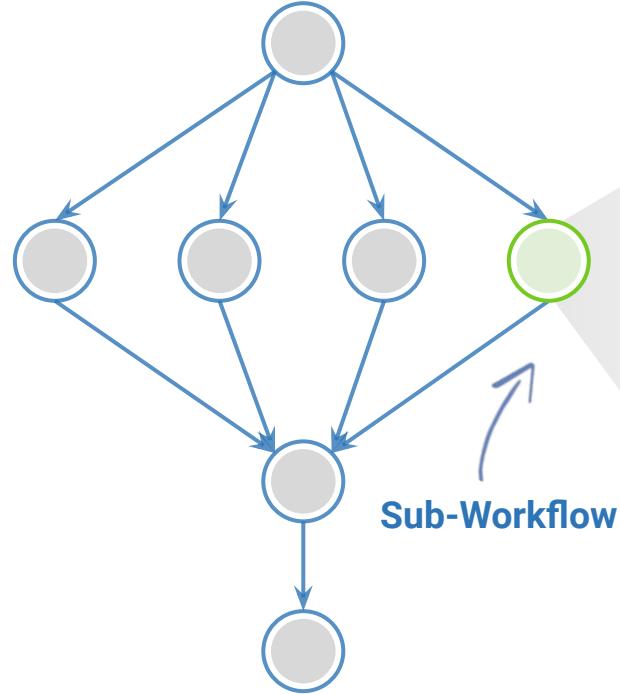
Performance.

Why not improve it?

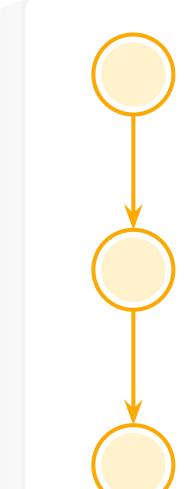




Pegasus also handles large-scale workflows

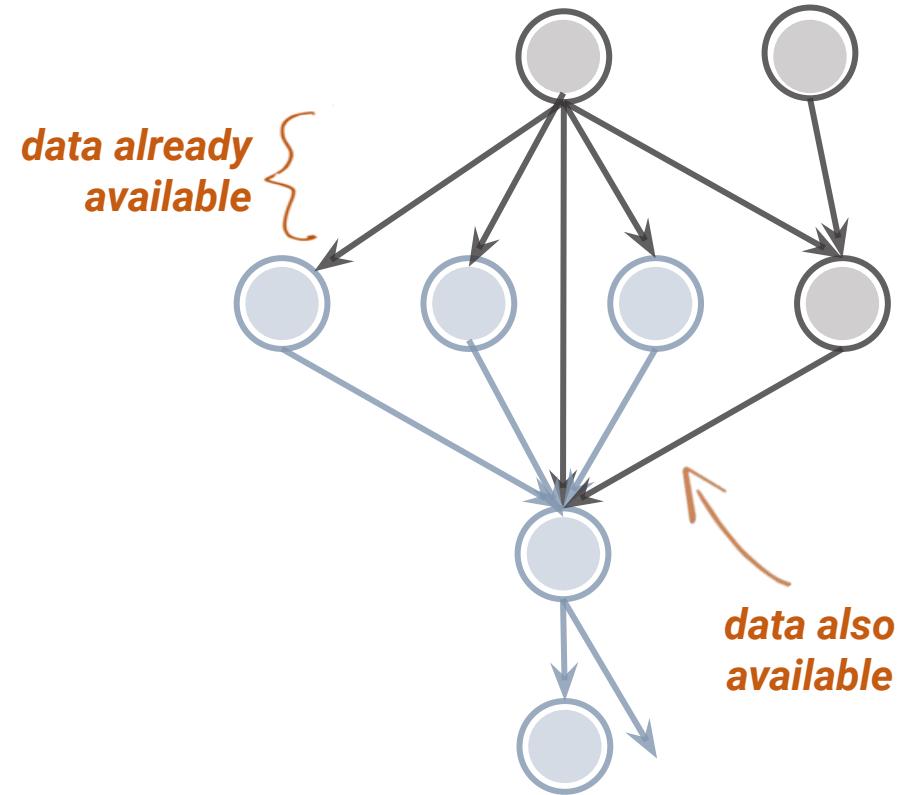


Sub-Workflow

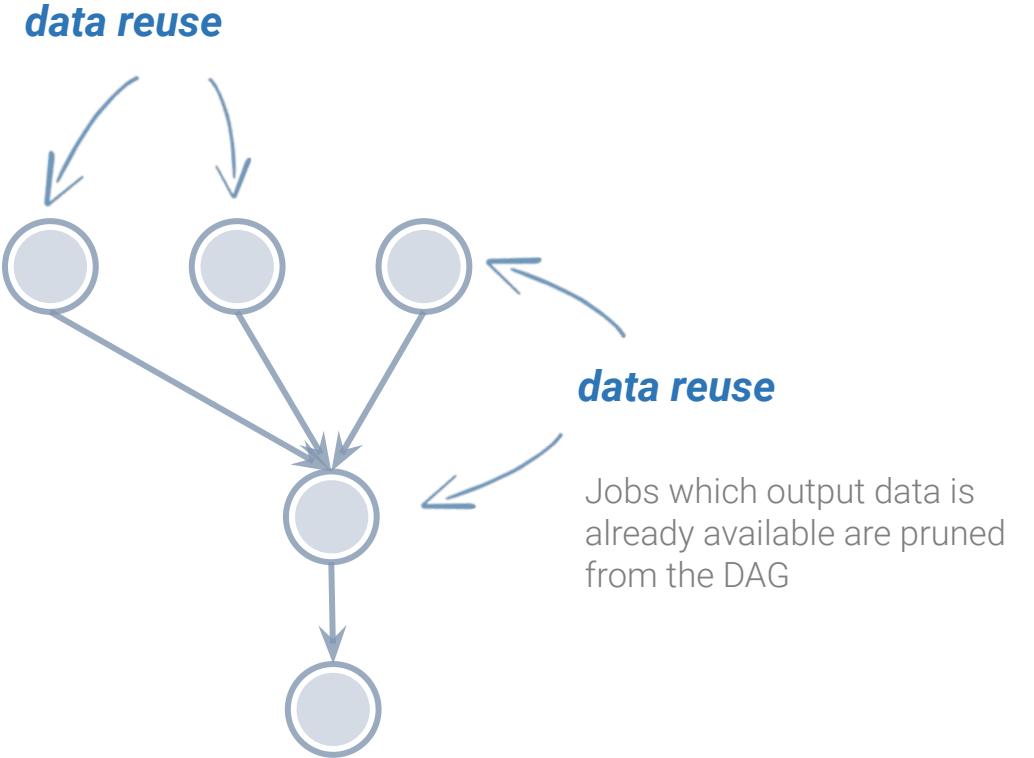


Recursion ends
When abstract
workflow with
only compute jobs
is encountered

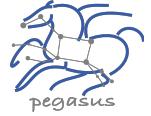
Data Reuse prune jobs if output data already exists



*workflow
reduction*



And if a job fails?



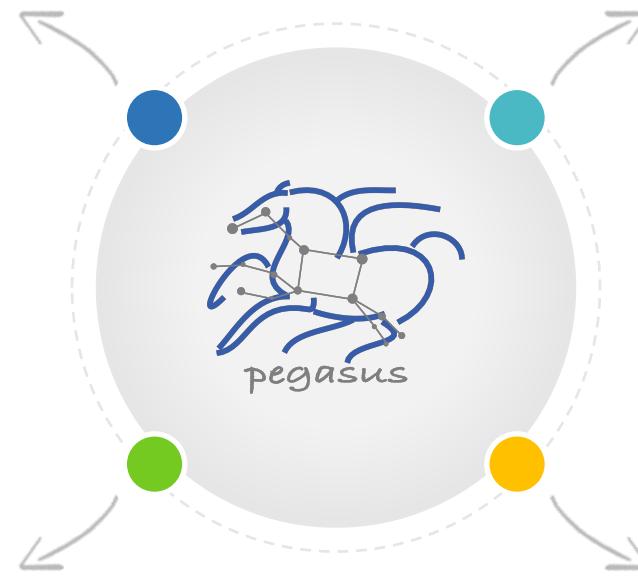
Postscript

detects non-zero exit code output
parsing for success or failure
message exceeded timeout do not
produced expected output files



Checkpoint Files

job generates checkpoint files
staging of checkpoint files is
automatic on restarts



Job Retry



helps with transient failures
set number of retries per job
and run



Rescue DAGs

workflow can be restarted from
checkpoint file recover from
failures with minimal loss



Metadata

Can associate arbitrary key-value pairs with workflows, jobs, and files

Data Registration

Output files get tagged with metadata on registration in the workflow database

Workflow,
Job, File

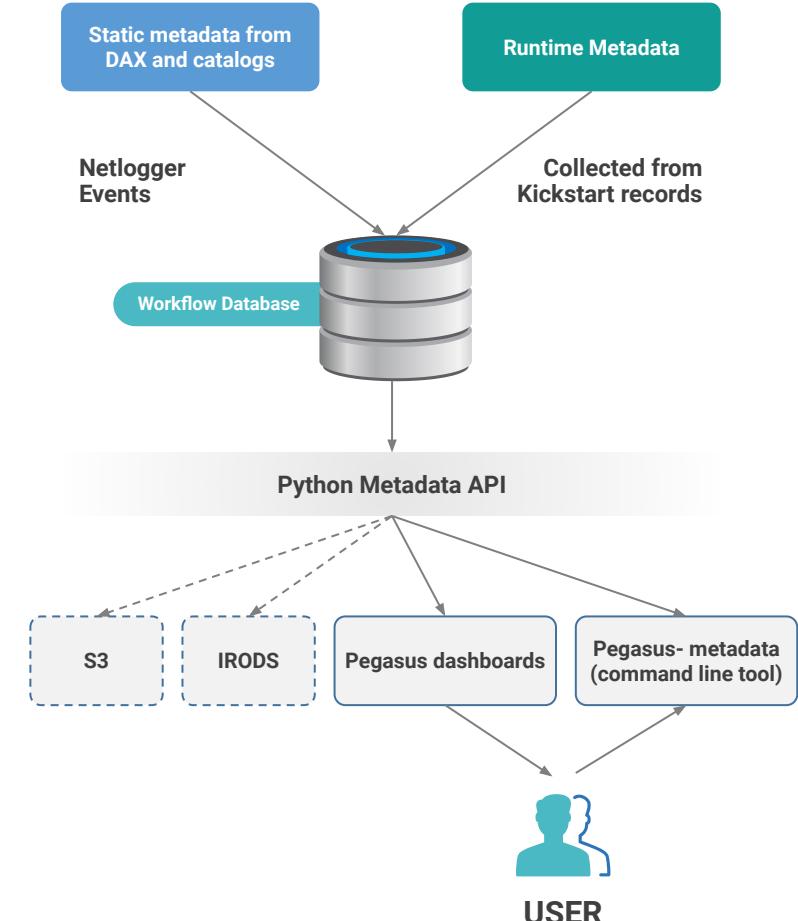
```
x-pegasus:  
apiLang: python  
createdBy: vahi  
createdOn: 12-08-20T10:08:48Z  
pegasus: "5.0"  
name: diamond  
metadata:  
    experiment:"par_all27_prot_lipid"  
jobs:  
- type: "job"  
  name: "namd"  
  id: "ID0000001"  
  arguments: ["equilibrate.conf"]  
  uses:  
  - lfn: "Q42.psf"  
    metadata:  
      type: "psf"  
      charge: "42"  
  type: "input"  
  - lfn: "eq.restart.coord"  
    type: "output"  
    metadata:  
      type: "coordinates"  
      stageOut: true  
      registerReplica: true  
  metadata:  
    timesteps:500000  
    temperature:200  
    pressure:1.01353
```

Static and Runtime Metadata

Static: application parameters
Runtime: performance metrics

Select Data
Based on Metadata

Register Data
With Metadata





Challenges to Scientific Data Integrity

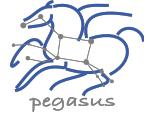
**Modern IT systems
are not perfect
- errors creep in.**

At modern “**Big Data**” sizes we
are starting to see checksums
breaking down.

**Plus there is the threat
of intentional changes:
*malicious attackers,
insider threats, etc.***

User Perception: “Am I not already protected? I have heard about TCP checksums, encrypted transfers, checksum validation, RAID and erasure coding – is that not enough?”

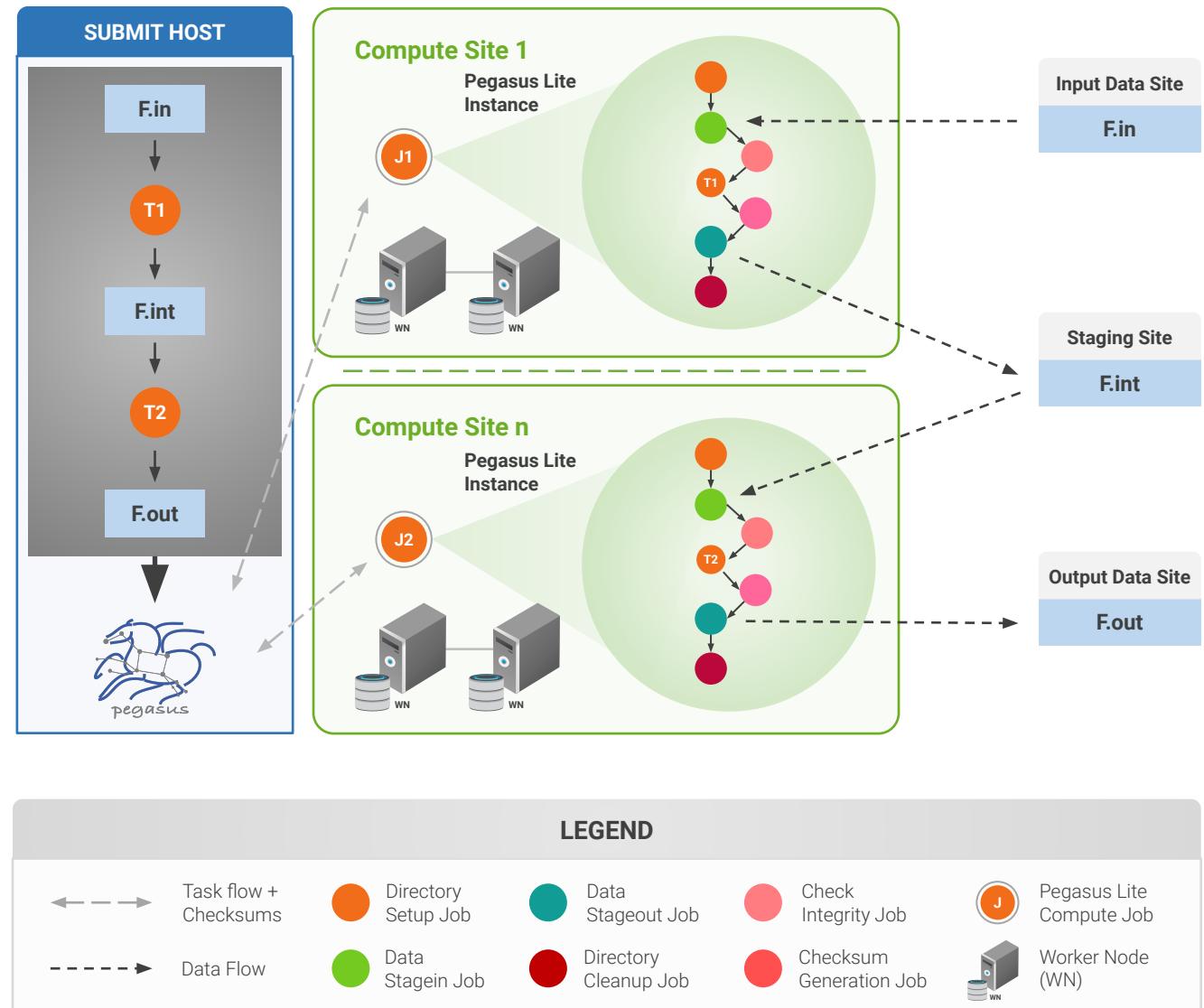
Automatic Integrity Checking in Pegasus



Pegasus performs integrity checksums on input files right before a job starts on the remote node.

- ▶ For raw inputs, **checksums specified in the input replica catalog** along with file locations
- ▶ All **intermediate** and **output** files checksums are generated and tracked within the system.
- ▶ Support for **sha256** checksums

Job failure is triggered if checksums fail



Job Submissions



LOCAL

Submit Machine

Personal HTCondor

Local Campus Cluster accessible via Submit Machine **

HTCondor via BLAHP

**** Both Glite and BOSCO build on HTCondor BLAHP**

Currently supported schedulers:
SLURM SGE PBS MOAB

REMOTE

BOSCO + SSH**

Each node in executable workflow submitted via SSH connection to remote cluster

BOSCO based Glideins**

SSH based submission of glideins

PyGlidein

IceCube glidein service

OSG using glideinWMS

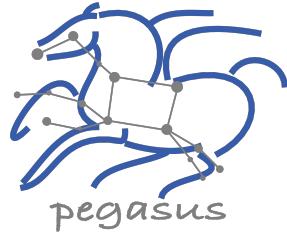
Infrastructure provisioned glideins

CREAMCE

Uses CondorG

Globus GRAM

Uses CondorG



Pegasus

est. 2001

Automate, recover, and debug scientific computations.

► Get Started

► Pegasus Website

<https://pegasus.isi.edu>

► Users Mailing List

pegasus-users@isi.edu

► Support

pegasus-support@isi.edu

► Slack

Ask for an invite by trying to join pegasus-users.slack.com in the Slack app

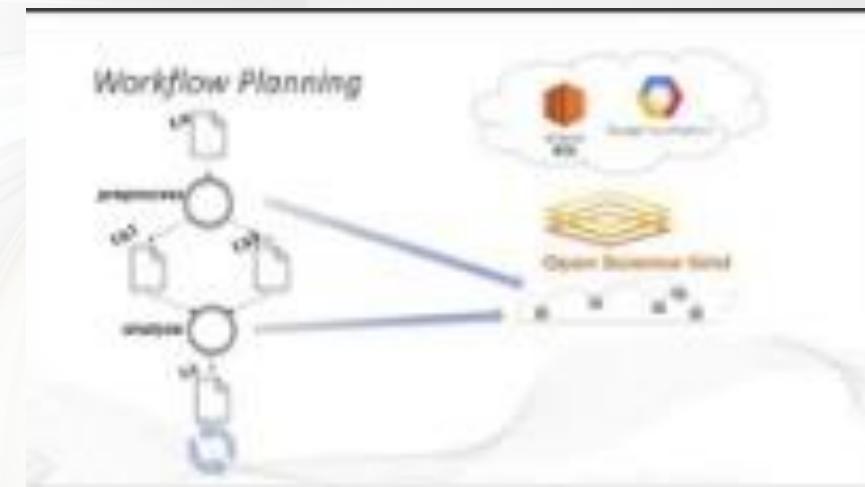
► Pegasus Online Office Hours

<https://pegasus.isi.edu/blog/online-pegasus-office-hours/>



YouTube Channel

<https://www.youtube.com/channel/UCwJQIn1CqBvTJqiNr9X9F1Q/featured>



[Pegasus in 5 Minutes](#)

Bi-monthly basis on second Friday of the month, where we address user questions and also apprise the community of new developments