# Pegasus - a dHTC friendly workflow manager

**Mats Rynge**
rynge@isi.edu

USC Viterbi
School of Engineering
*Information Sciences Institute*

https://pegasus.isi.edu

# Pegasus Concepts

Users describe their pipelines in a portable format called Abstract Workflow, without worrying about low level execution details.

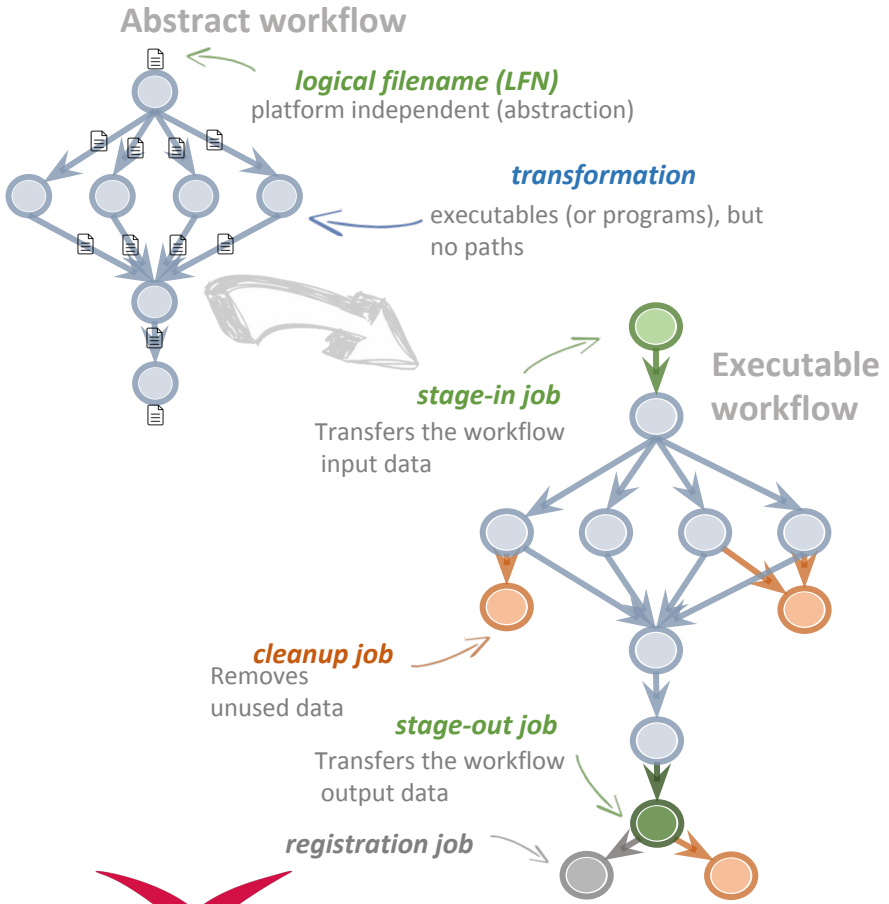
Workflows are DAGs

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

Pegasus takes this and generates an executable workflow

- Data management tasks added
- Transforms the workflow for performance and reliability
- HTCondor DAGMan DAG

Planning occurs before execution




HTCondor
High Throughput Computing

# Pegasus 5.0

Automate, recover, and debug scientific computations

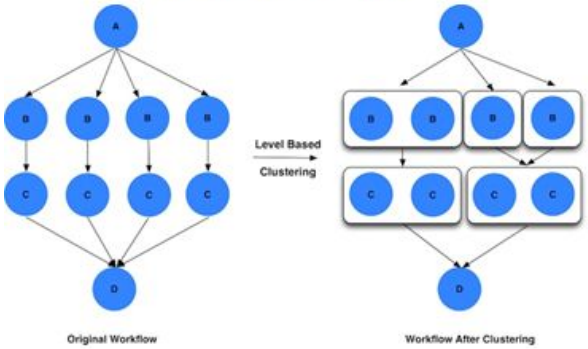
Coming soon! Beta1 is out.

- New and fresh Python3 API to compose, submit and monitor workflows, and configure catalogs
- New Catalog Formats
- Python 3
  - All Pegasus tools are Python 3 compliant
  - Python PIP packages for workflow composition and monitoring
- Zero configuration required to submit to local HTCondor pool.
- Data Management Improvements
  - New output replica catalog that registers outputs including file metadata such as size and checksums
  - Improved support for hierarchical workflows
- Major documentation improvements
  - https://pegasus.isi.edu/docs/5.0.0dev/index.html

```
#!/usr/bin/env python3
import logging
import sys

from Pegasus.api import *

# logs to be sent to stdout
logging.basicConfig(level=logging.DEBUG, stream=sys.stdout)

# --- Transformations -----------------------------------------
echo = Transformation(
        "echo",
        pfn="/bin/echo",
        site="condorpool"
    )

tc = TransformationCatalog()\
        .add_transformations(echo)

# --- Workflow ------------------------------------------------
Workflow("hello-world", infer_dependencies=True)\
    .add_jobs(
        Job(echo)
            .add_args("Hello World")
            .set_stdout("hello.out")
    ).add_transformation_catalog(tc)\
    .plan(submit=True)\
    .wait()
```

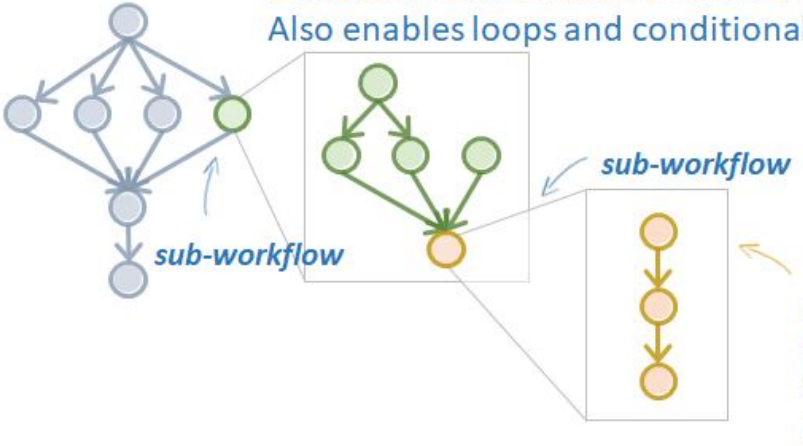
# Optimizations

## Task clustering

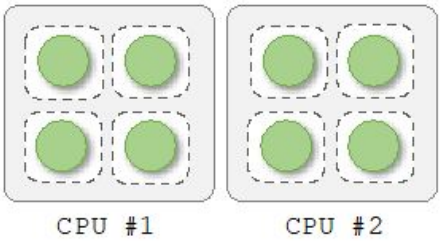



## Hierarchical workflows

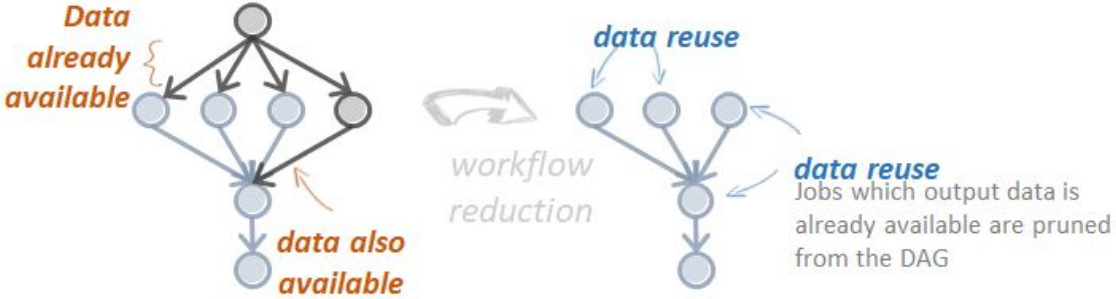
Enacts the execution of **millions of tasks**
Also enables loops and conditionals in DAGs




## Task-resource co-allocation



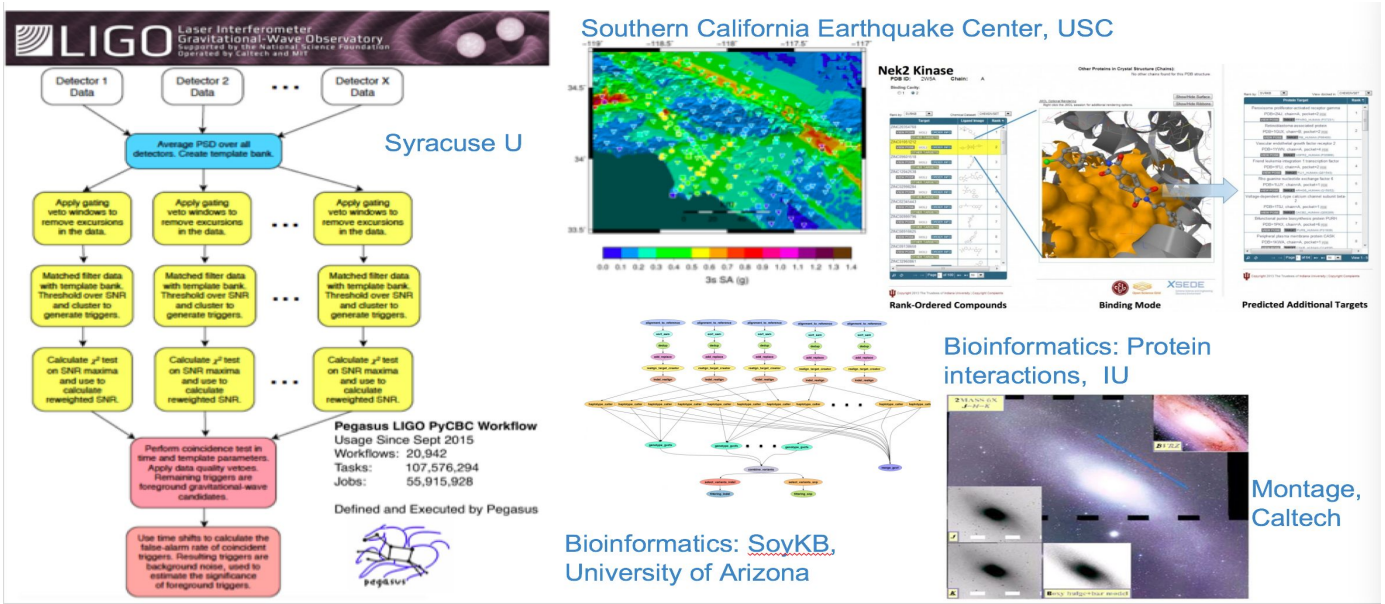

## Data Reuse



# Pegasus Workflow Management System, Production Use

**Last 12 months: Pegasus users ran 240K workflows, 145M jobs**

**Majority of these include data transfers, using LAN, the Internet, local and remote storage**




https://pegasus.isi.edu/

# Data Staging Configurations

**HTCondor I/O** (HTCondor pools, OSG, …)

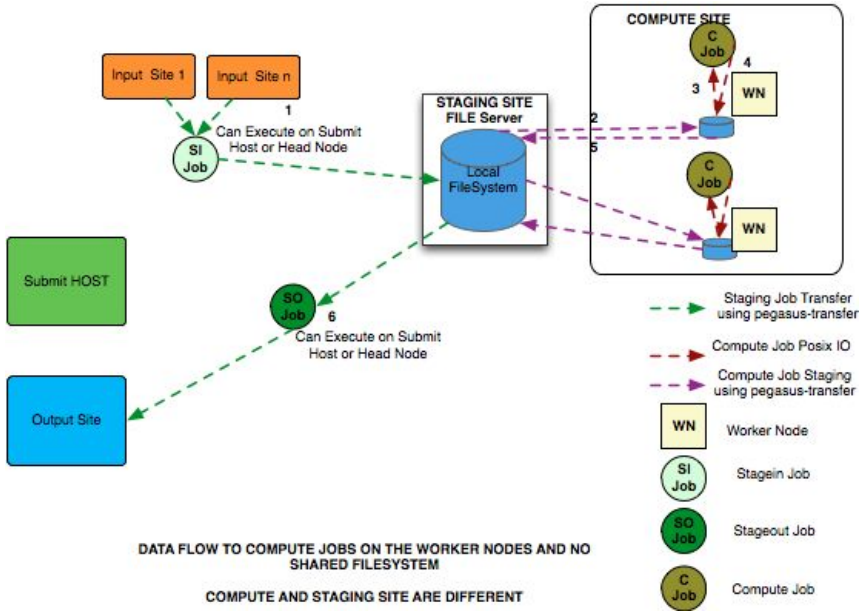
- Worker nodes do not share a file system
- Data is pulled from / pushed to the submit host via HTCondor file transfers
- Staging site is the submit host

**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



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

# pegasus-transfer

Directory creation, file removal

- If protocol can support it, also used for cleanup

Two stage transfers between incompatible protocols

- e.g., GridFTP to S3 is executed as: GridFTP to local file, local file to S3

Parallel transfers

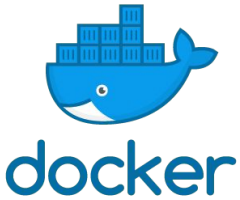
Automatic retries

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
```

# Containers are data too!





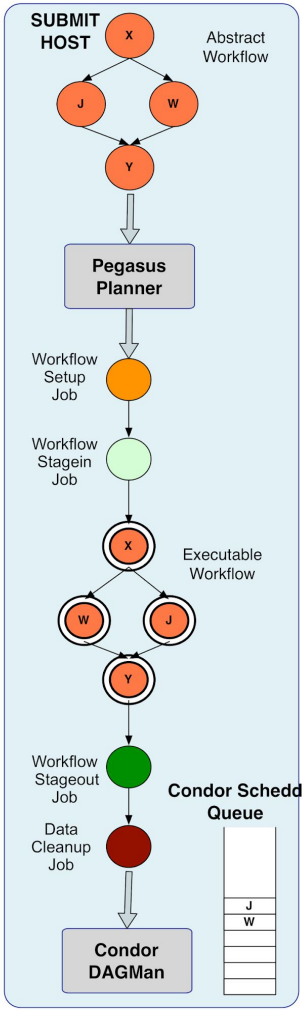
Users can specify to use images from Docker Hub, Singularity Library, or a file using URLs

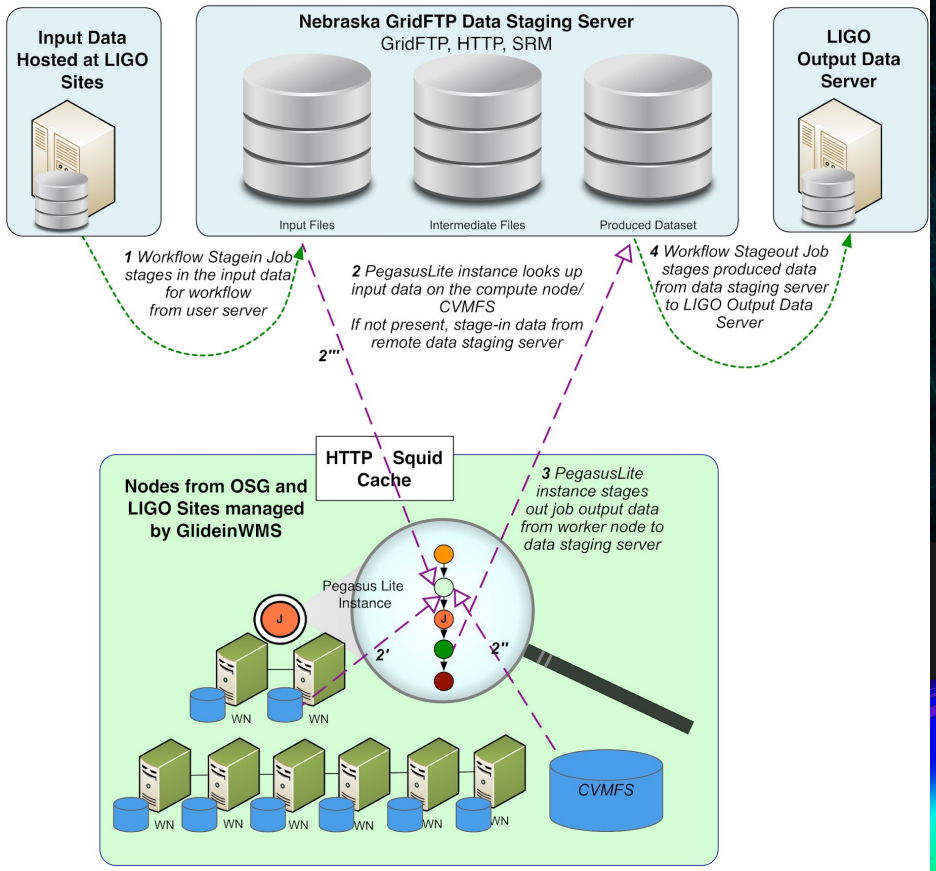
The image is pulled down as a tar file as part of data stage-in jobs in the workflow

- The exported tar file / image file is then transferred to the job as any other piece of data
- Motivation: Avoid overwhelming Docker Hub/Singularity Library/… with by repeatedly requesting the same image
- Motivation: Optimize workflow data placement and movement

Symlink against a container image if available on shared file systems. For example, CVMFS hosted images on Open Science Grid

# Data Flow for LIGO Pegasus Workflows in OSG



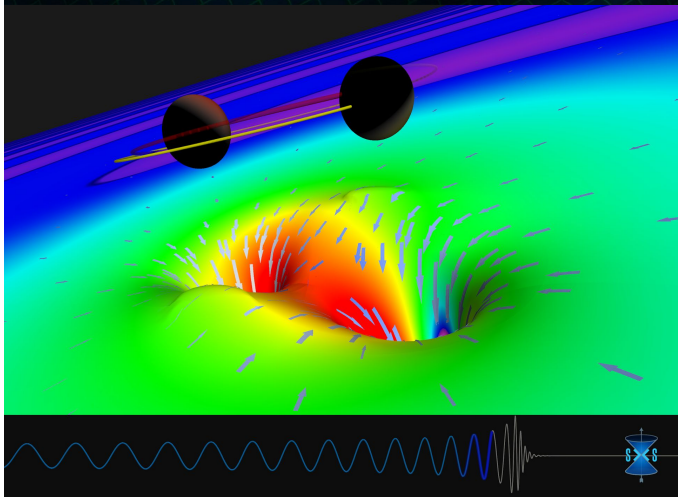

**LEGEND**

- Directory Setup Job
- Data Stageout Job
- Pegasus Lite Compute Job
- Data Stagein Job
- Directory Cleanup Job
- Worker Node

## Advanced LIGO – Laser Interferometer Gravitational Wave Observatory

40,000 compute tasks
Inputs files: 1,100
Output files: 63
Processed Data: 725 GB

Executing on LIGO Data Grid, EGI, Open Science Grid and XSEDE

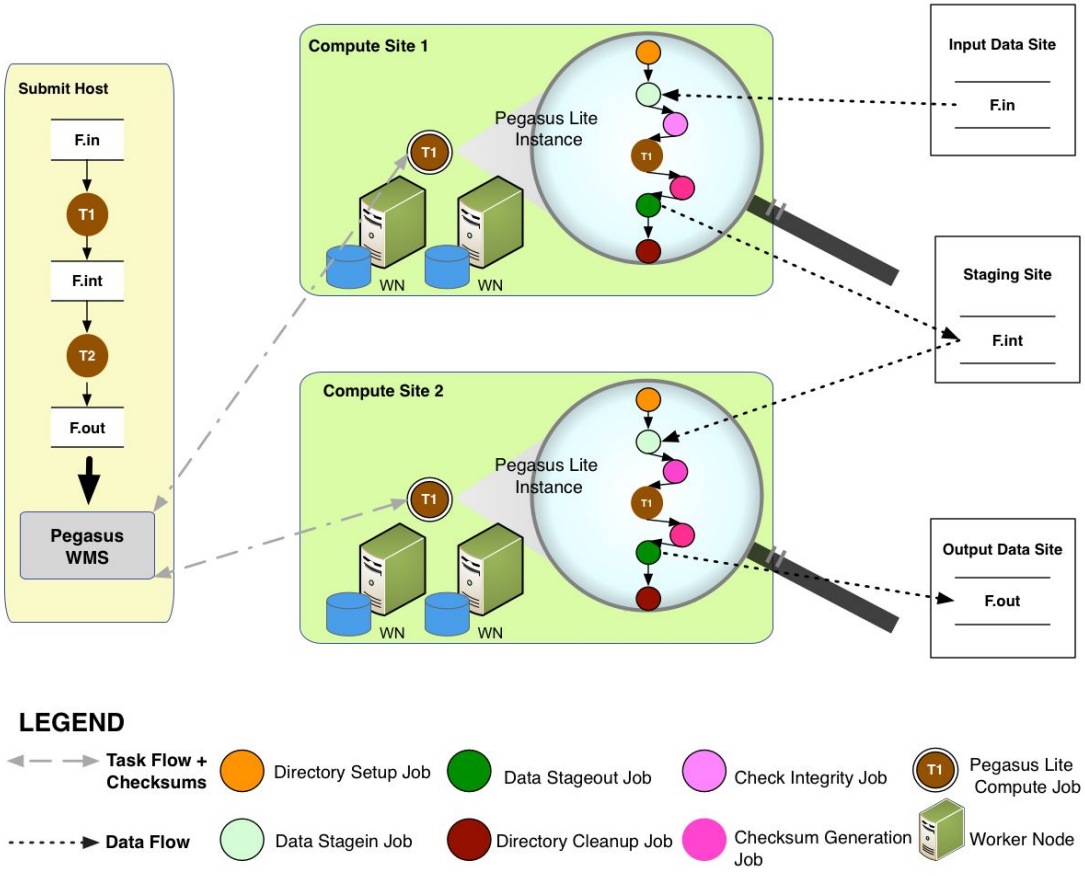


# Automatic Integrity Checking

Pegasus performs integrity checksums on input files right before a job starts, ensuring the computation is on the expected piece of data

- For inputs from external sources, checksums specified in the input replica catalog along with file locations, or generated first time we encounter the file
- All intermediate and output files checksums are generated and tracked within the system.

Checksums validation failures is a job failure



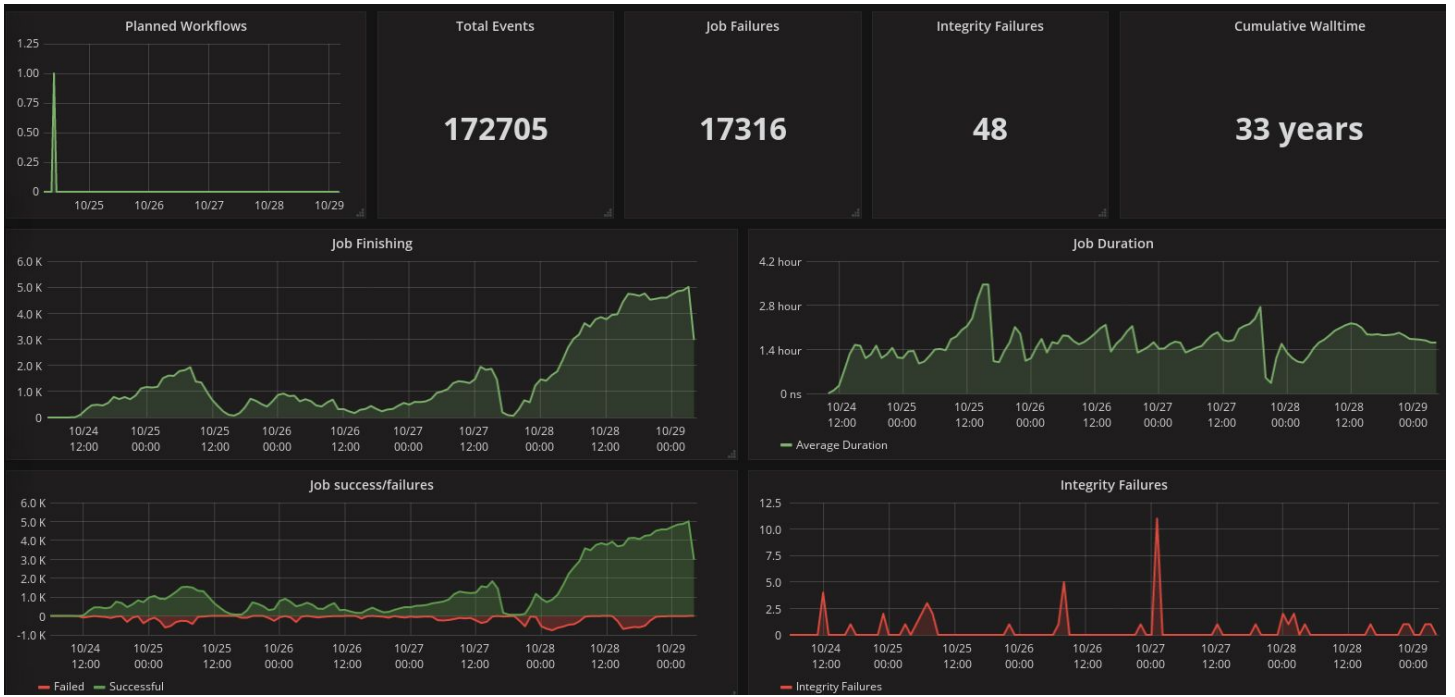
# VERITAS / Nepomuk Otte, GATech

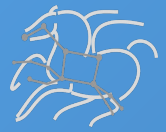
Seeing very small, but steady stream of corrected integrity errors from reporting back to Pegasus dashboard.

For VERITAS, ~.04% of transfers fail with integrity errors. (~1 / 2,500 transfers)

Cause uncertain (diagnosis is harder than detection).

Possibly errors in http based transfers (s3 protocol against CEPH)





# Pegasu

est. 2001

Automate, recover, and debug scientific computations.

S

## Get Started

**Pegasus Website**
https://pegasus.isi.edu

**Users Mailing List**
pegasus-users@isi.edu

**Support**
pegasus-support@isi.edu

**Pegasus Online Office Hours**
https://pegasus.isi.edu/blog/online-pegasus-office-hours/

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

# Initial Results with Integrity Checking on

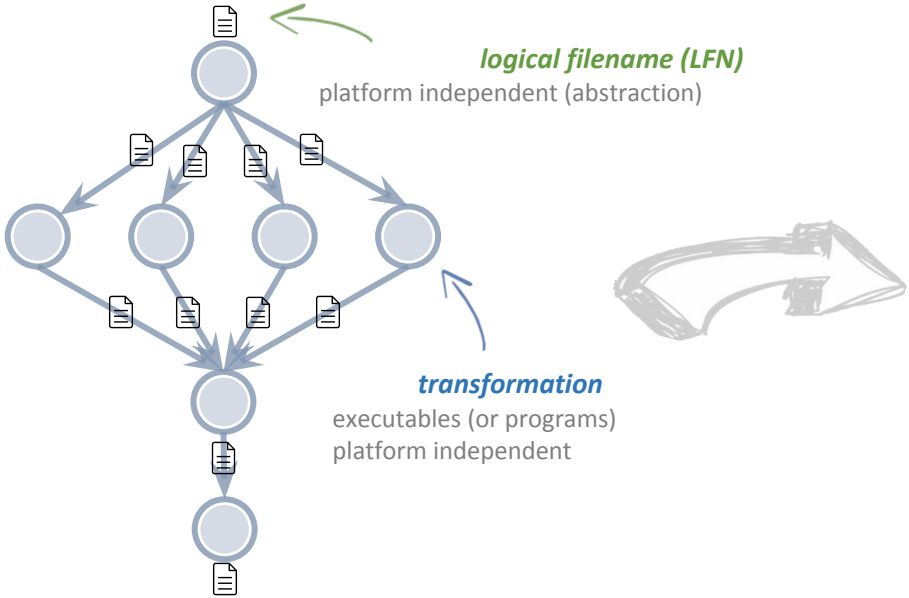
- OSG-KINC workflow (50,606 jobs) encountered 60 integrity errors in the wild (production OSG). The problematic jobs were automatically retried and the workflow finished successfully.
- The 60 errors took place on 3 different hosts. The first one at UColorado, and group 2 and 3 at UNL hosts.
- Error Analysis (by hand)
  - 1 input file error at University of Colorado.
  - 3 input file (kinc executable) errors on one node at University of Nebraska. The timespan across the failures was 16 seconds. We suspect that the node level cache got corrupted.
  - 56 input file errors on a different compute nodes at University of Nebraska. The timespan across the failures was 1,752 seconds. We suspect that the site level cache got corrupted.

# Abstract

API

**Portable Description**

Users do not worry about
low level execution details



***logical filename (LFN)***

platform independent (abstraction)

***transformation***

executables (or programs)
platform independent

# Executable

DAGMan

***stage-in job***

Transfers the workflow input data

***cleanup job***

Removes unused data

***stage-out job***

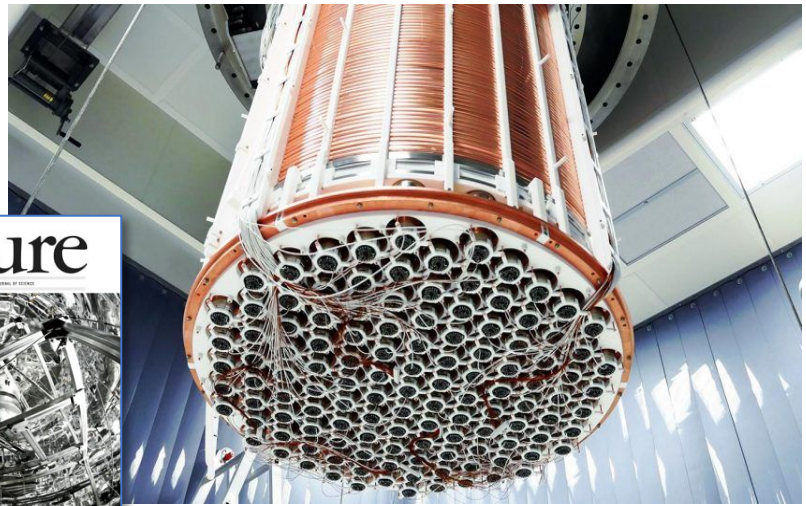
Transfers the workflow output data

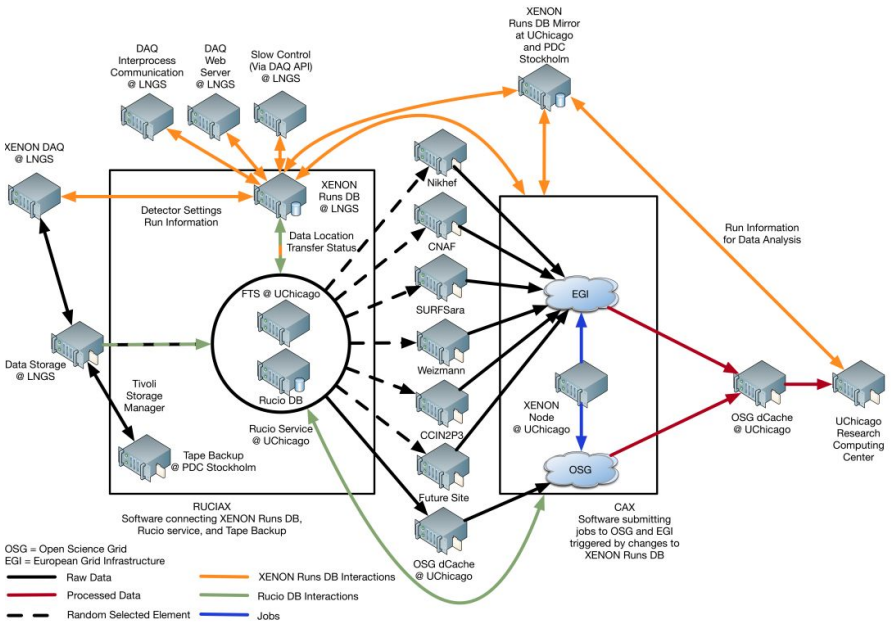
***registration job***

Registers the workflow output data

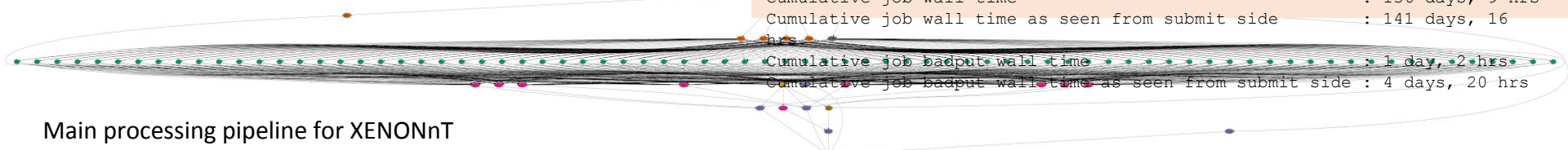
# *XENONnT - Dark Matter Search*

Detector at Laboratori Nazionali del Gran Sass (LNGS) in Italy. Data is distributed world-wide with Rucio. Workflows execute across Open Science Grid (OSG) and European Grid Infrastructure (EGI)




```
------------------------------------------------------------------------
----
Type           Succeeded Failed  Incomplete  Total     Retries
Total+Retries
Tasks          4000      0       0           4000      267       4267
Jobs           4484      0       0           4484      267       4751
Sub-Workflows  0         0       0           0         0         0
------------------------------------------------------------------------
----

Workflow wall time                                     : 5 hrs, 2 mins
Cumulative job wall time                               : 136 days, 9 hrs
Cumulative job wall time as seen from submit side      : 141 days, 16
hrs
Cumulative job badput wall time                        : 1 day, 2 hrs
Cumulative job badput wall time as seen from submit side : 4 days, 20 hrs
```



Main processing pipeline for XENONnT

# *Why* Pegasus?

**Automates** complex, multi-stage processing pipelines

Enables parallel, **distributed or remote computations**

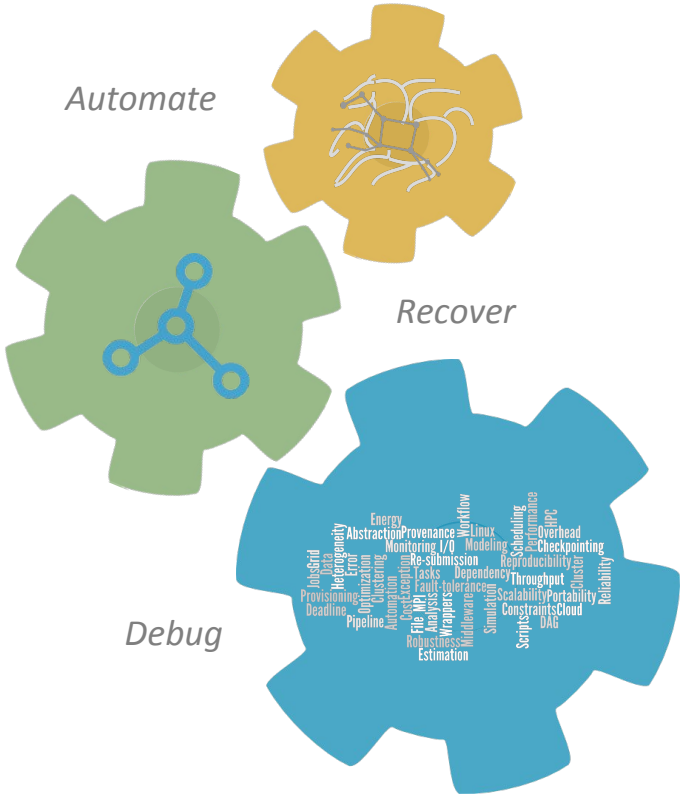
Automatically executes **data transfers**

Reusable, aids **reproducibility**

Records how data was produced (**provenance**)

Handles **failures** with to provide reliability

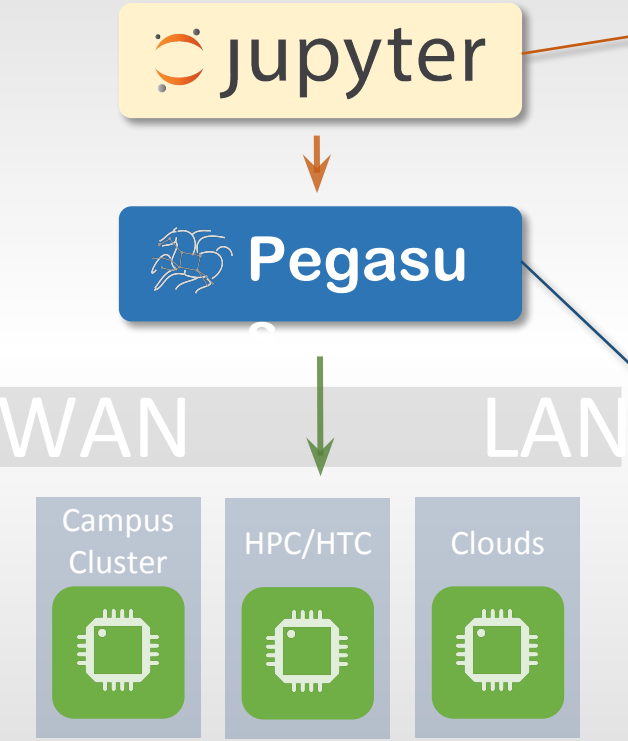
Keeps track of data and **data integrity**

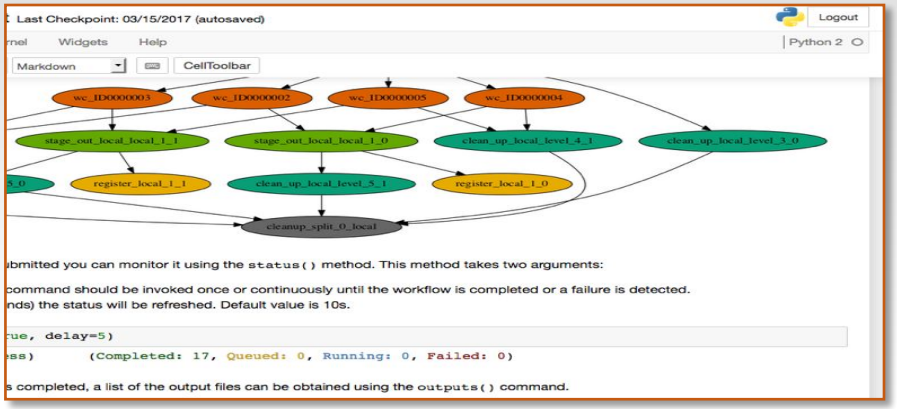


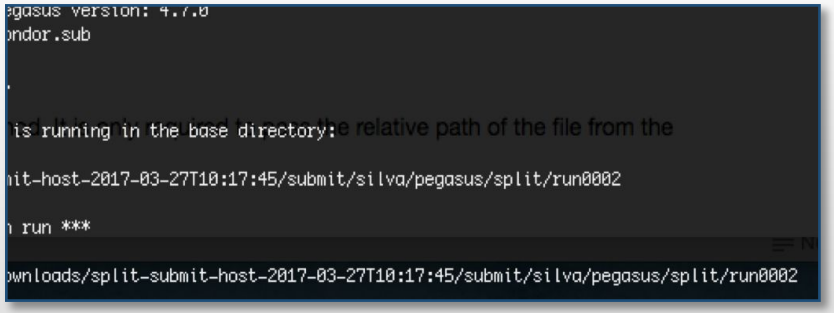

HTCondor High Throughput Computing

NSF funded project since 2001, with close collaboration with HTCondor team

# Running Pegasus workflows with Jupyter



# command-line

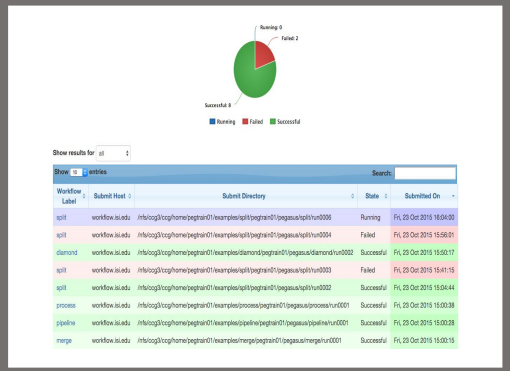
```
$ pegasus-status pegasus/examples/split/run0001
STAT IN_STATE JOB
Run 00:39 split-0 (/home/pegasus/examples/split/run0001)
Idle 00:03  └─split_ID0000001
Summary: 2 Condor jobs total (I:1 R:1)

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 :
```

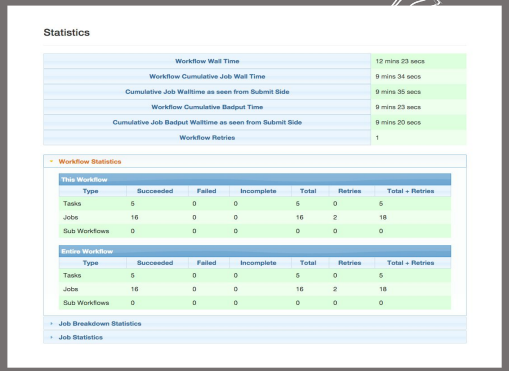
**Provenance data can be summarized**
**pegasus-statistics**

**or used for debugging**
**pegasus-analyzer**

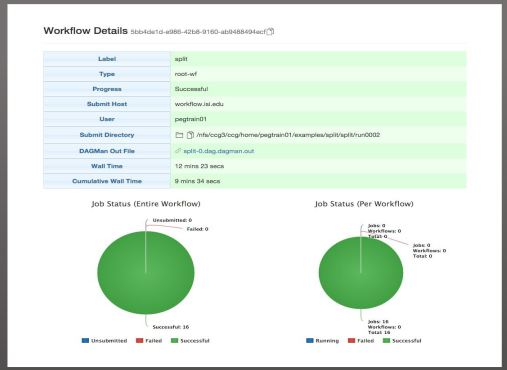



# Pegasus dashboard

web interface for monitoring and debugging workflows




Real-time <u>monitoring</u> of workflow executions. It shows the <u>status</u> of the workflows and jobs, job <u>characteristics</u>, <u>statistics</u> and <u>performance</u> metrics. <u>Provenance</u> data is stored into a relational database.




Real-time Monitoring

Reporting

Debugging

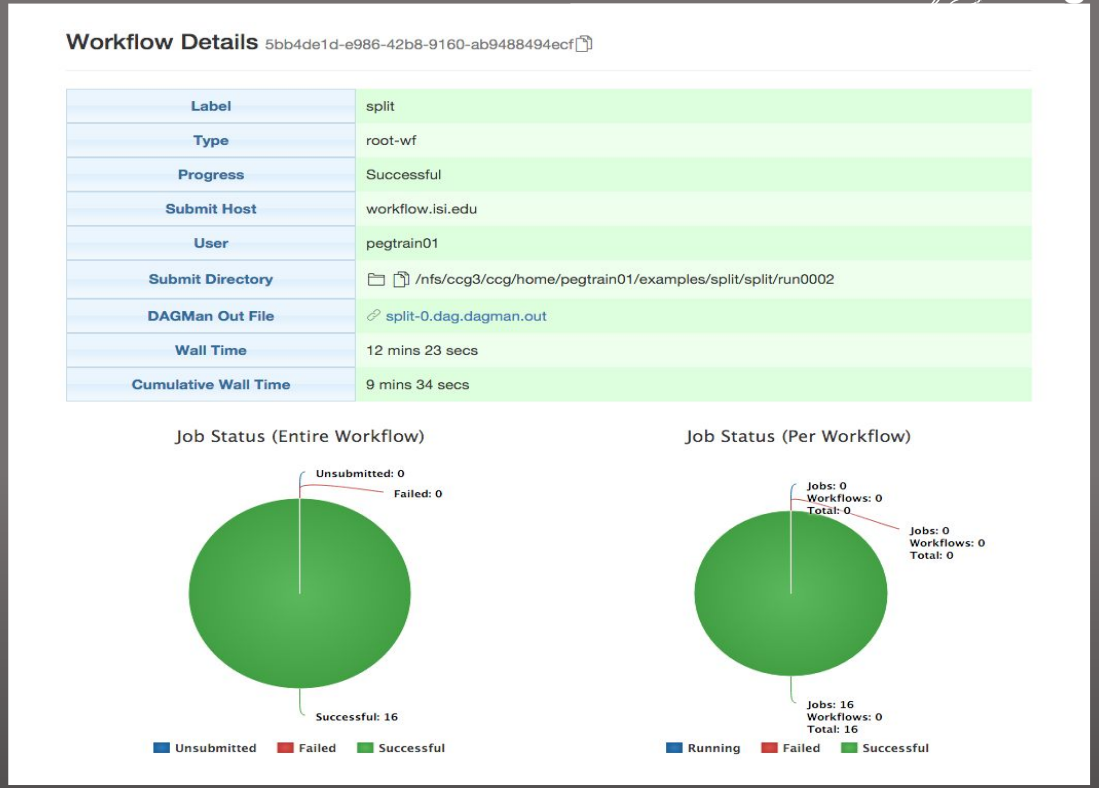
Troubleshooting

RESTful API

# Pegasus dashboard

web interface for monitoring and debugging workflows

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.




https://pegasus.isi.edu