NSF U.S. DEPARTMENT OF ENERGY NIH

Pegasus

Automate, recover, and debug scientific computations.

Rafael Ferreira da Silva

USC Viterbi
School of Engineering
Information Sciences Institute

http://pegasus.isi.edu

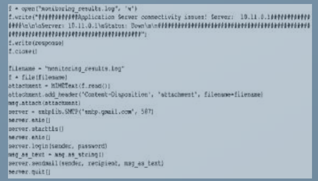
Experiment Timeline

Scientific Problem

Earth Science, Astronomy,
Neuroinformatics,
Bioinformatics, etc.

$$U\frac{\mathrm{d}^2}{\mathrm{d}x^2}U^{-1}g = \left(U\frac{\mathrm{d}}{\mathrm{d}x}U^{-1}\right)\times\left(U\frac{\mathrm{d}}{\mathrm{d}x}U^{-1}\right)g$$
$$= \frac{\mathrm{d}}{\mathrm{d}\psi}\left[g'\psi' + \frac{1}{2}g\frac{\psi''}{\psi'}\right]\cdot\psi' + \frac{1}{2}\left[g'\psi' + \frac{1}{2}g\frac{\psi''}{\psi'}\right]\times\frac{\psi''}{\psi'}$$
$$= g''\psi'^2 + 2g'\psi'' + \frac{1}{2}g\times\left[\frac{\psi'''}{\psi'} + \frac{\psi''^2}{\psi'^2}\right]$$

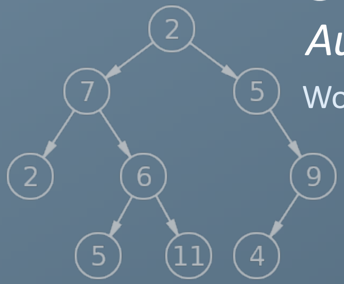
Analytical Solution



```
f = open("monitoring_results.log", 'w')
f.write("############Application Server connectivity issues! Server:  10.11.0.1###########
####\n\n\nServer: 10.11.0.1\nStatus: Down\n\n##################################################
#######################################");
f.write(response)
f.close()

filename = "monitoring_results.log"
f = file(filename)
attachment = MIMEText(f.read())
attachment.add_header('Content-Disposition', 'attachment', filename=filename)
msg.attach(attachment)
server = smtplib.SMTP('smtp.gmail.com', 587)
server.ehlo()
server.starttls()
server.ehlo()
server.login(sender, password)
msg_as_text = msg.as_string()
server.sendmail(sender, recipient, msg_as_text)
server.quit()
```

Computational Scripts

Shell scripts, Python, Matlab, etc.

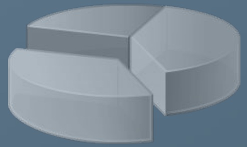


Automation

Workflows, MapReduce, etc.

Distributed Computing

Clusters, HPC,
Cloud, Grid, etc.

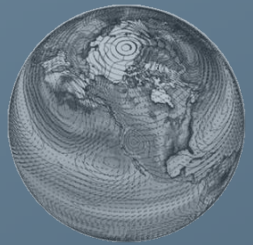


Monitoring and Debug

Fault-tolerance, Provenance, etc.

Scientific Result

Models, Quality Control,
Image Analysis, etc.



Pegasus

http://pegasus.isi.edu

What is involved in an experiment execution?

Energy
Heterogeneity
Abstraction
Provenance
Workflow
Linux
Scheduling
Performance
HPC
Overhead
Monitoring
I/O
Modeling
Checkpointing
Jobs
Grid
Data
Error
Optimization
Clustering
Exception
Re-submission
Reproducibility
Cluster
Reliability
Tasks
Dependency
Throughput
Automation
Fault-tolerance
Provisioning
Cost
MPI
Analysis
Wrappers
Middleware
Simulation
Scalability
Portability
Deadline
Constraints
Cloud
Pipeline
File
Scripts
DAG
Robustness
Estimation

Why Pegasus?

Automates complex, multi-stage processing pipelines

Enables parallel, distributed computations

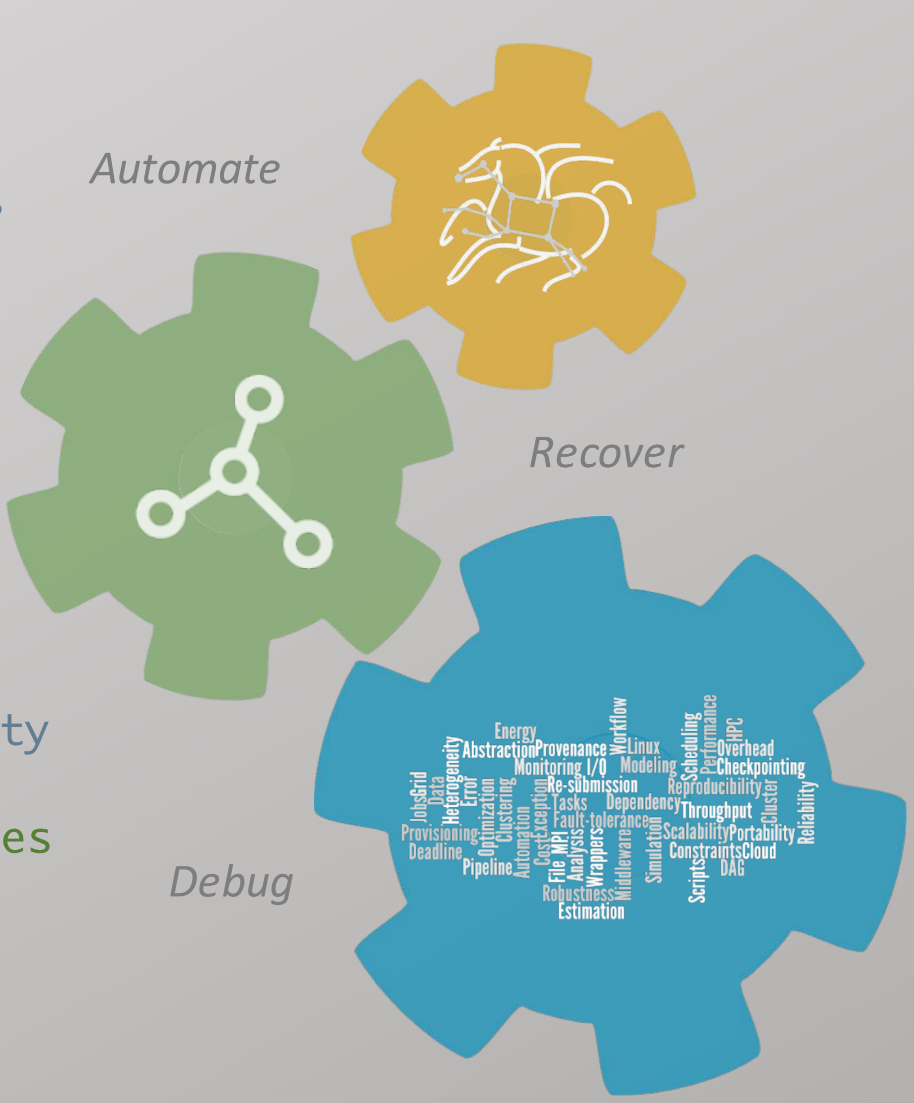
Automatically executes data transfers

Reusable, aids reproducibility

Records how data was produced (provenance)

Handles failures with to provide reliability

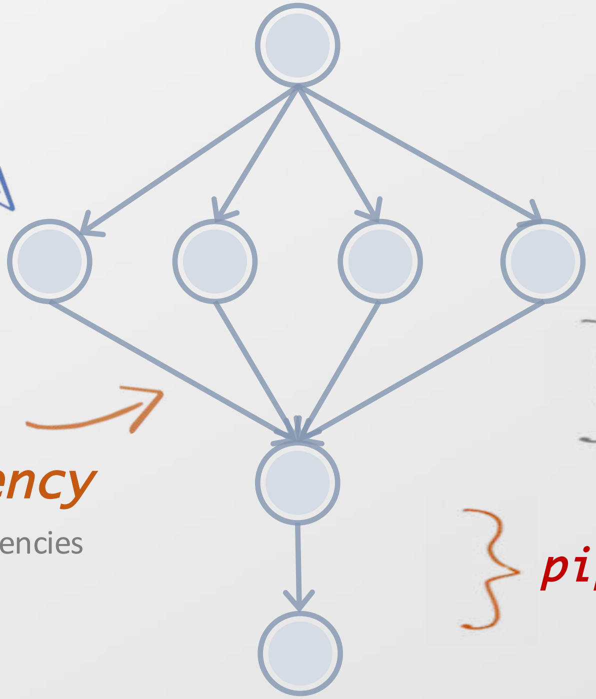
Keeps track of data and files



Taking a closer look into a workflow…

DAG

directed-acyclic graphs



job

Command-line programs

dependency

Usually data dependencies

split

merge

pipeline

DAX

DAG in XML

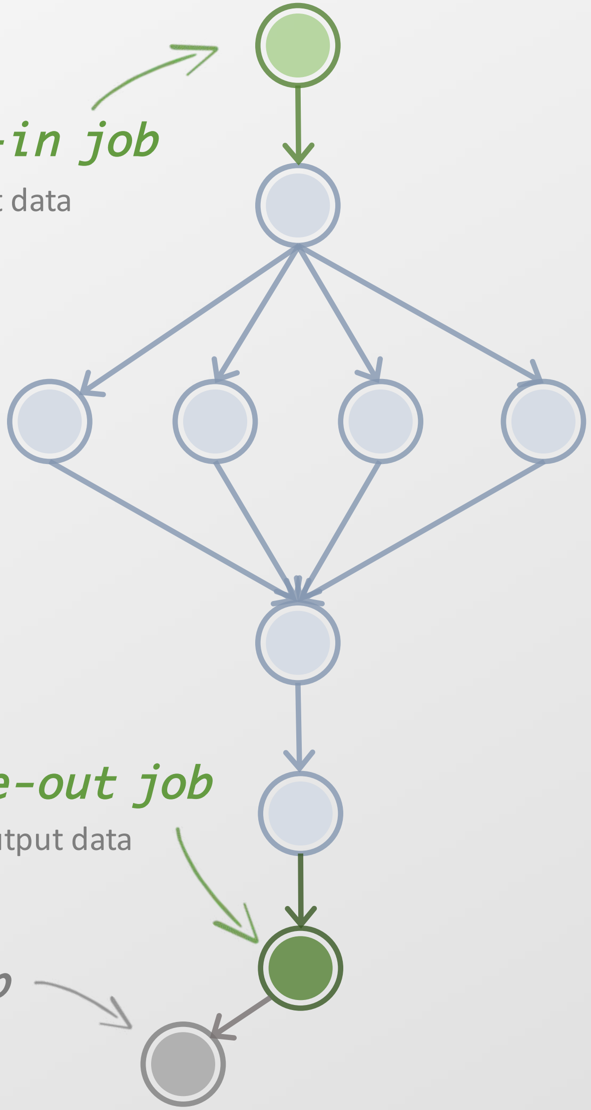
From the abstraction to execution!

abstract workflow

executable workflow

optimizations

storage constraints



stage-in job

Transfers the workflow input data

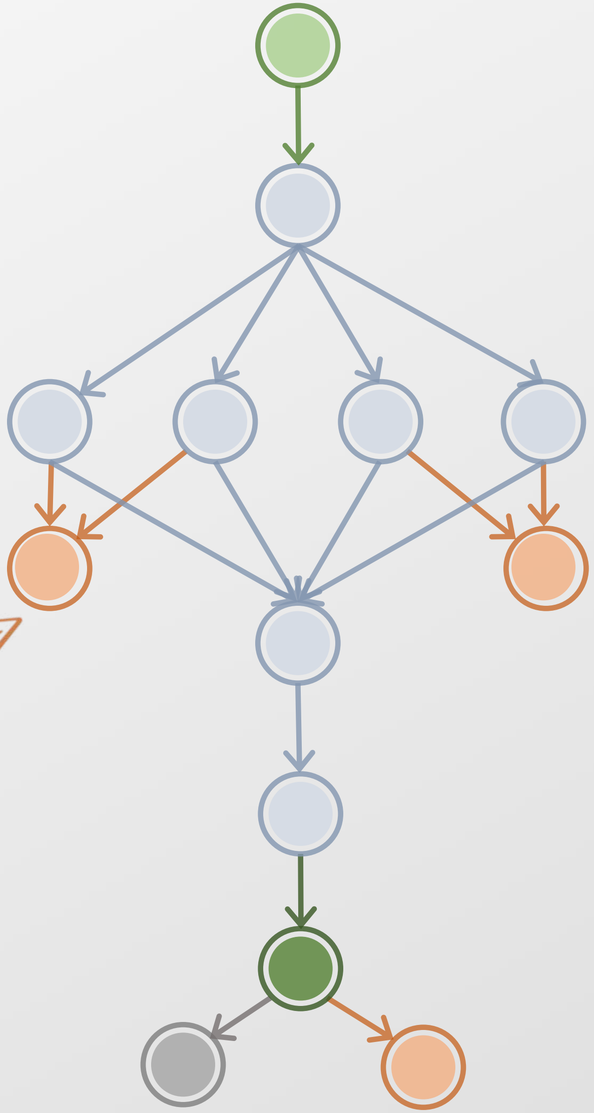
stage-out job

Transfers the workflow output data

registration job

Registers the workflow output data

Optimizing storage usage…

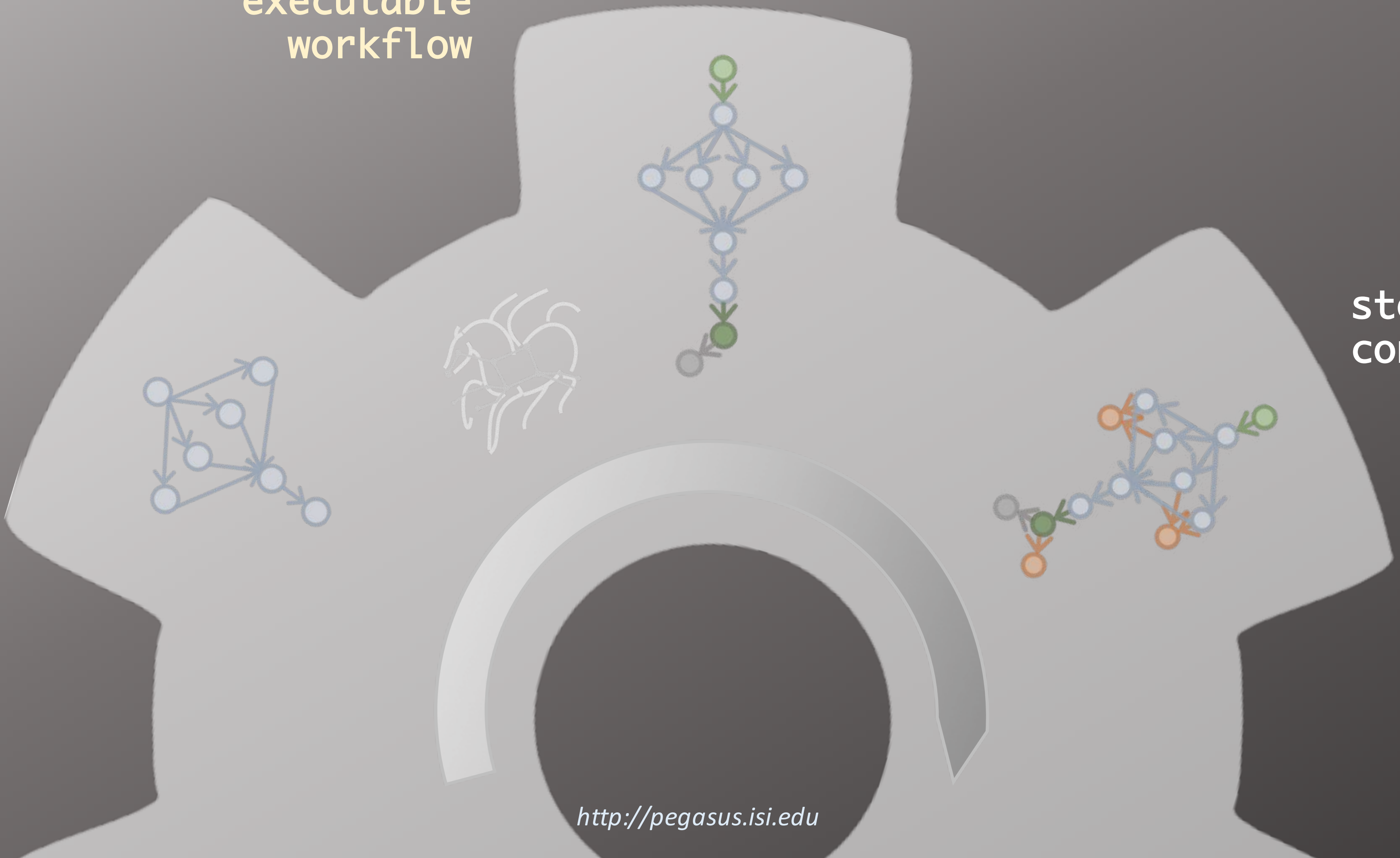


cleanup job

Removes unused data

In a nutshell…

…and all automatically!

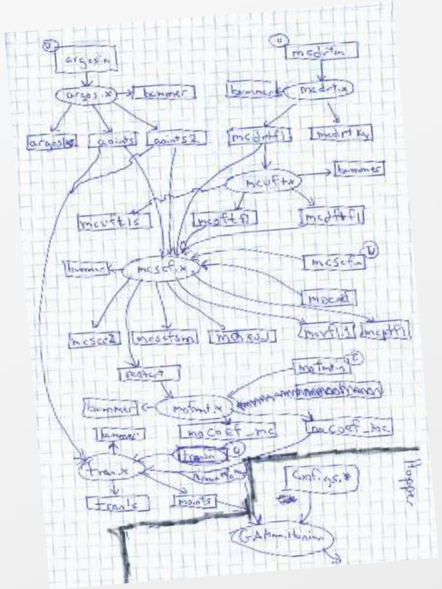


executable workflow

abstract workflow

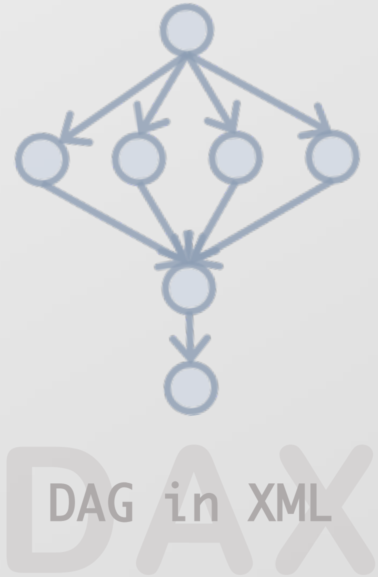
storage constraints

Pegasus also provides tools to generate the abstract workflow



python Java perl

```
dax = ADAG("test_dax")
firstJob = Job(name="first_job")
firstInputFile = File("input.txt")
firstOutputFile = File("tmp.txt")
firstJob.addArgument("input=input.txt", "output=tmp.txt")
firstJob.uses(firstInputFile, link=Link.INPUT)
firstJob.uses(firstOutputFile, link=Link.OUTPUT)
dax.addJob(firstJob)
for i in range(0, 5):
    simulJob = Job(id="%s" % (i+1), name="simul_job”)
    simulInputFile = File("tmp.txt”)
    simulOutputFile = File("output.%d.dat" % i)
    simulJob.addArgument("parameter=%d" % i, "input=tmp.txt”,
        output=%s" % simulOutputFile.getName())
    simulJob.uses(simulInputFile, link=Link.INPUT)
    simulJob.uses(simulOutputFile, line=Link.OUTPUT)
dax.addJob(simulJob)
dax.depends(parent=firstJob, child=simulJob)
fp = open("test.dax", "w”)
dax.writeXML(fp)
fp.close()
```

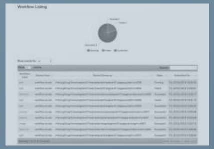


While you wait…

…or the execution is finished.

Does everything executed successfully?

How my workflow behaves?



Web-based interface

Real-time monitoring, graphs, provenance, etc.

Statistics

Workflow execution and job performance metrics

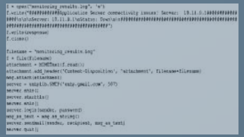
Debug

Set of debugging tools to unveil issues

RESTful API

Monitoring and reporting information on your own application interface

Past executions?



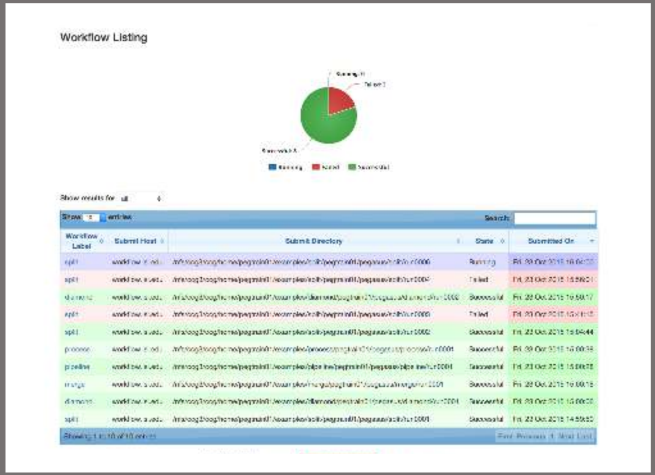
```
f = open("monitoring_results.log", 'w')
f.write("##########Application Server connectivity issues: Server: 10.11.0.1##########
####\n\n\nServer: 10.11.0.1\nStatus: Down\n\n###############################################
####################################")
f.write(response)
f.close()

filename = "monitoring_results.log"
f = file(filename)
attachment = MIMEText(f.read())
attachment.add_header('Content-Disposition', 'attachment', filename=filename)
msg.attach(attachment)
server = smtplib.SMTP('smtp.gmail.com', 587)
server.ehlo()
server.starttls()
server.ehlo()
server.login(sender, password)
msg_as_text = msg.as_string()
server.sendmail(sender, recipient, msg_as_text)
server.quit()
```

Command-line tools

Tools for monitor and debug workflows

Pegasus

http://pegasus.isi.edu



Workflow Listing

Show results for all

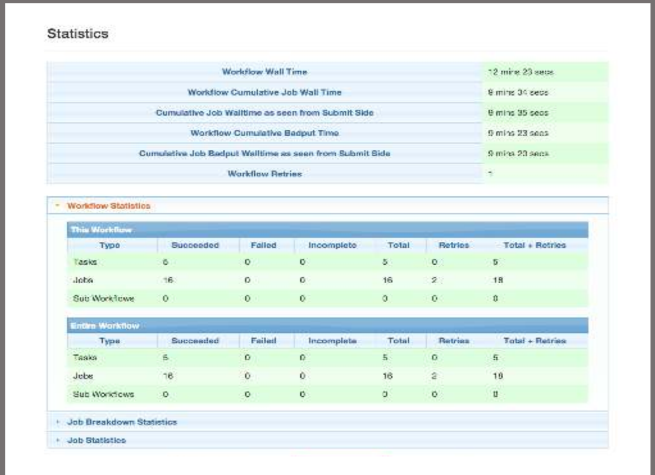
Show 10 entries Search:

Workflow Label	Submit Host	Submit Directory	State	Submitted On
split	workflow.isi.edu	/nfs/ccg3/ccg/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0006	Running	Fri, 23 Oct 2015 16:04:22
split	workflow.isi.edu	/nfs/ccg3/ccg/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0004	Failed	Fri, 23 Oct 2015 15:59:01
diamond	workflow.isi.edu	/nfs/ccg3/ccg/home/pegtrain01/examples/diamond/pegtrain01/pegasus/diamond/run0002	Successful	Fri, 23 Oct 2015 15:50:17
split	workflow.isi.edu	/nfs/ccg3/ccg/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0003	Failed	Fri, 23 Oct 2015 15:41:15
split	workflow.isi.edu	/nfs/ccg3/ccg/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0002	Successful	Fri, 23 Oct 2015 15:04:44
process	workflow.isi.edu	/nfs/ccg3/ccg/home/pegtrain01/examples/process/pegtrain01/pegasus/process/run0001	Successful	Fri, 23 Oct 2015 15:00:38
pipeline	workflow.isi.edu	/nfs/ccg3/ccg/home/pegtrain01/examples/pipeline/pegtrain01/pegasus/pipeline/run0001	Successful	Fri, 23 Oct 2015 15:00:28
merge	workflow.isi.edu	/nfs/ccg3/ccg/home/pegtrain01/examples/merge/pegtrain01/pegasus/merge/run0001	Successful	Fri, 23 Oct 2015 15:00:15
diamond	workflow.isi.edu	/nfs/ccg3/ccg/home/pegtrain01/examples/diamond/pegtrain01/pegasus/diamond/run0001	Successful	Fri, 23 Oct 2015 15:00:00
split	workflow.isi.edu	/nfs/ccg3/ccg/home/pegtrain01/examples/split/pegtrain01/pegasus/split/run0001	Successful	Fri, 23 Oct 2015 14:59:55

Showing 1 to 10 of 10 entries

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 23 secs
Workflow Retries	1

Workflow Statistics

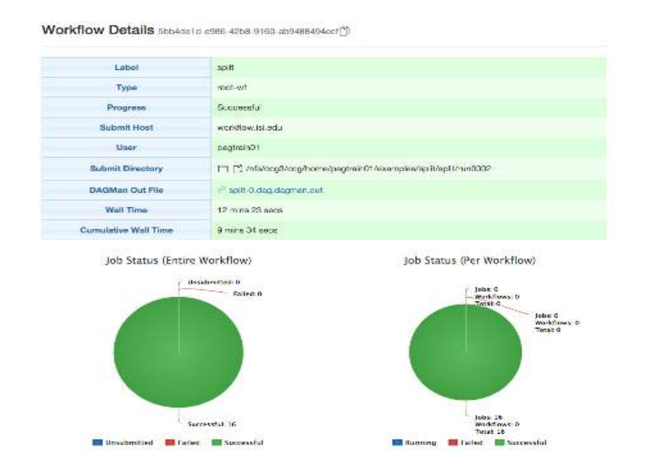
This Workflow

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

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-c986-42b8-9160-ab9488494ccf

Label	split
Type	root-wf
Progress	Successful
Submit Host	workflow.isi.edu
User	pegtrain01
Submit Directory	/nfs/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

Real-time Monitoring

Reporting

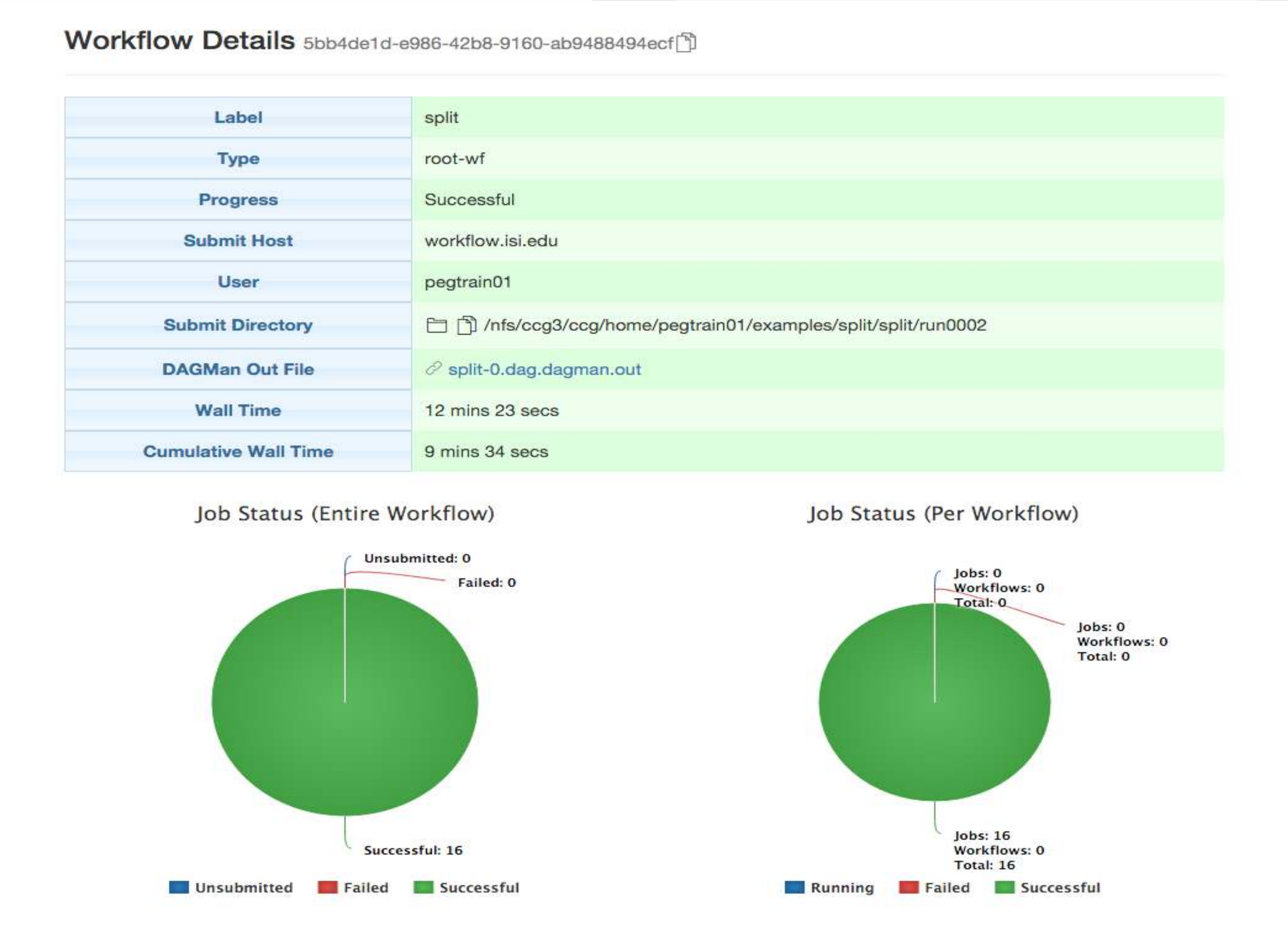
Debugging

Troubleshooting

RESTful API

Pegasus dashboard

web interface for monitoring and debugging workflows

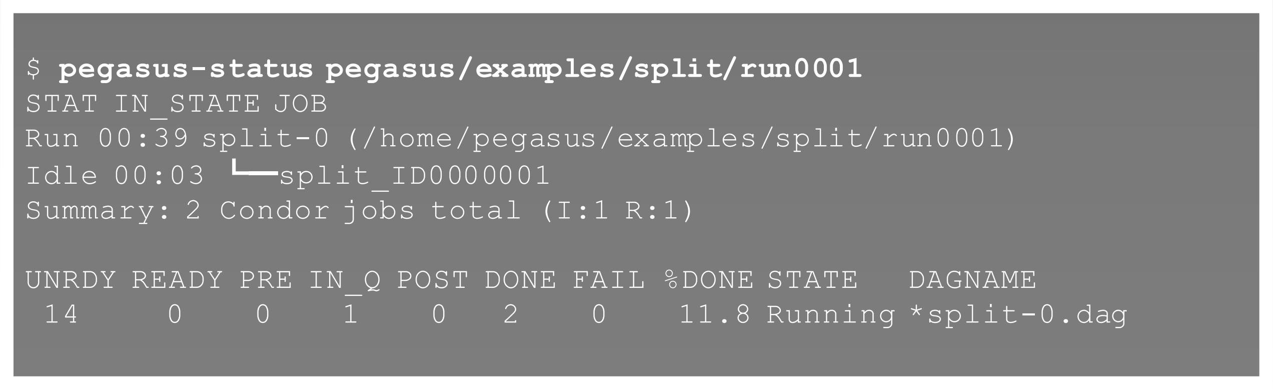


Workflow Details 5bb4de1d-e986-42b8-9160-ab9488494ecf

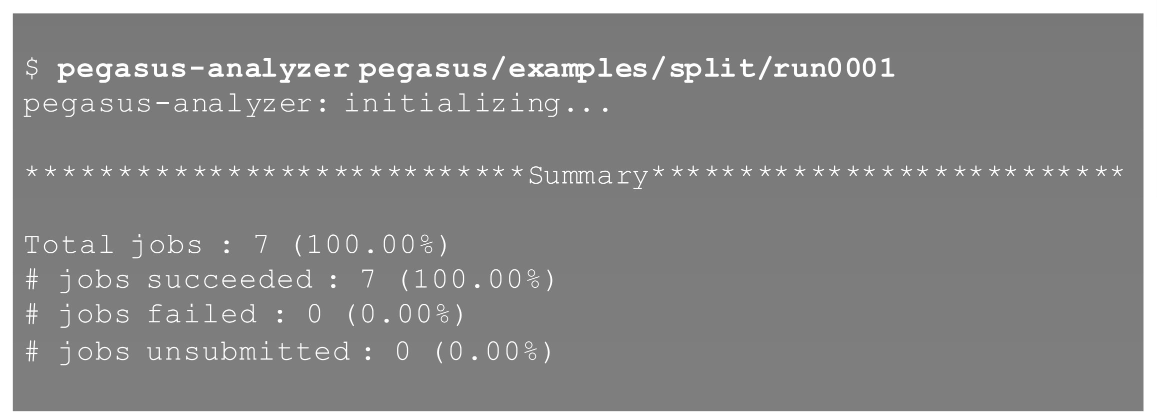
Label	split
Type	root-wf
Progress	Successful
Submit Host	workflow.isi.edu
User	pegtrain01
Submit Directory	/nfs/ccg3/ccg/home/pegtrain01/examples/split/split/run0002
DAGMan Out File	split-0.dag.dagman.out
Wall Time	12 mins 23 secs
Cumulative Wall Time	9 mins 34 secs

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.

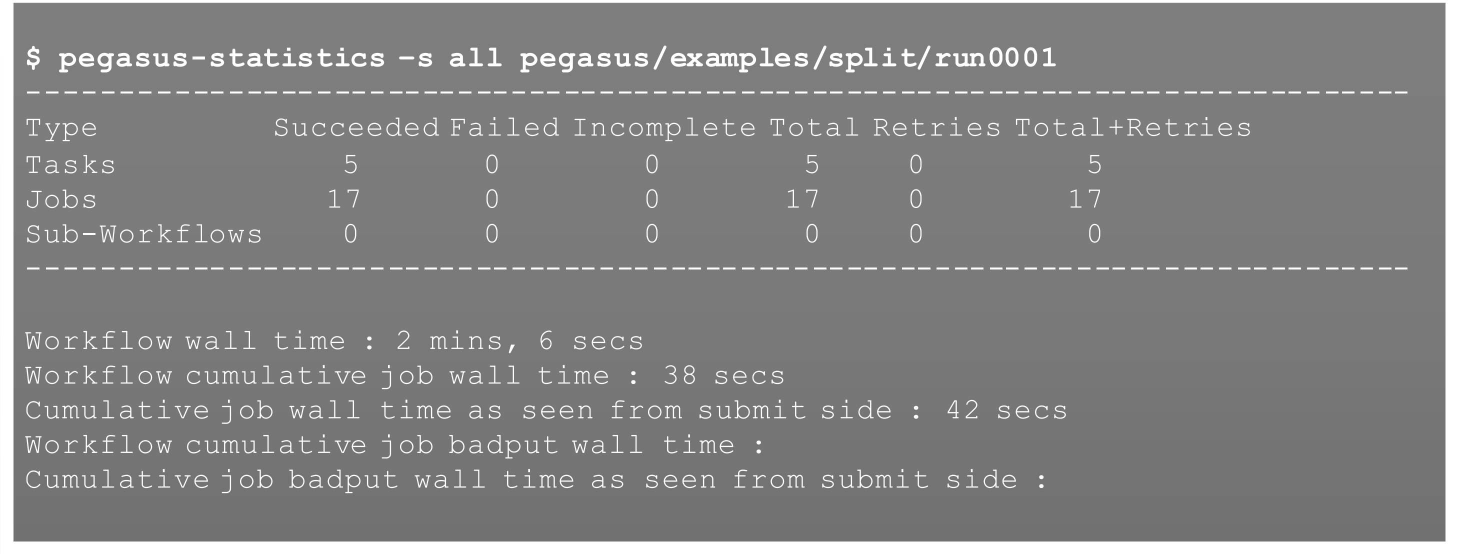
>_ But, if you prefer the command-line…



```
$ 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
```



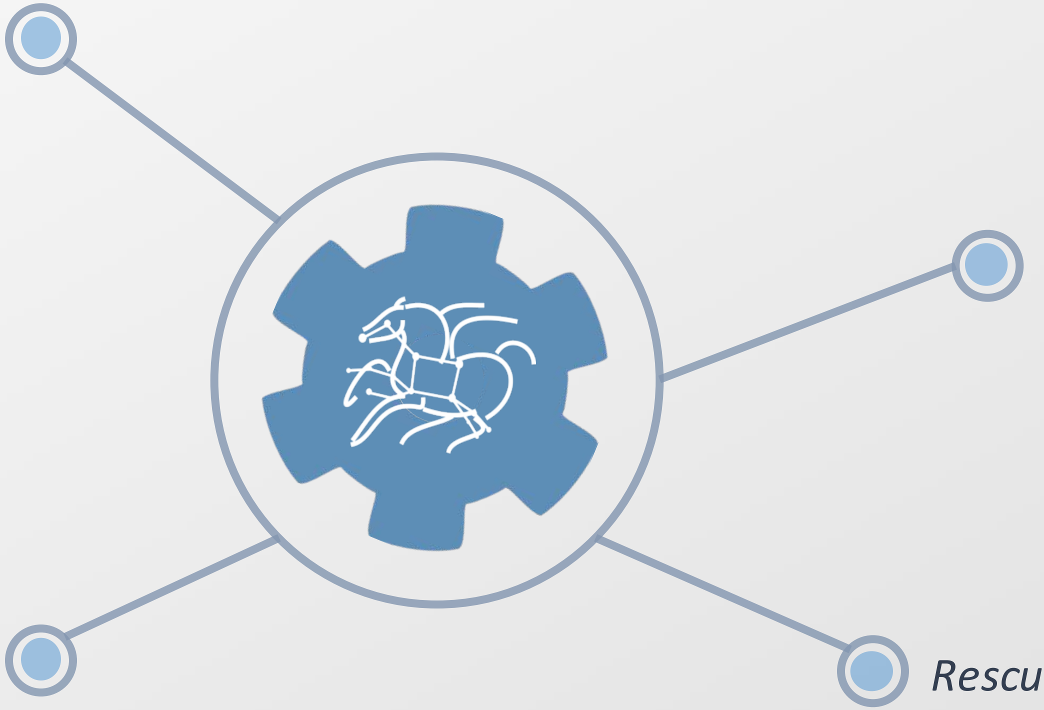
```
$ pegasus-analyzer pegasus/examples/split/run0001
pegasus-analyzer: initializing...

****************************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 :
```

…Pegasus provides a set of concise and powerful tools

And if a job fails?



Job Failure Detection

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

Job Retry

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

Checkpoint Files

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

Rescue DAGs

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

SRM

http

Local disk

Amazon S3

Worried about data?

Let Pegasus manage it for you

GridFTP

Shared filesystem

ftp

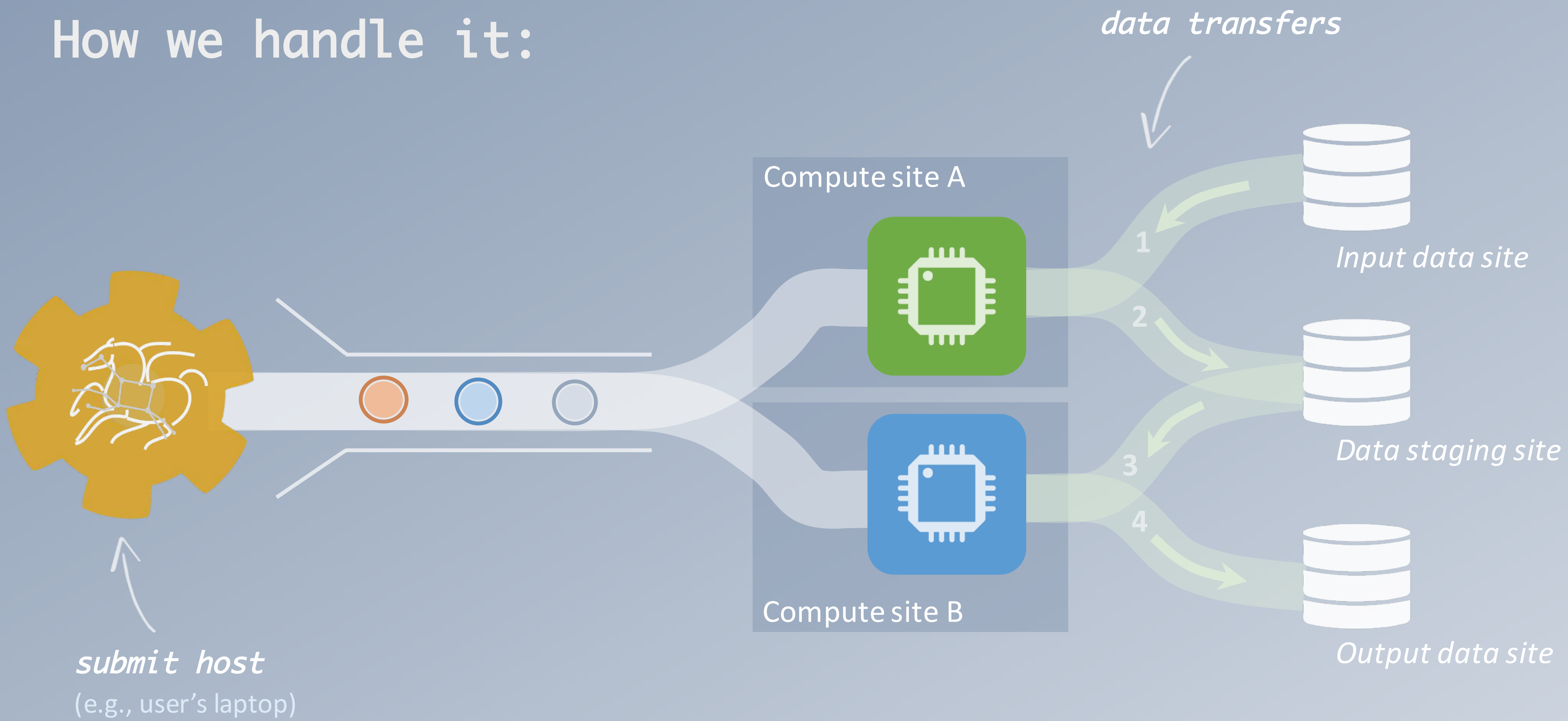
scp

Google Storage

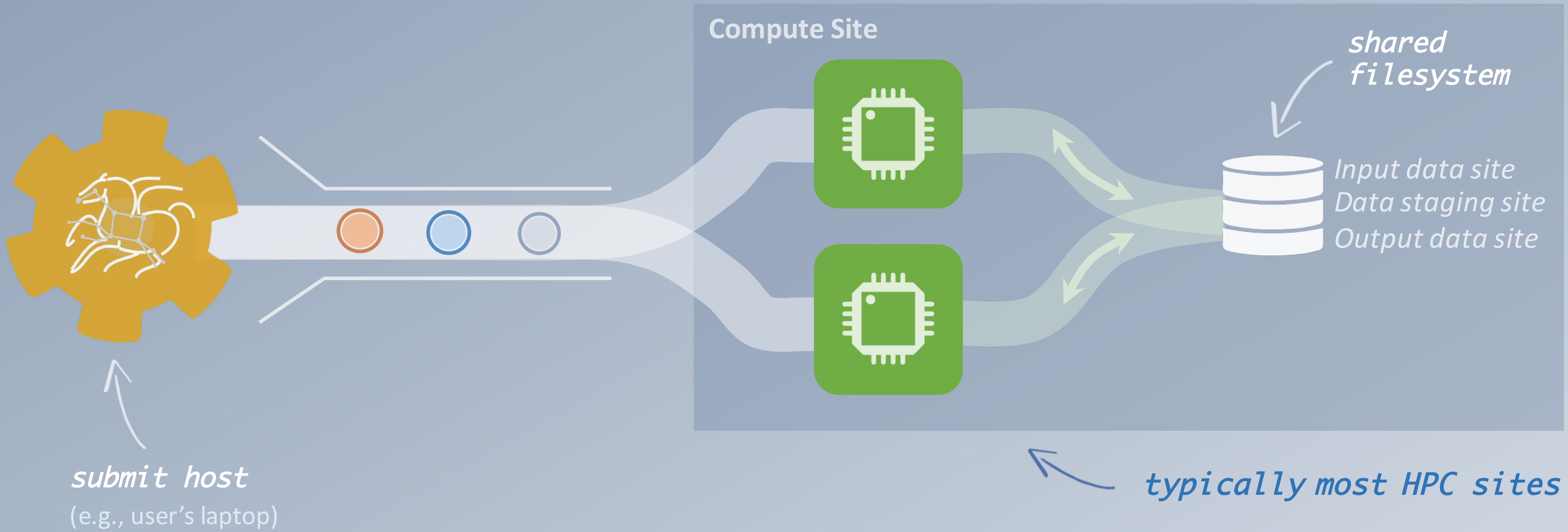
StashCache

iRODS

How we handle it:

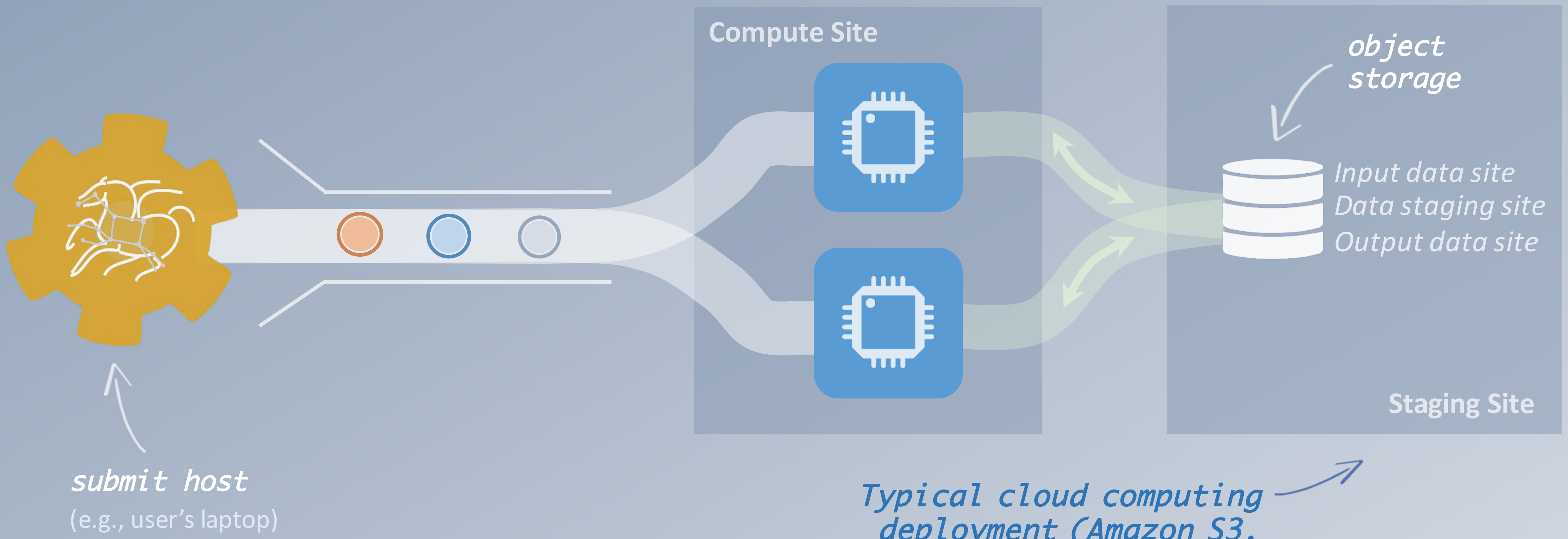


However, there are several possible configurations for data sites…



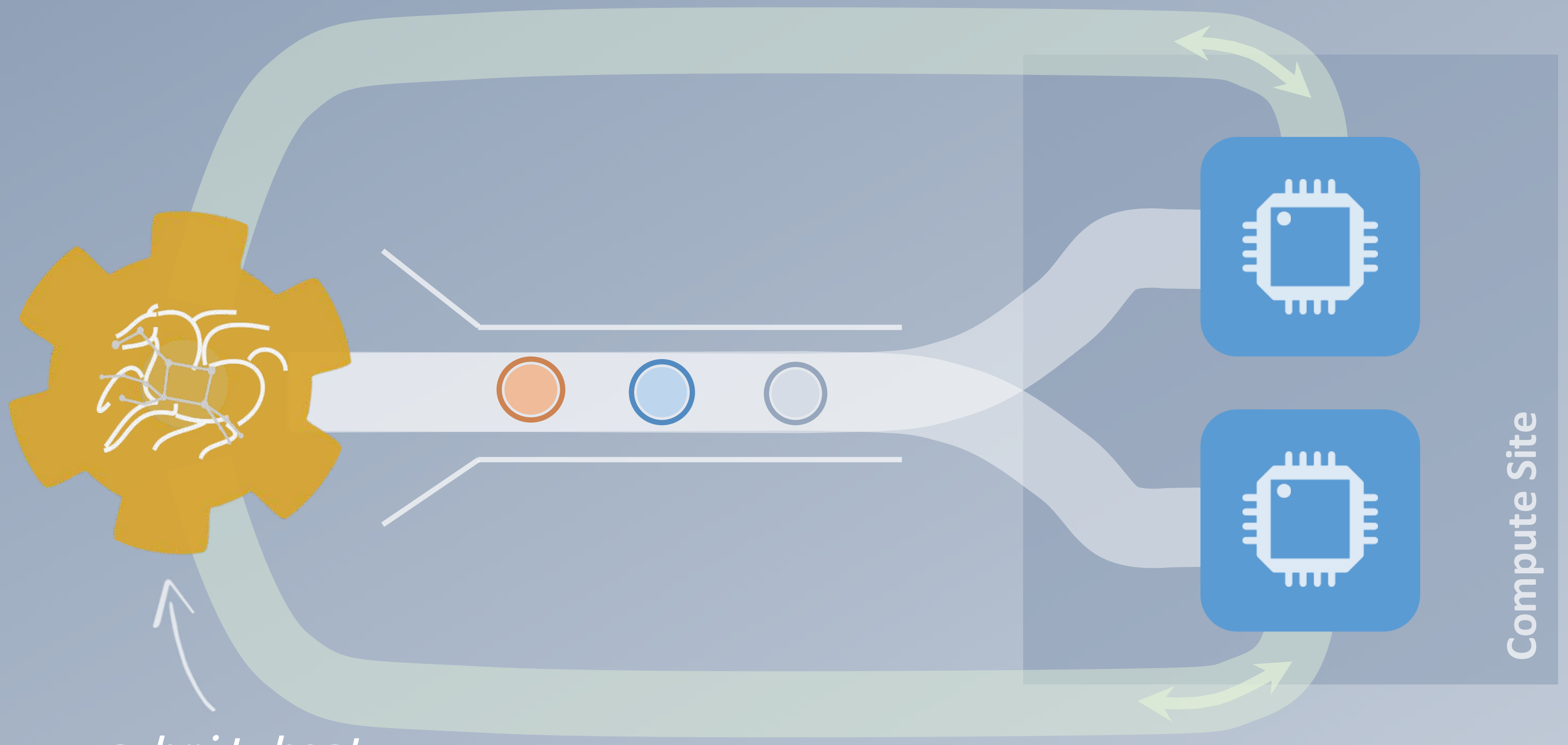
typically most HPC sites

Pegasus also handles high-scalable object storages



Typical cloud computing deployment (Amazon S3, Google Storage)

Pegasus can also manage data over the submit host…



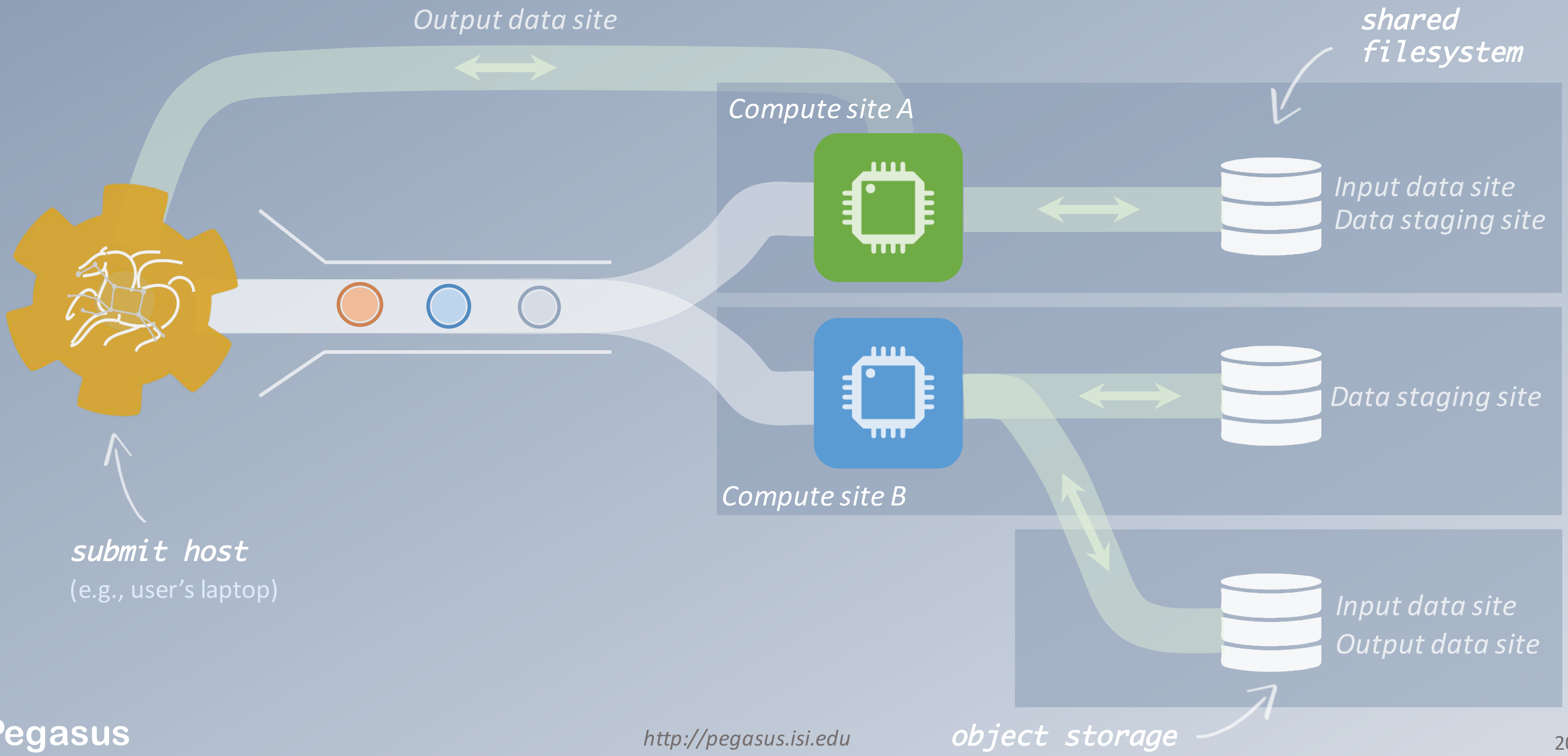
Typical OSG sites

Open Science Grid

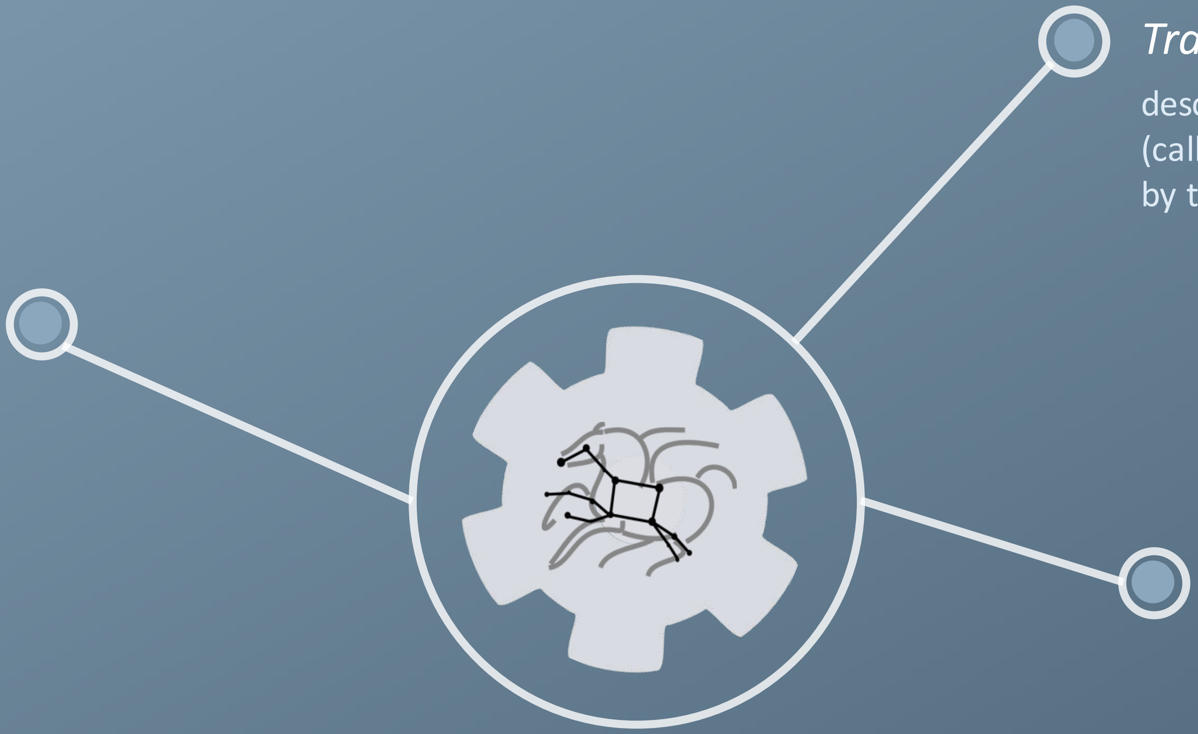
submit host

(e.g., user's laptop)

And yes… you can mix everything!



So, what information does Pegasus need?



Site Catalog

describes the sites where the workflow jobs are to be executed

Transformation Catalog

describes all of the executables (called “transformations”) used by the workflow

Replica Catalog

describes all of the input data stored on external servers

How does Pegasus decide where to execute?

site description
describes the compute resources

scratch
tells where temporary data is stored

storage
tells where output data is stored

profiles
key-pair values associated per job level

```
...
  <!-- The local site contains information about the submit host -->
    <!-- The arch and os keywords are used to match binaries in the transformation
catalog -->
    <site handle="local" arch="x86_64" os="LINUX">

      <!-- These are the paths on the submit host were Pegasus stores data -->
      <!-- Scratch is where temporary files go -->
      <directory type="shared-scratch" path="/home/tutorial/run">
        <file-server operation="all" url="file:///home/tutorial/run"/>
      </directory>

      <!-- Storage is where pegasus stores output files -->
      <directory type="local-storage" path="/home/tutorial/outputs">
        <file-server operation="all" url="file:///home/tutorial/outputs"/>
      </directory>

      <!-- This profile tells Pegasus where to find the user's private key for SCP
transfers -->
      <profile namespace="env" key="SSH_PRIVATE_KEY">/home/tutorial/.ssh/id_rsa</profile>

    </site>
...
```

How does it know where the executables are or which ones to use?

executables description

list of executables locations per site

physical executables

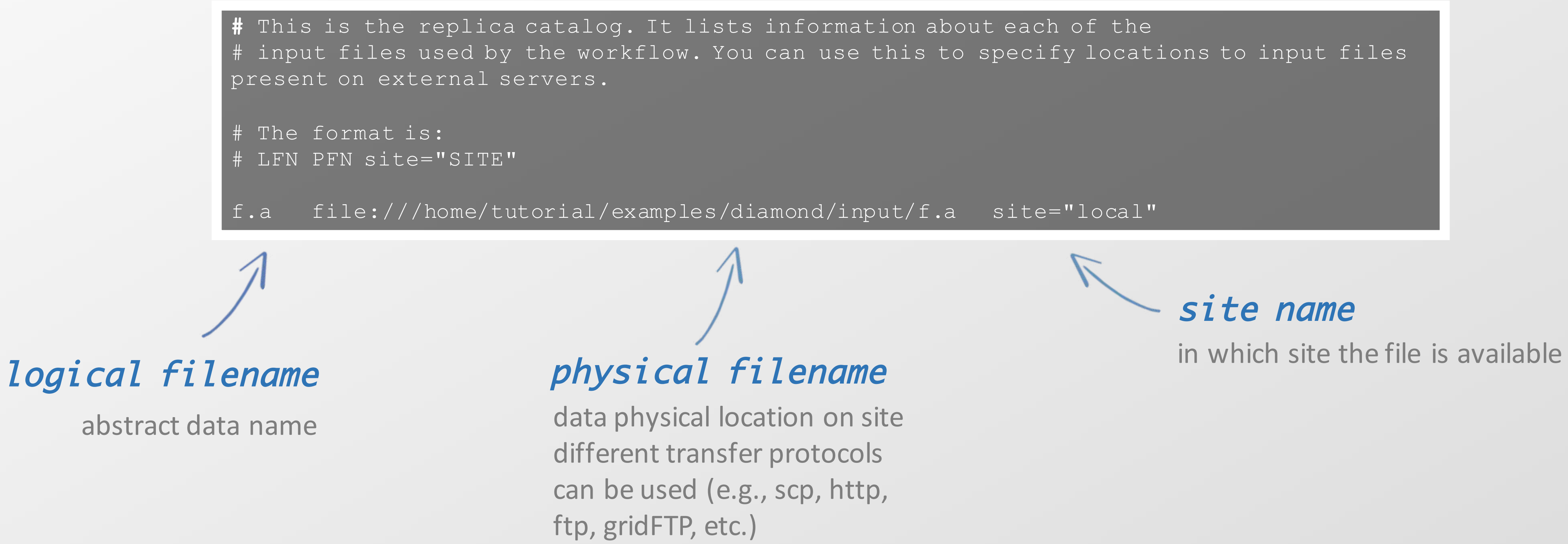
mapped from logical transformations

transformation type

whether it is installed or available to stage

```
...
# This is the transformation catalog. It lists information about each of the
# executables that are used by the workflow.

tr ls {
  site PegasusVM {
    pfn "/bin/ls"
    arch "x86_64"
    os "linux"
    type "INSTALLED”
  }
}
...
```

What if data is not local to the submit host?



```
# This is the replica catalog. It lists information about each of the
# input files used by the workflow. You can use this to specify locations to input files
present on external servers.

# The format is:
# LFN PFN site="SITE"

f.a   file:///home/tutorial/examples/diamond/input/f.a   site="local"
```

logical filename

abstract data name

physical filename

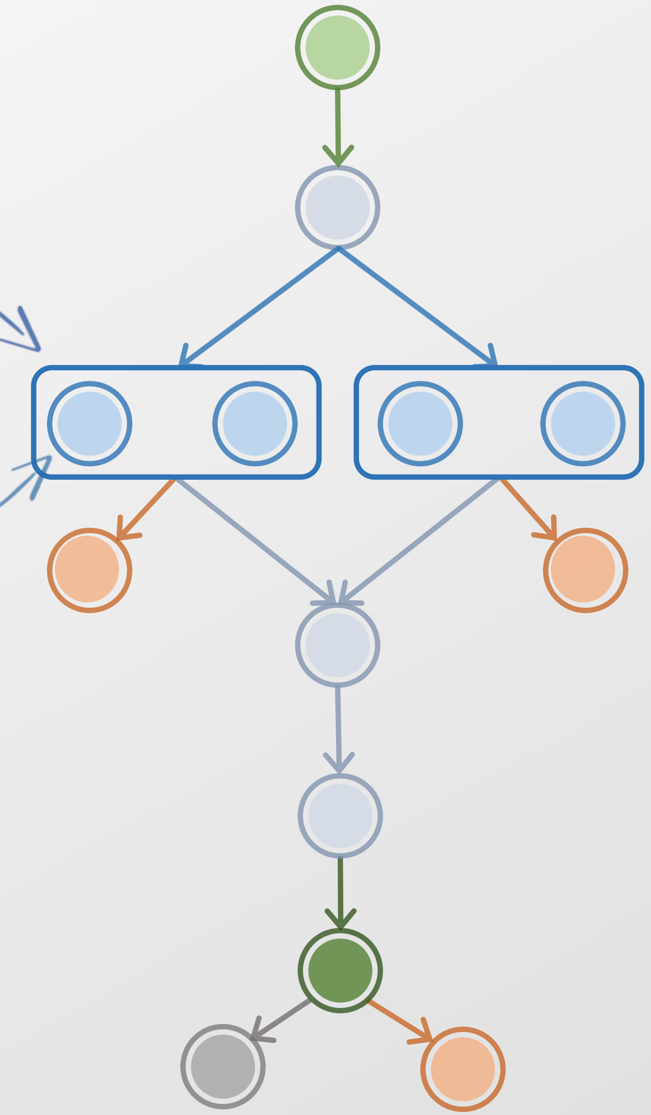
data physical location on site
different transfer protocols
can be used (e.g., scp, http,
ftp, gridFTP, etc.)

site name

in which site the file is available

A few more features…

Performance, why not improve it?



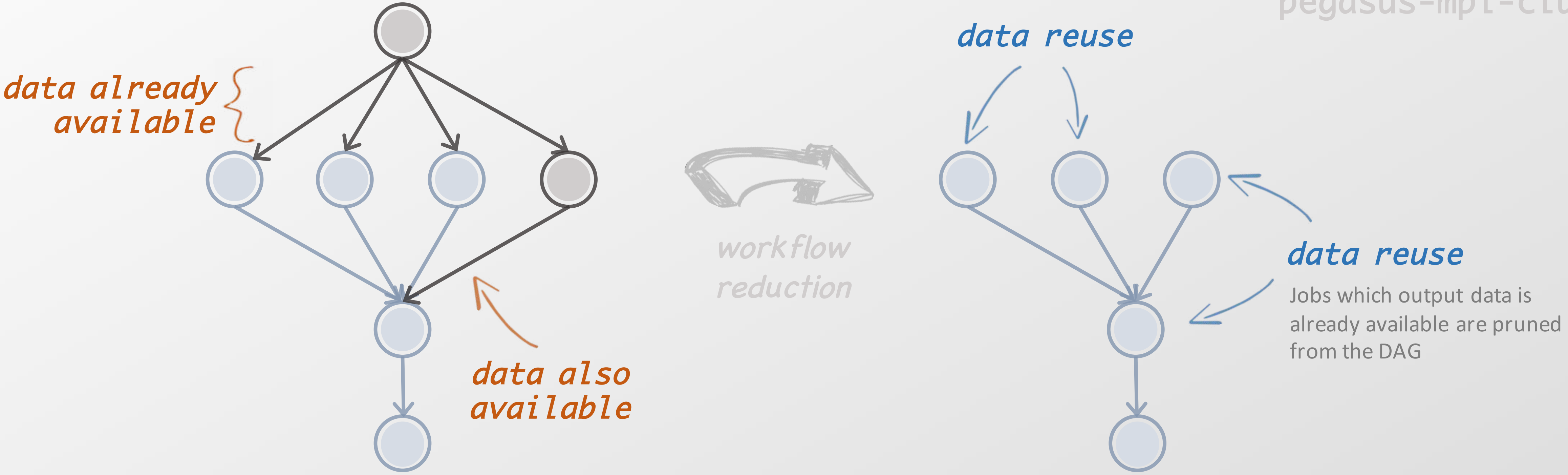
clustered job

Groups small jobs together to improve performance

task

small granularity

What about **data reuse?**



Jobs which output data is already available are pruned from the DAG

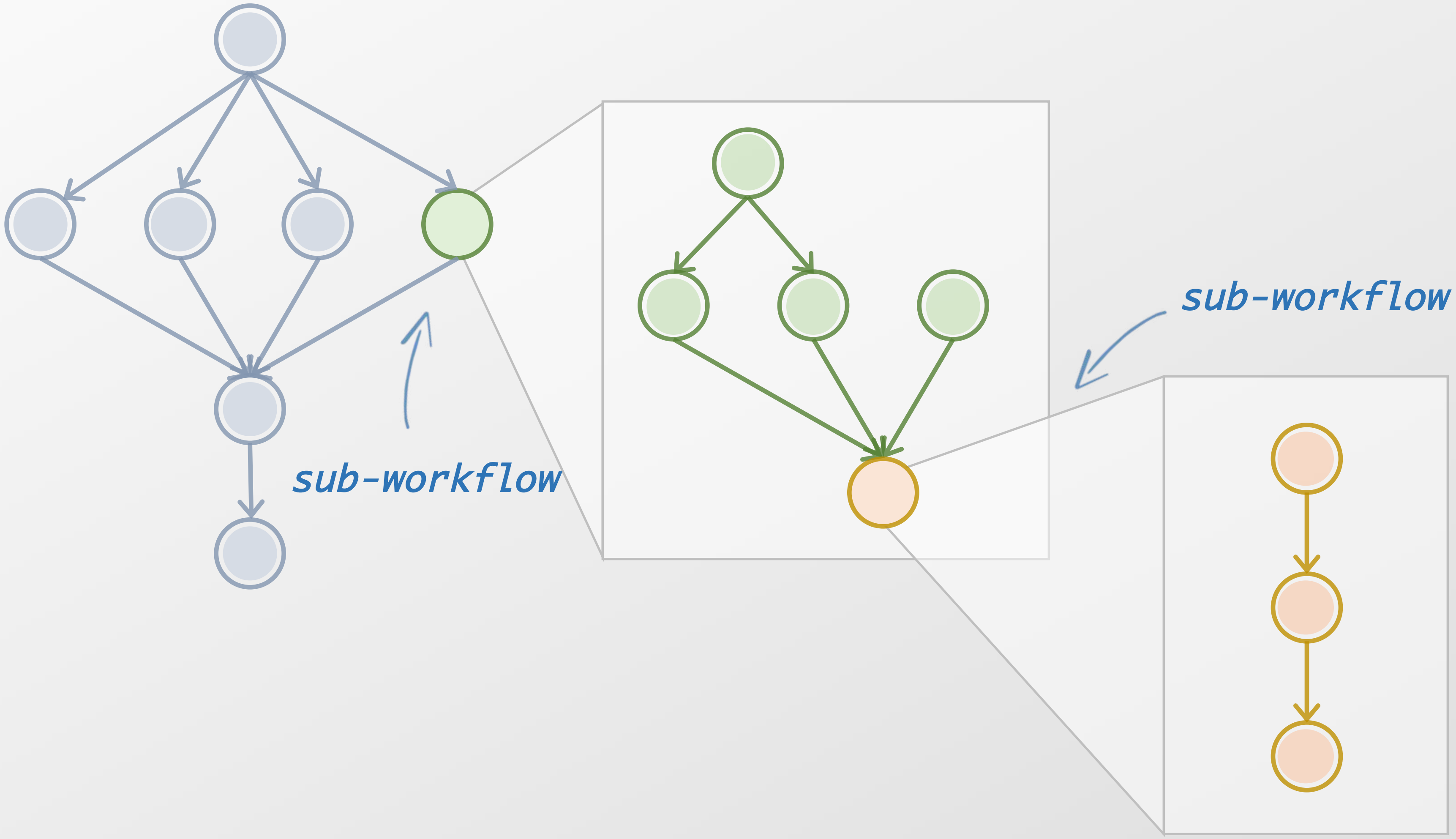
Pegasus also handles **large-scale workflows**

workflow restructuring

workflow reduction

hierarchical workflows

pegasus-mpi-cluster



sub-workflow

sub-workflow

recursion ends when DAX with only compute jobs is encountered

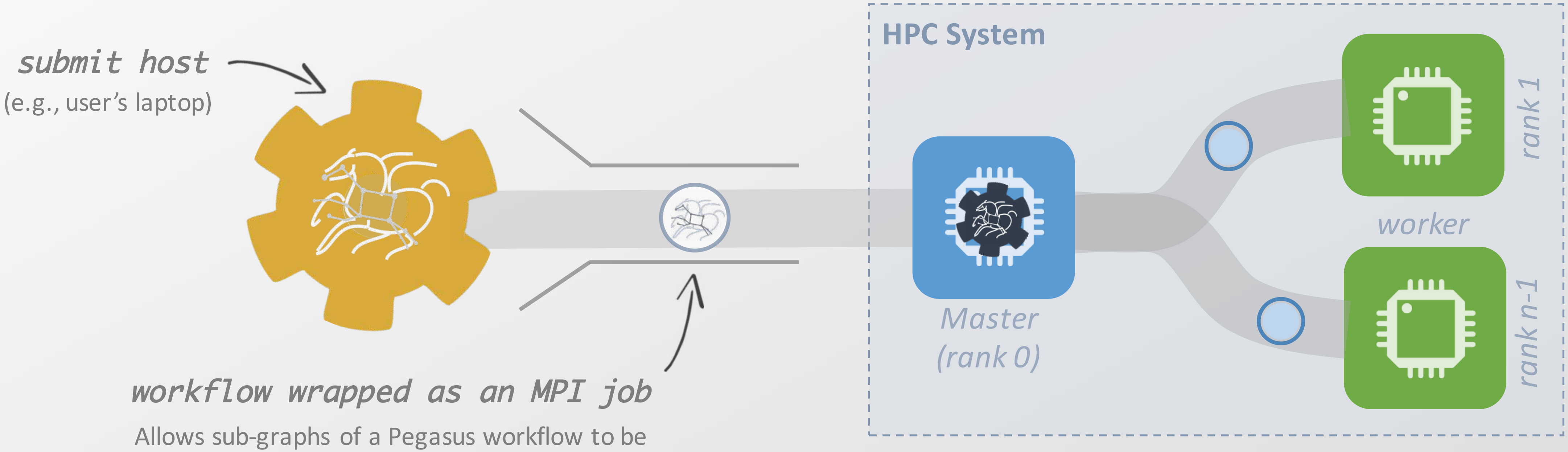
Running **fine-grained** workflows on HPC systems…

workflow restructuring

workflow reduction

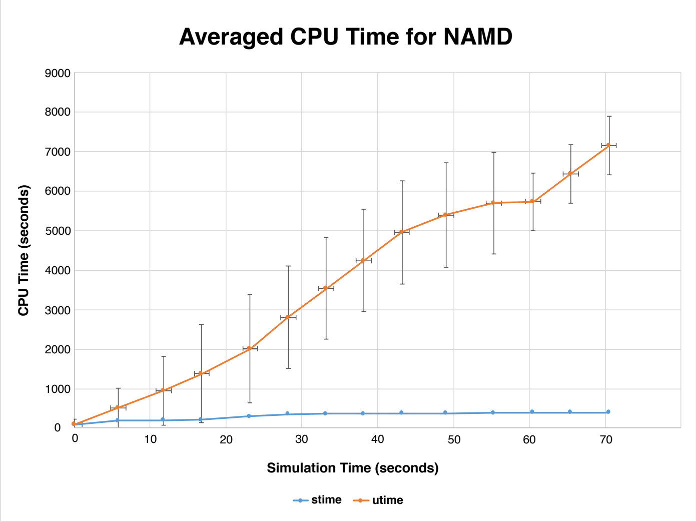
hierarchical workflows

pegasus-mpi-cluster



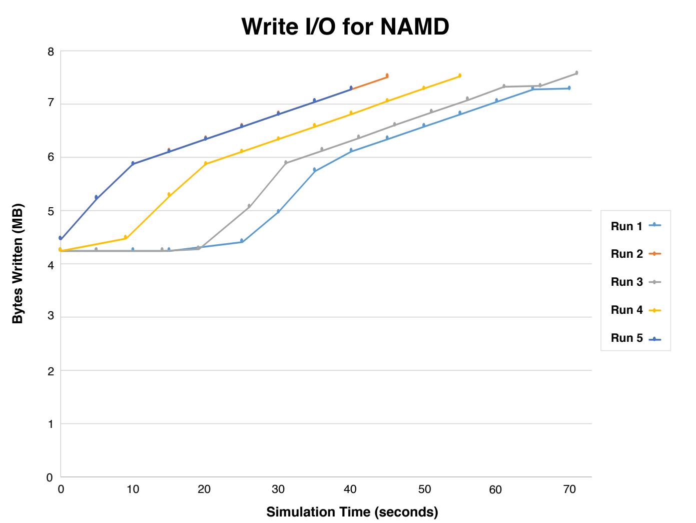
Allows sub-graphs of a Pegasus workflow to be submitted as monolithic jobs to remote resources

Real-time collection of **time-series** of workflow performance metrics



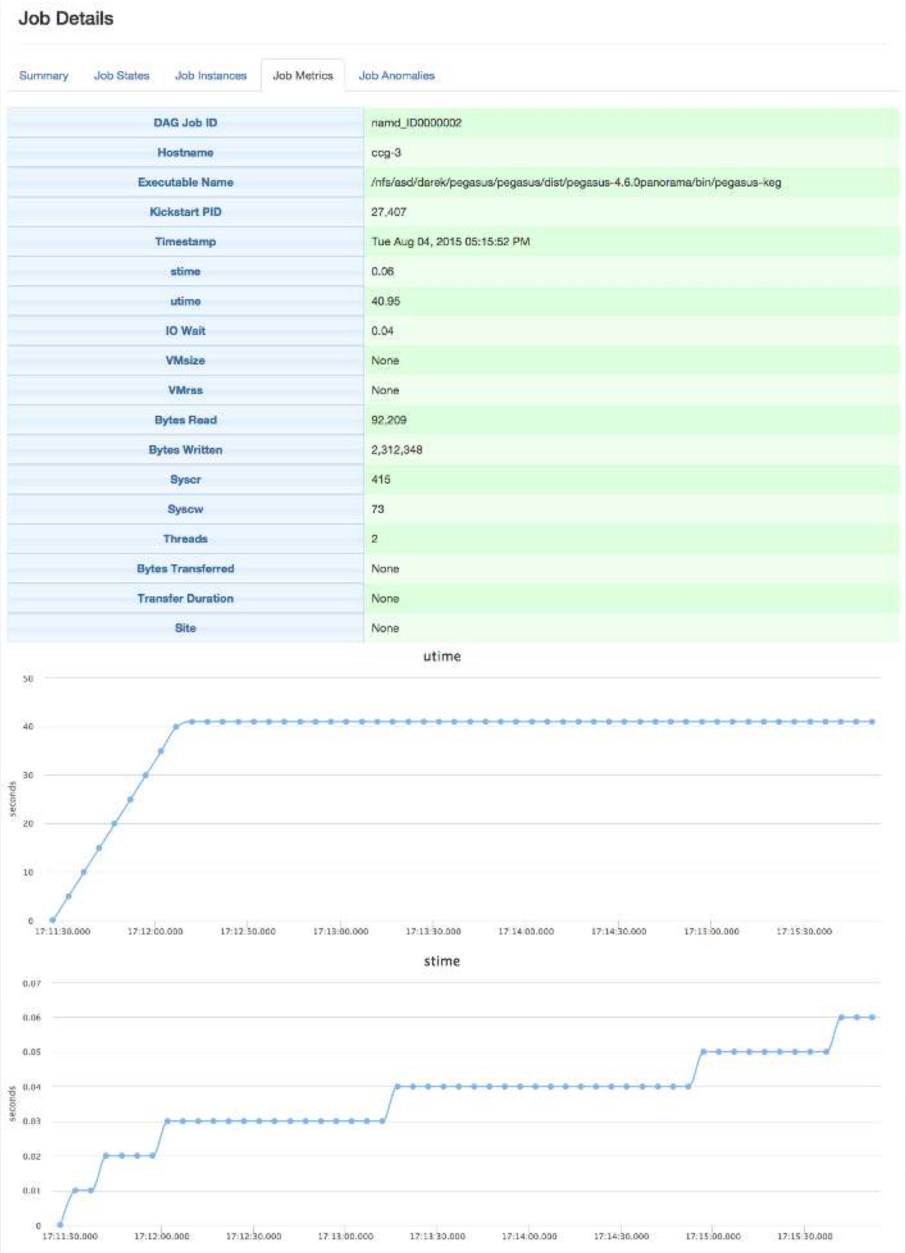
time-series data

I/O (read, write), memory, CPU



time-series data in real-time

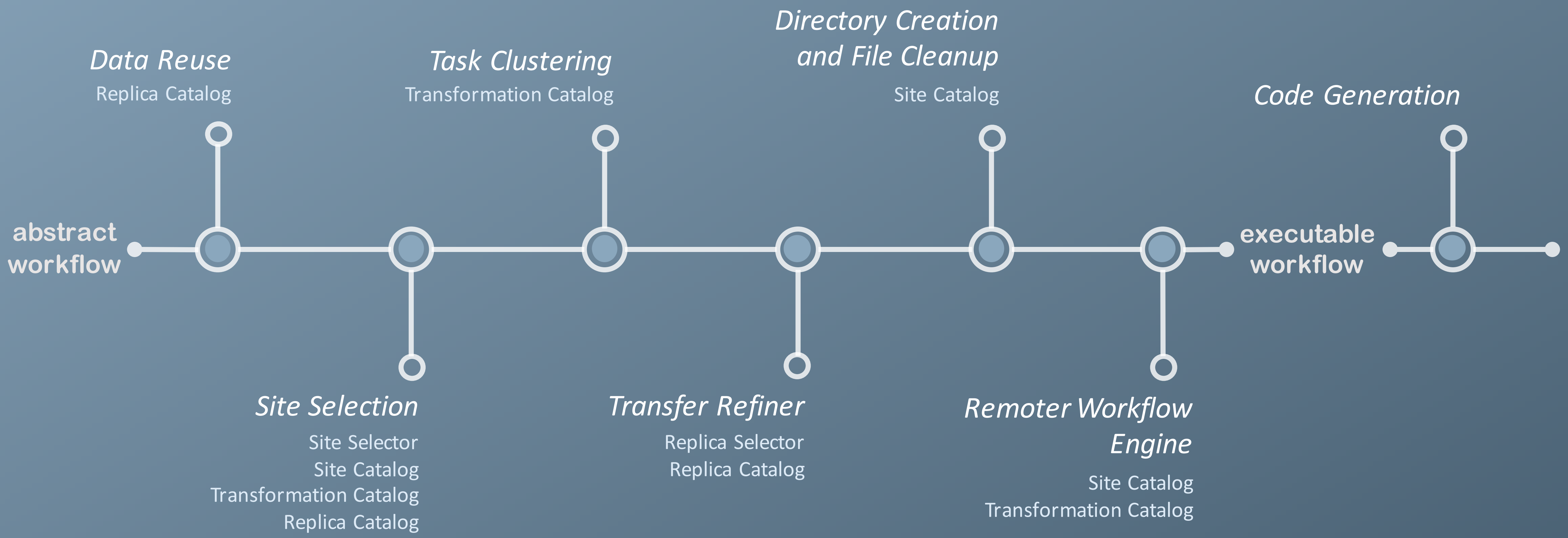
integrated with Pegasus dashboard



Job Details

DAG Job ID	namd_ID0000002
Hostname	ccg-3
Executable Name	/nfs/asd/darek/pegasus/pegasus/dist/pegasus-4.6.0panorama/bin/pegasus-keg
Kickstart PID	27,407
Timestamp	Tue Aug 04, 2015 05:15:52 PM
stime	0.06
utime	40.95
IO Wait	0.04
VMsize	None
VMrss	None
Bytes Read	92,209
Bytes Written	2,312,348
Syscr	415
Syscw	73
Threads	2
Bytes Transferred	None
Transfer Duration	None
Site	None

Pegasus’ flow at a glance



abstract workflow

Data Reuse

Replica Catalog

Site Selection

Site Selector
Site Catalog
Transformation Catalog
Replica Catalog

Task Clustering

Transformation Catalog

Transfer Refiner

Replica Selector
Replica Catalog

Directory Creation and File Cleanup

Site Catalog

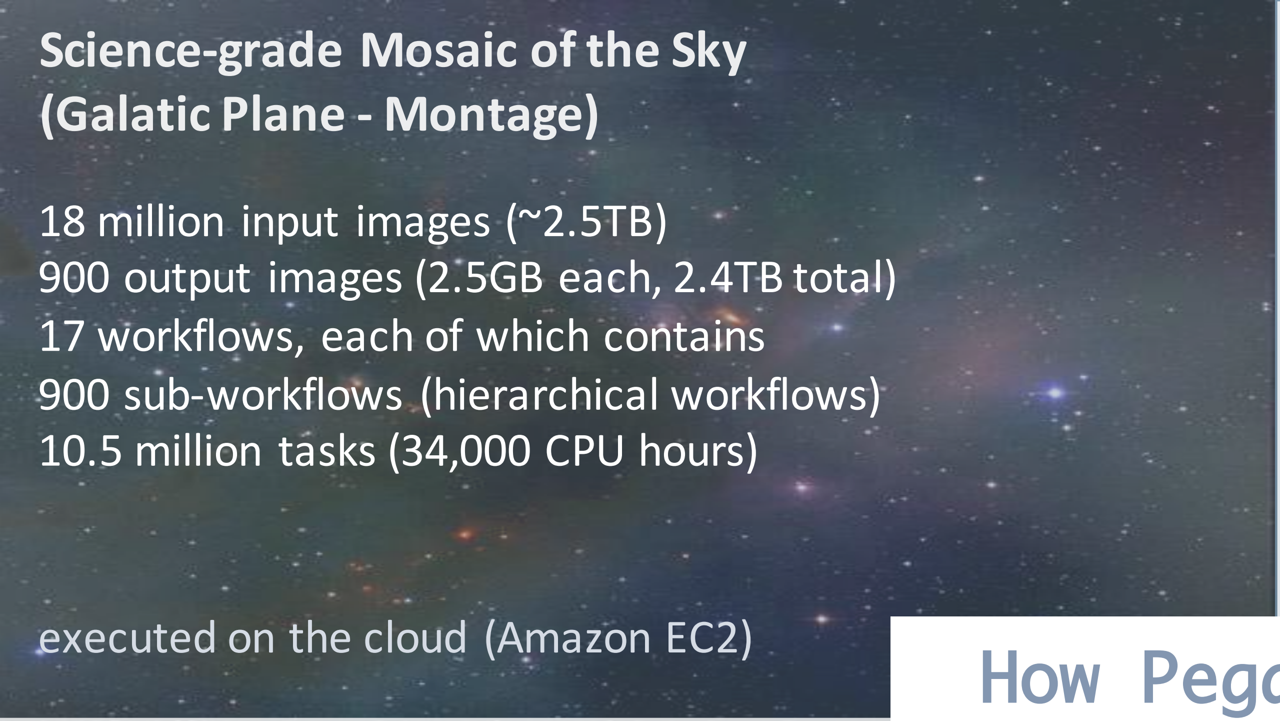
Remoter Workflow Engine

Site Catalog
Transformation Catalog

executable workflow

Code Generation

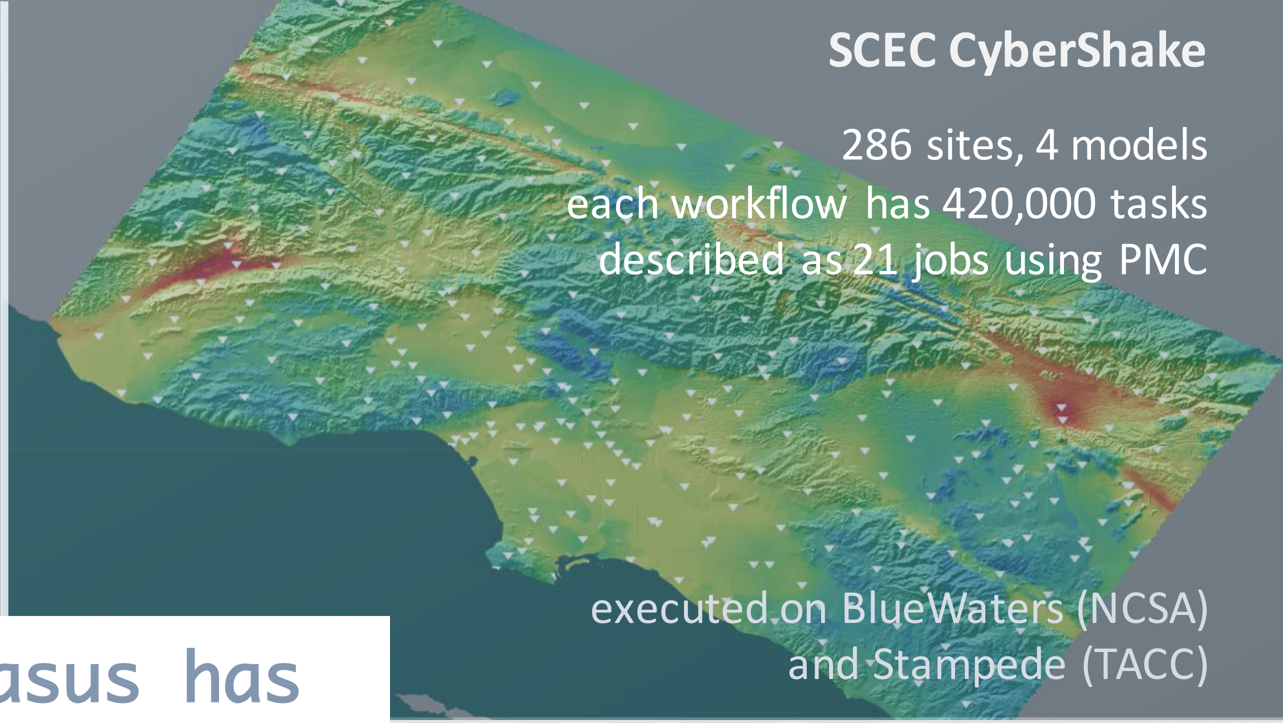
How Pegasus has been used?



Science-grade Mosaic of the Sky (Galatic Plane - Montage)

18 million input images (~2.5TB)
900 output images (2.5GB each, 2.4TB total)
17 workflows, each of which contains
900 sub-workflows (hierarchical workflows)
10.5 million tasks (34,000 CPU hours)

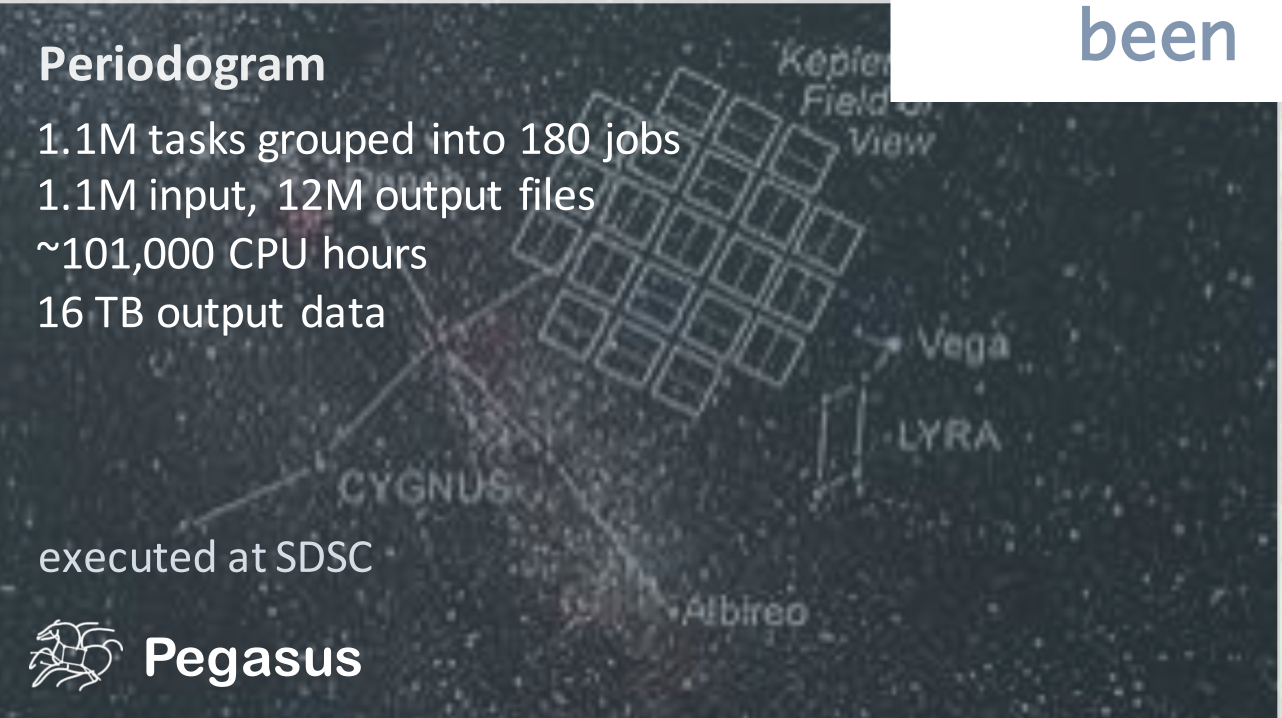
executed on the cloud (Amazon EC2)



SCEC CyberShake

286 sites, 4 models
each workflow has 420,000 tasks
described as 21 jobs using PMC

executed on BlueWaters (NCSA)
and Stampede (TACC)



Periodogram

1.1M tasks grouped into 180 jobs
1.1M input, 12M output files
~101,000 CPU hours
16 TB output data

executed at SDSC

ORNL Spallation Neutron Source (SNS)

5 jobs that consumes about
900 cores for more than 12 hours

executed on Hopper (NERSC)

Pegasus est. 2001

Automate, recover, and debug scientific computations.

Get Started

Pegasus Website

http://pegasus.isi.edu

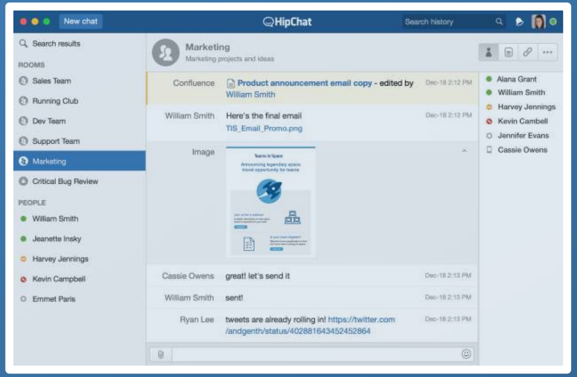
Users Mailing List

pegasus-users@isi.edu

Support

pegasus-support@isi.edu

HipChat



Confluence — Product announcement email copy - edited by William Smith — Dec-18 2:12 PM

William Smith — Here's the final email TIS_Email_Promo.png — Dec-18 2:12 PM

Image

Cassie Owens — great! let's send it — Dec-18 2:13 PM

William Smith — sent! — Dec-18 2:13 PM

Ryan Lee — tweets are already rolling in! https://twitter.com/andgenth/status/402881643452452864 — Dec-18 2:13 PM

Pegasus est. 2001

Automate, recover, and debug scientific computations.

Thank You

Questions?

Rafael Ferreira da Silva

USC Viterbi
School of Engineering
Information Sciences Institute

Meet our team

Ewa Deelman

Karan Vahi

Gideon Juve

Mats Rynge

Rajiv Mayani

Rafael Ferreira da Silva

NSF U.S. DEPARTMENT OF ENERGY NIH