Introduction

Big Data Sciences Era, Data Intensive Computing applications

Scientific communities

Fetch large set of input data

Apply methods between live and archive data

Move data between stages

Perform computations for simulation, analyses, data preparation …

Clean-up intermediate data

Store final results

Why Scientific Workflows ?

General Features

Abstraction, scientists can focus on their research and not computation management

Easy composition and execution

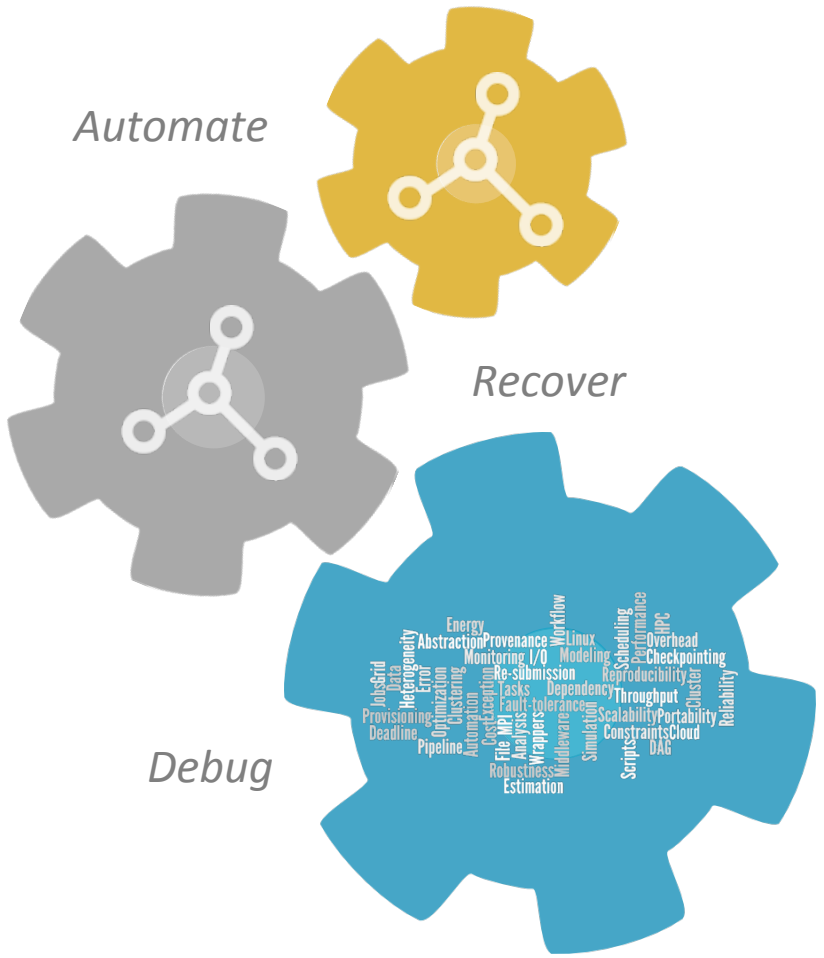
Enables parallel, distributed computations

Different types

abstract vs. concrete

task-flow vs. data-flow

files vs. stream-based



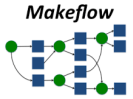
Workflow Management Systems (WMS)

Provide tools to generate the scientific workflow

WORKFLOW MANAGEMENT SYSTEMS

Taverna
https://taverna.incubator.apache.org

Kepler
https://kepler-project.org



Makeflow
http://ccl.cse.nd.edu/software/makeflow

dispel4py
http://dispel4py.org

Nextflow
https://www.nextflow.io

KNIME
https://www.knime.org

VisTrails
http://vistrails.org

FireWorks
https://pythonhosted.org/FireWorks

Swift
http://swift-lang.org

Pegasus
http://pegasus.isi.edu

Asterism Framework

Easy to understood, platform-independent, open-source

Simplifies the development applications running across multiple heterogeneous

How ?

Combining the strengths of

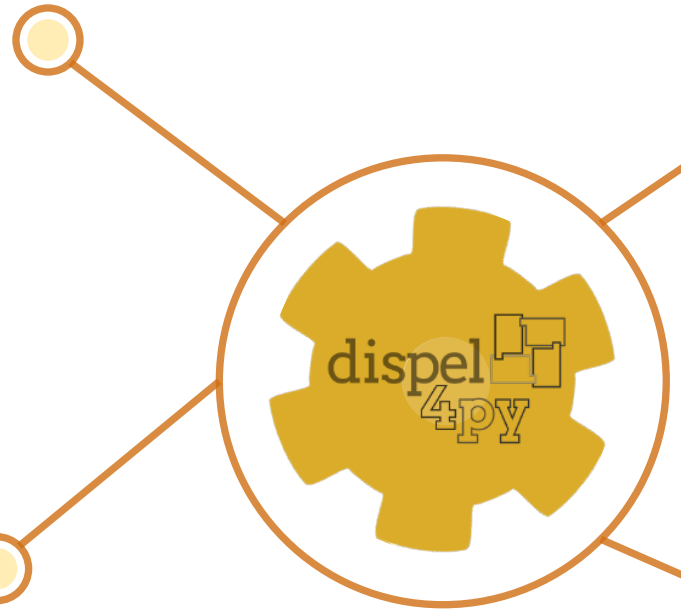
Traditional WMS
Pegasus

Rafael Ferreira da Silva, ISI-USC, US
Ewa Deelman, ISI-USC,US

New stream-based data-flow systems
dispel4py

Rosa Filgueira, BSC-NERC, UK
Amrey Krause, EPCC-UoE, UK
Malcolm Atkinson, Informatics-UoE, UK

dispel4py parallel stream-based dataflow system



Automation

Automates pipeline executions
Concurrent, distributed computations
Stream-based model

Workflow Composition

Python Library
Groupings

High-level stream-based data pipeline
~ Apache Beam for classic & new parallel engines (Keynote)

Mapping

Sequential
Multiprocessing
MPI
Apache Storm
Apache Spark (Prototype)

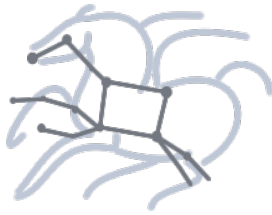
Optimisation

Multiple streams
Avoids I/O operations

Key-features: Automatic parallelization/mappings, concurrent & stream-based

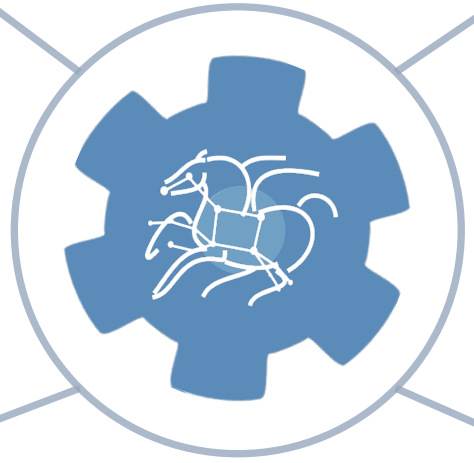
https://github.com/dispel4py/dispel4py

Pegasus workflow system



Automation

Automates pipeline executions
Parallel, distributed computations
Automatically executes data transfers
Heterogeneous resources
Task-oriented model
Application is seen as a black box



Debugging

Workflow execution and job performance metrics
Real-time monitoring, graphs, provenance

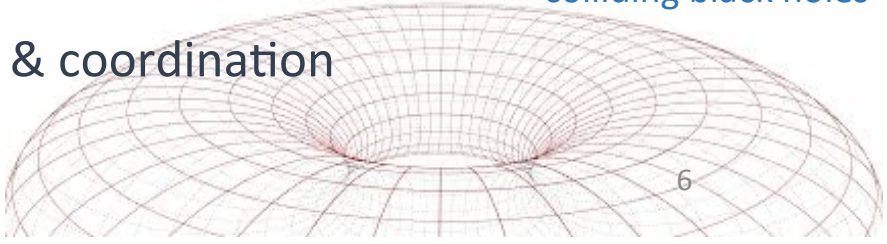
Recovery

Job failure detection
Checkpoint Files
Job Retry
Rescue DAGs

Optimisation

Job clustering
Data cleanup

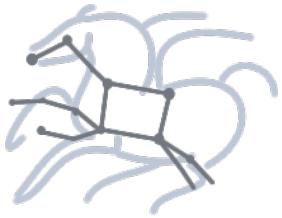
LIGO project:
First **detection of gravitational waves** from colliding black holes

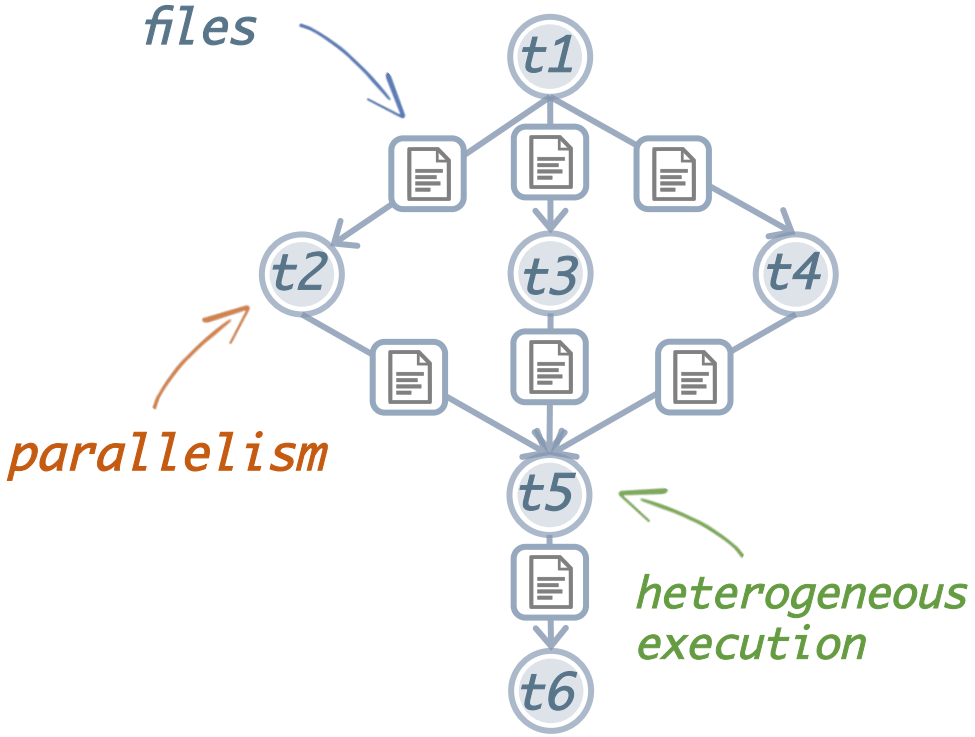


Key-features: Automatic data movement, cleanup, heterogeneous & coordination

http://pegasus.isi.edu

Complementary systems

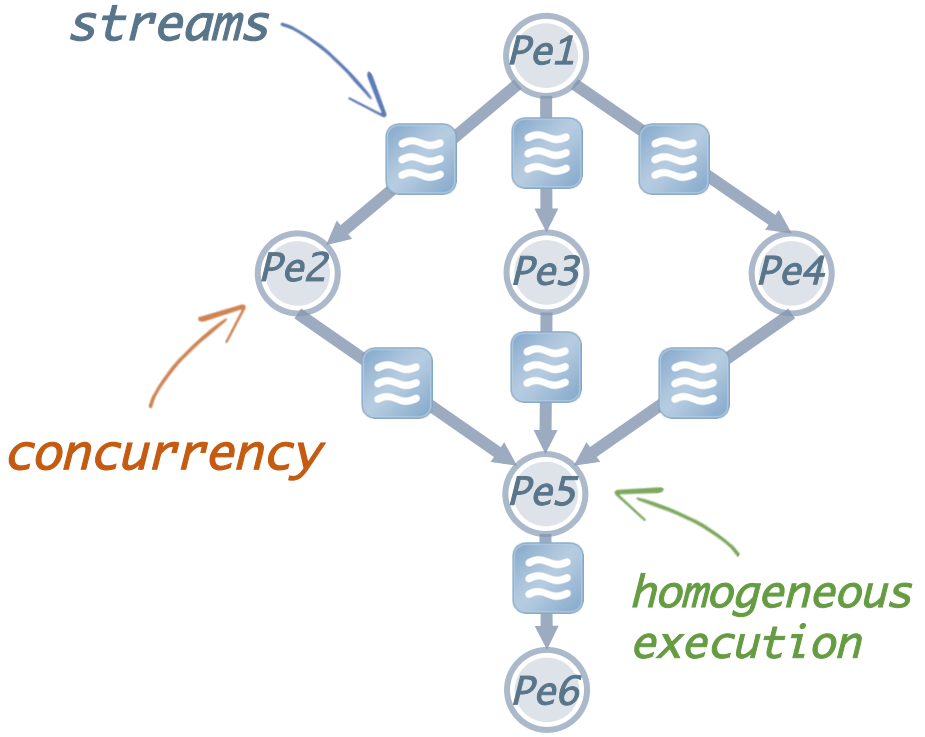




Task-flow

Task: Executable (black box)

dispel4py



Data-flow

Processing Elements: python objects

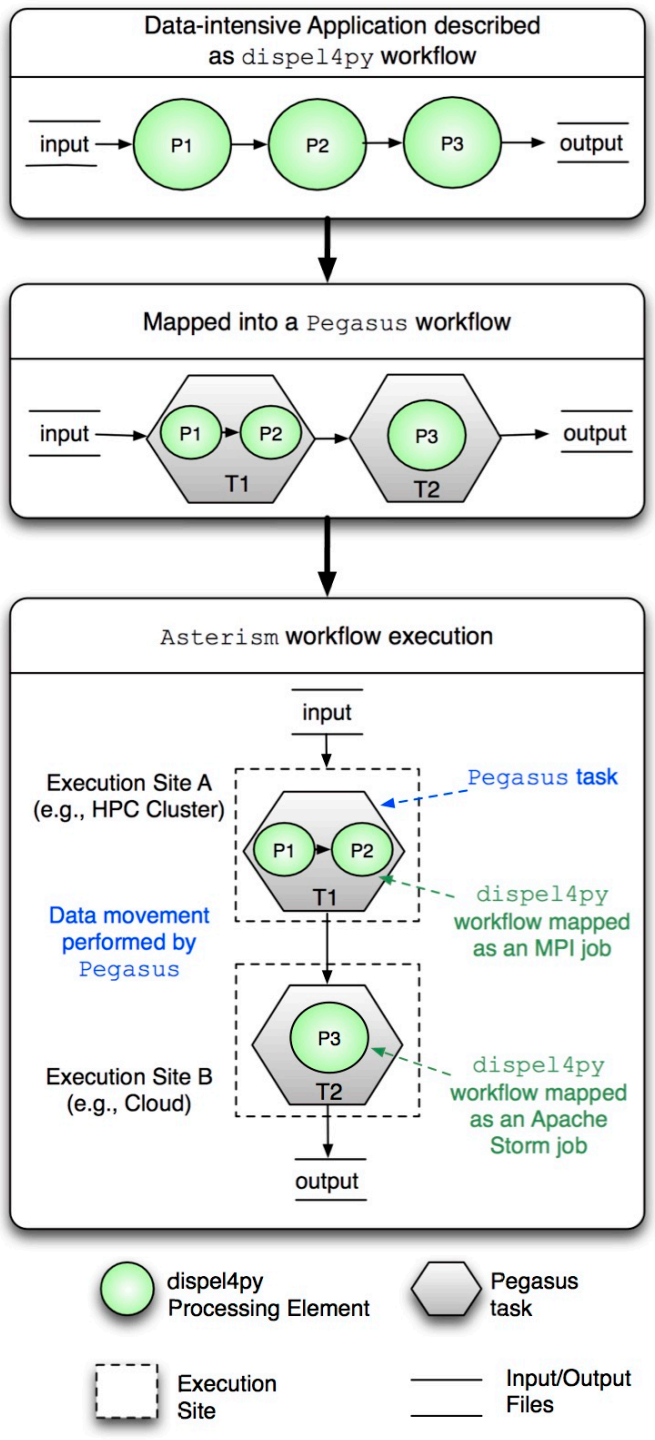
dispel4py

+

Pegasus

=

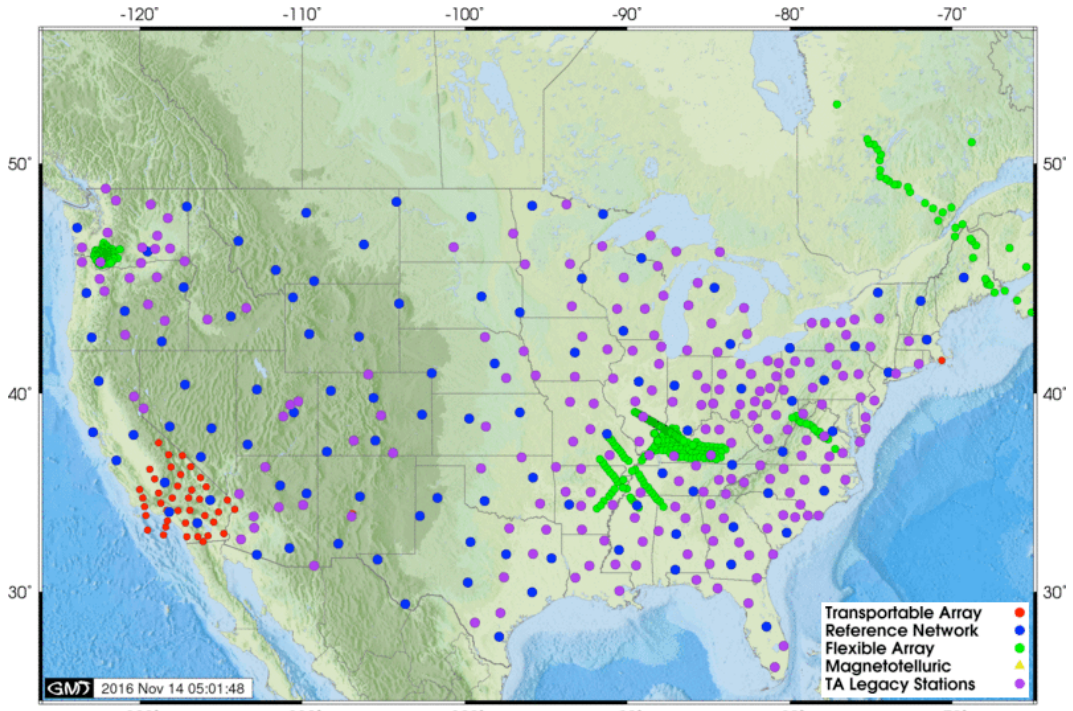
ASTERISM



dispelpy to represent different parts of applications

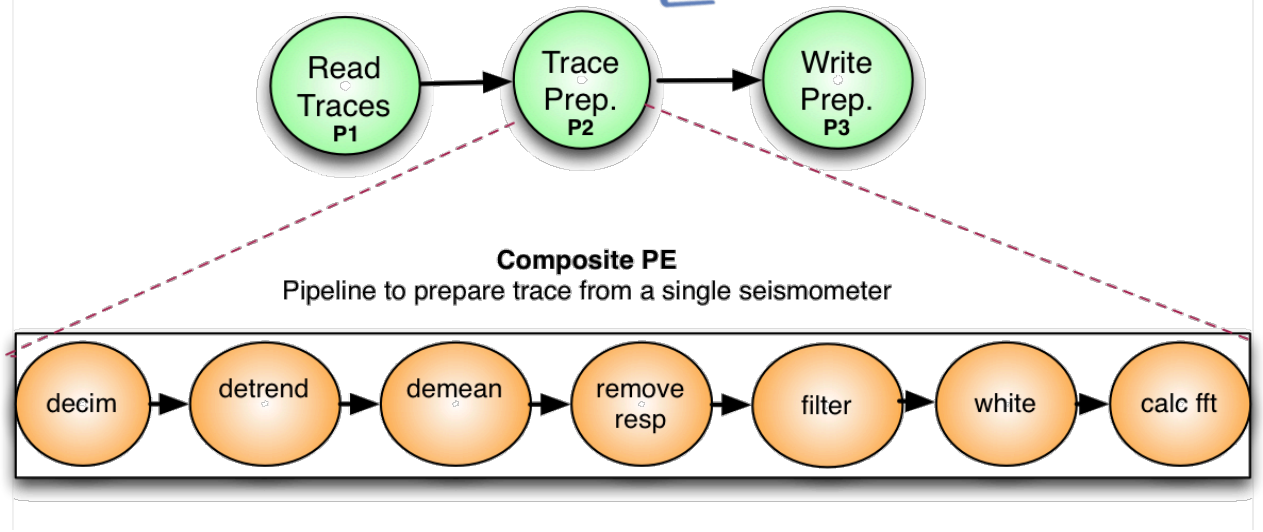
Pegasus to distribute and execute each dispel4py workflow

Seismic Ambient Noise Cross-Correlation

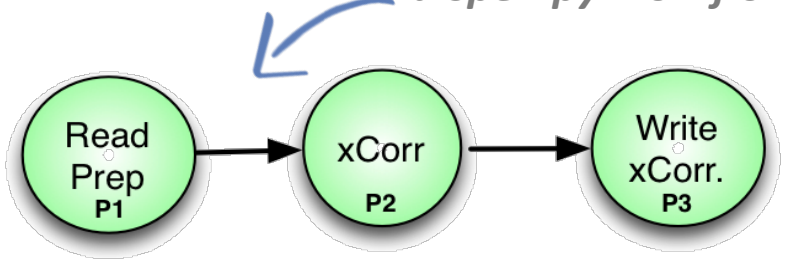


IRIS Usa TA array – 1000 stations

Phase 1
dispel4py workflow



Phase 2
dispel4py workflow



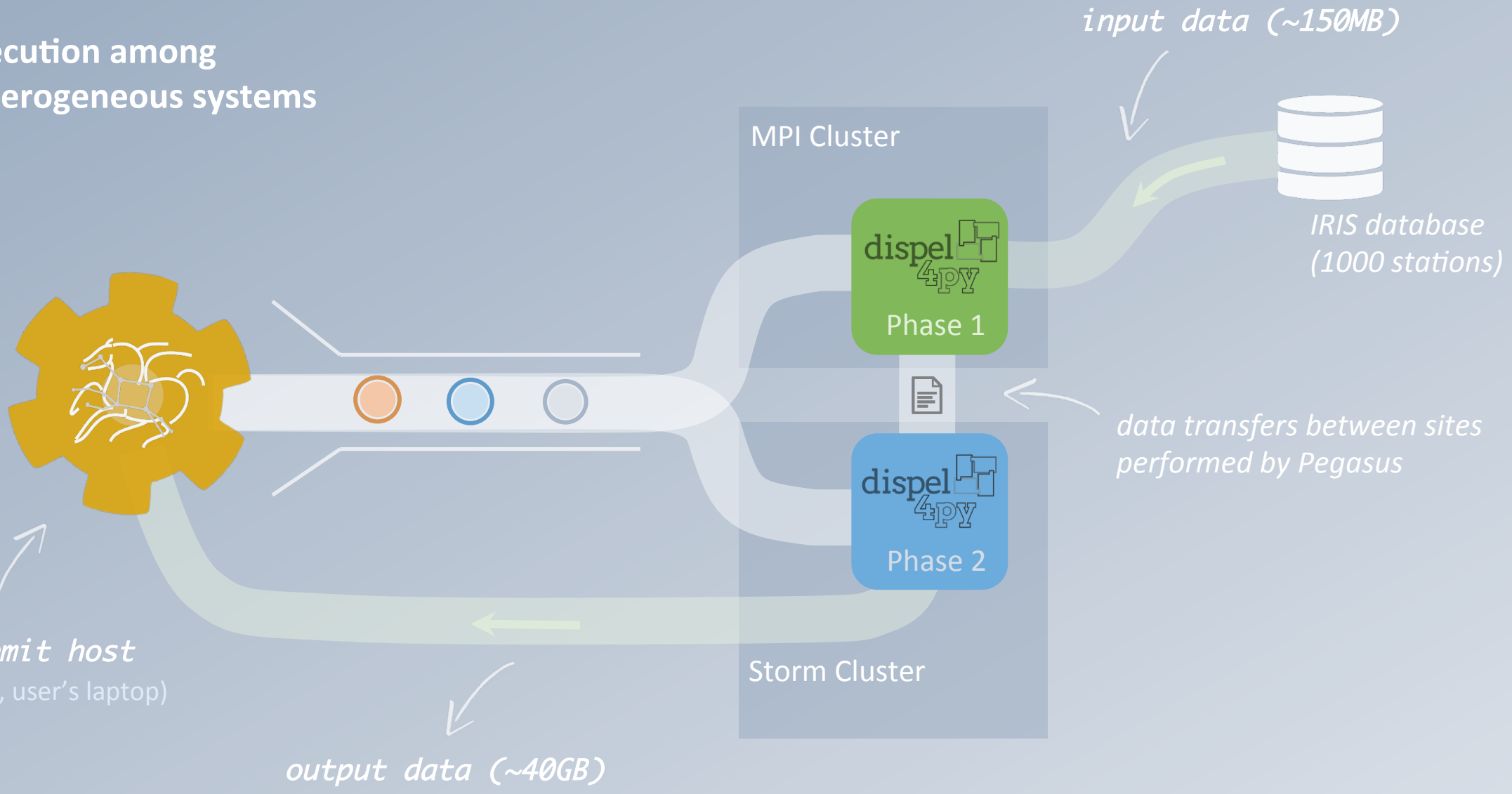
Preprocesses (Phase 1) and cross-correlates traces (Phase 2) from multiple seismic stations

VERCE project [Filgueira *et.al.*] -
both phases on the same HPC resource

[Filgueira *et al.* dispel4py: *A Python Framework for Data-Intensive Scientific Computing*. IJHPCA, 2016]

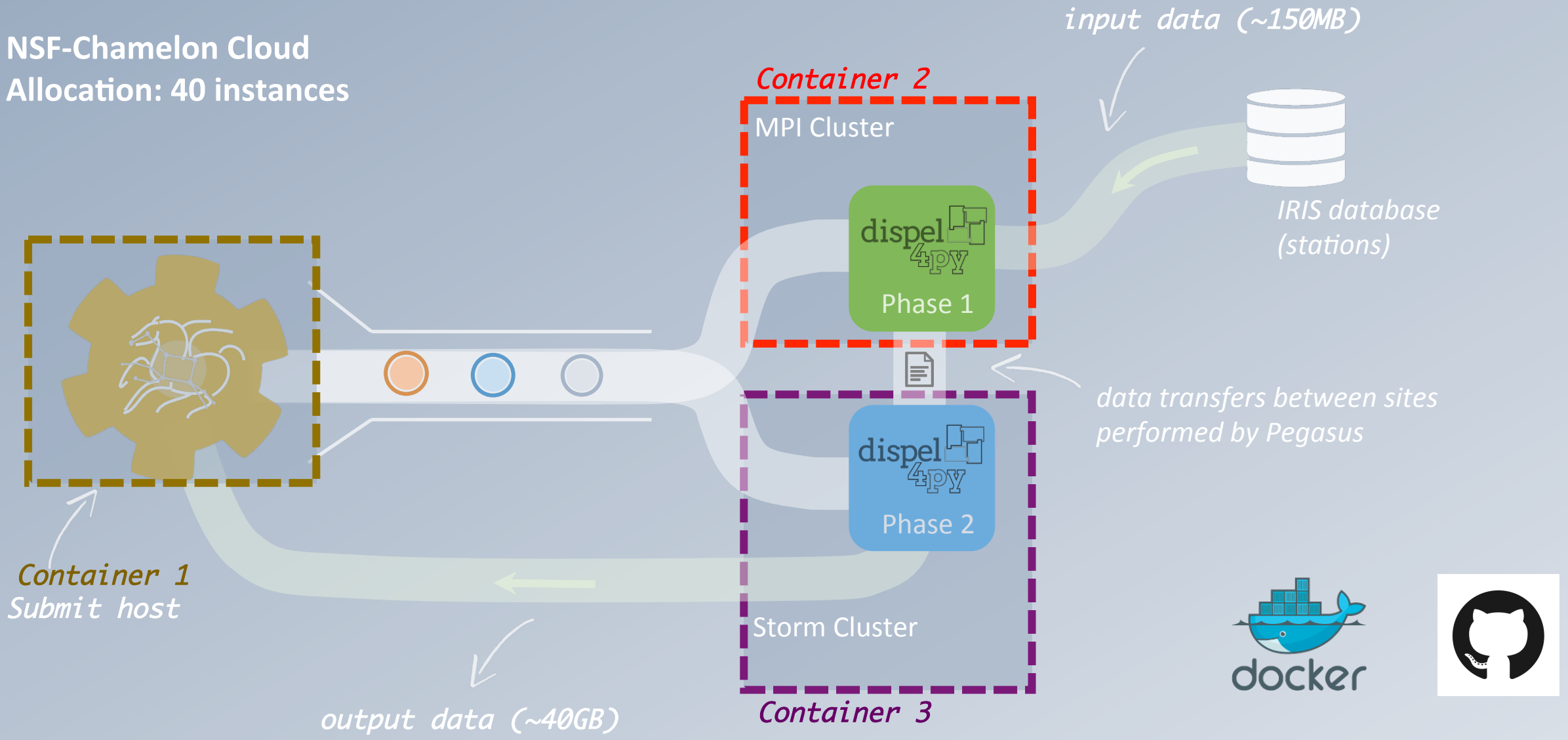
Evaluation- Seismic Ambient Noise Cross-Correlation

Execution among heterogeneous systems



Evaluation- Seismic Ambient Noise Cross-Correlation

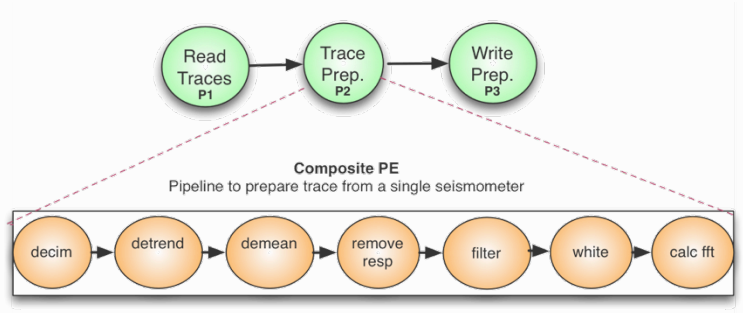
NSF-Chamelon Cloud
Allocation: 40 instances



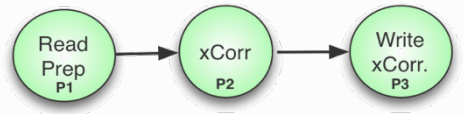
https://github.com/dispel4py/pegasus_dispel4py

Reminder

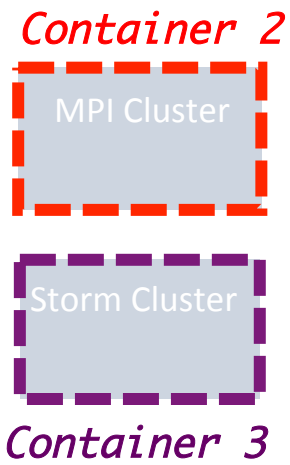
Data-Intensive Application



dispel4py preproc. (Phase 1)



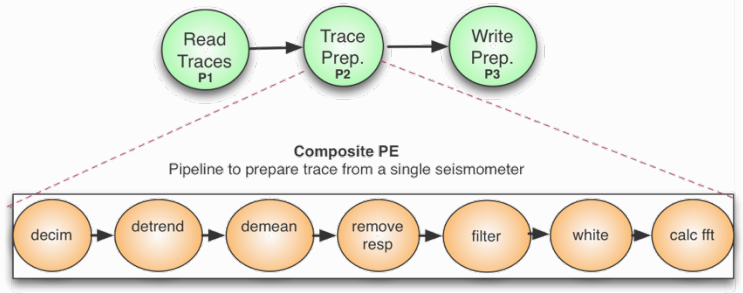
dispel4py proc. (Phase 2)



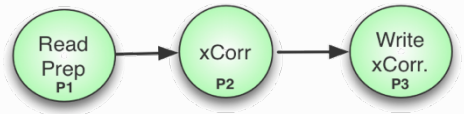
Dockerfiles to configure containers **images** → Stored in our **GitHub** → Linked to **DockerHub** → stored/share/download images

Reminder

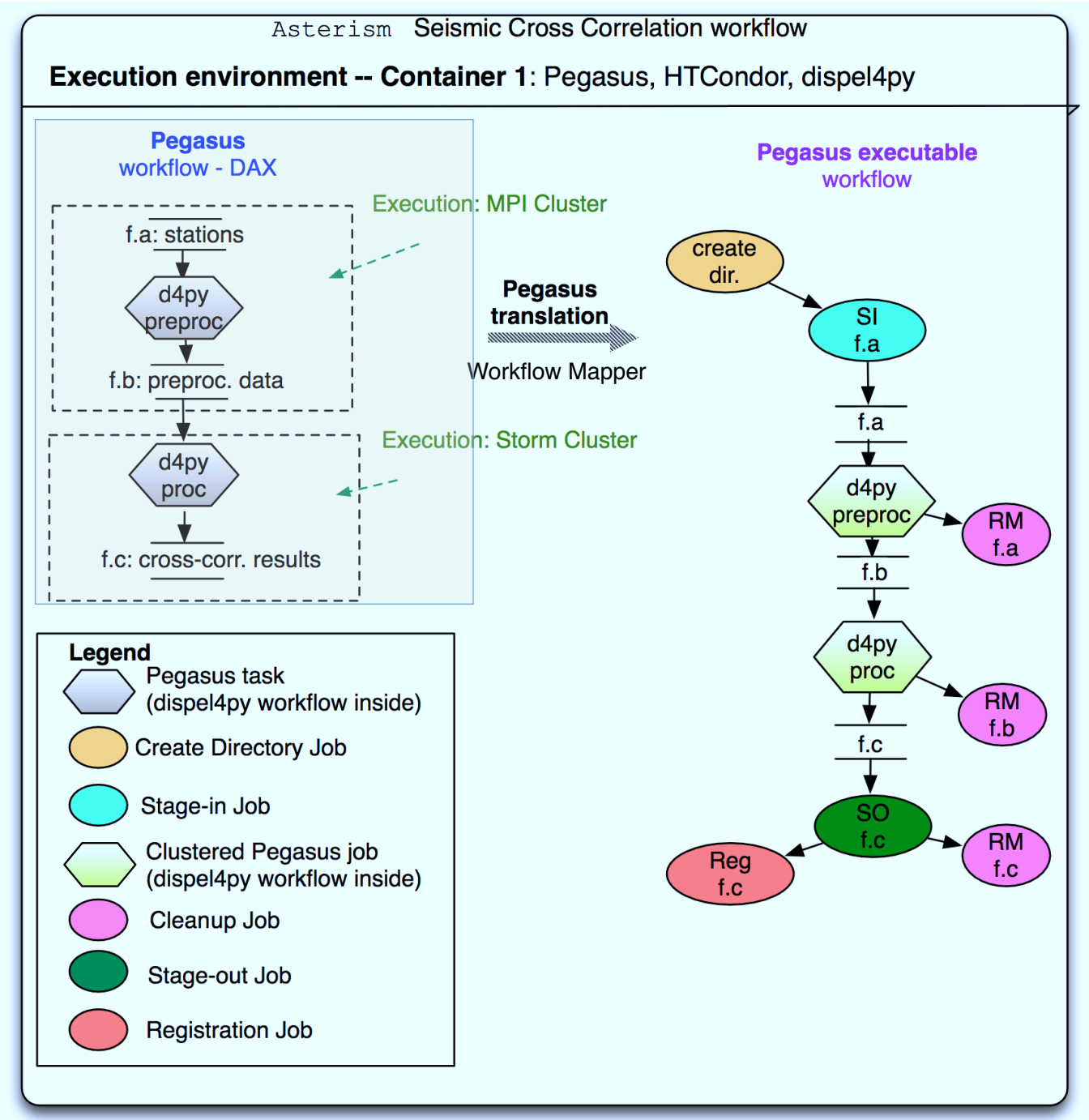
Data-Intensive Application



dispel4py preproc. (Phase 1)



dispel4py proc. (Phase 2)



1 instance as Container 1

Container 1

Submit Asterism

Container 2

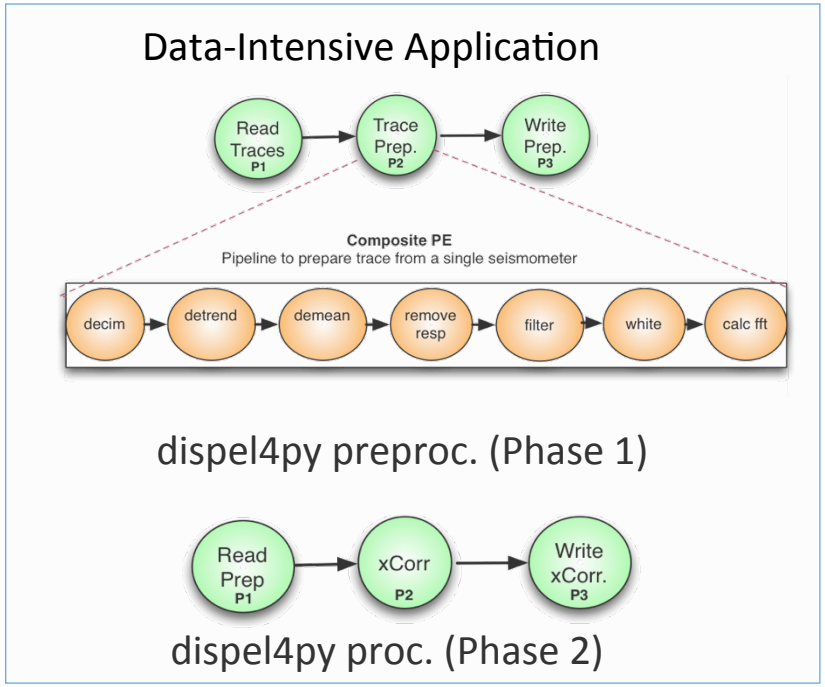
MPI Cluster

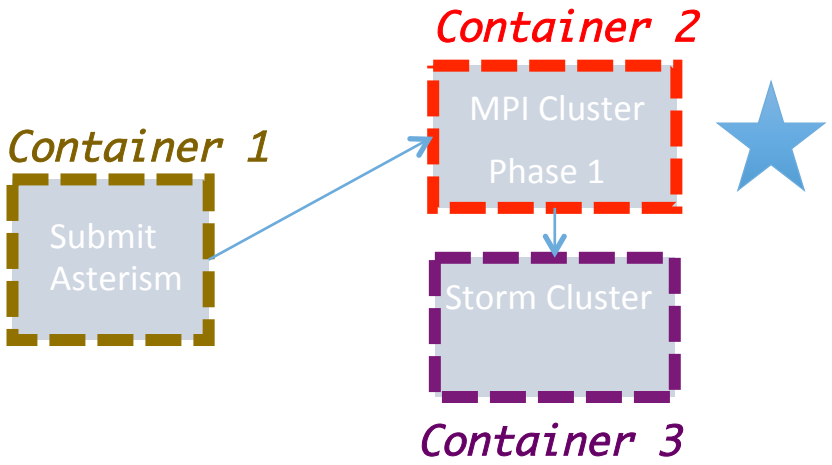
Storm Cluster

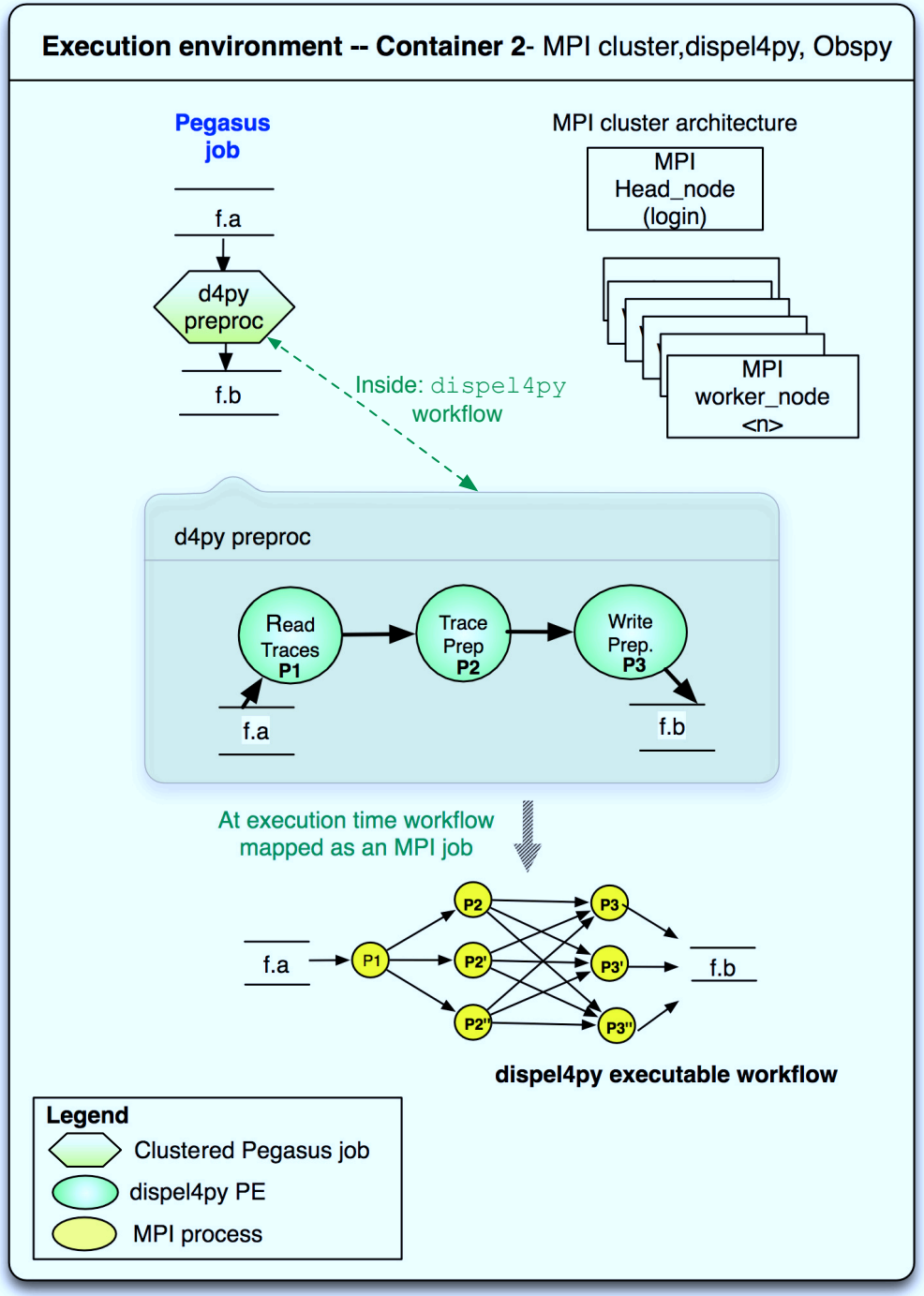
Container 3

Dockerfiles to configure containers **images** → Stored in our **GitHub** → Linked to **DockerHub** → stored/share/download images

Reminder





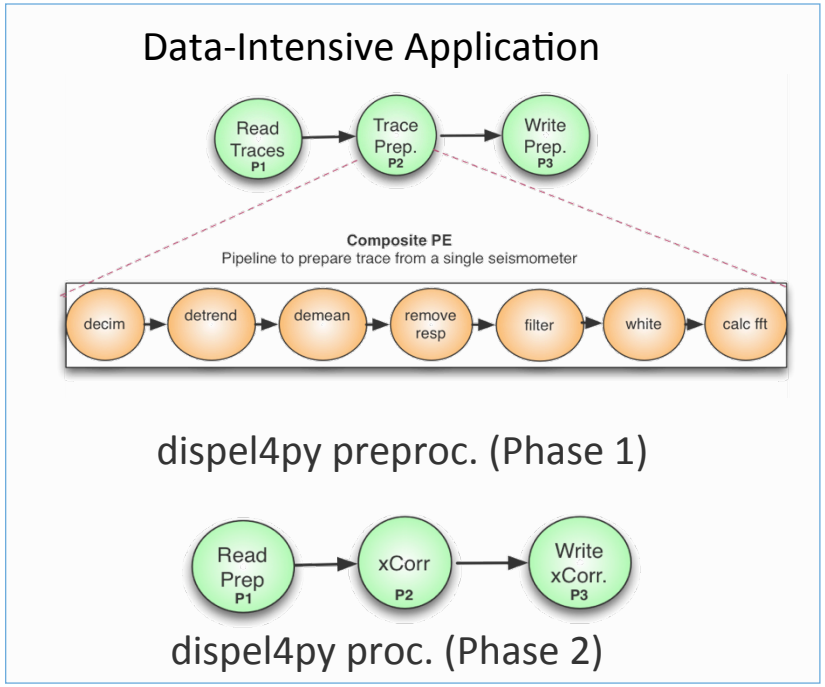


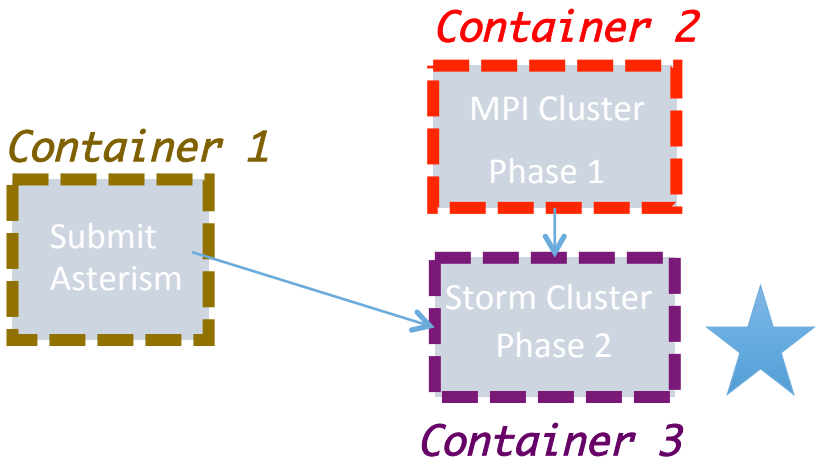
1 instance as Container 2 (MPI head node)

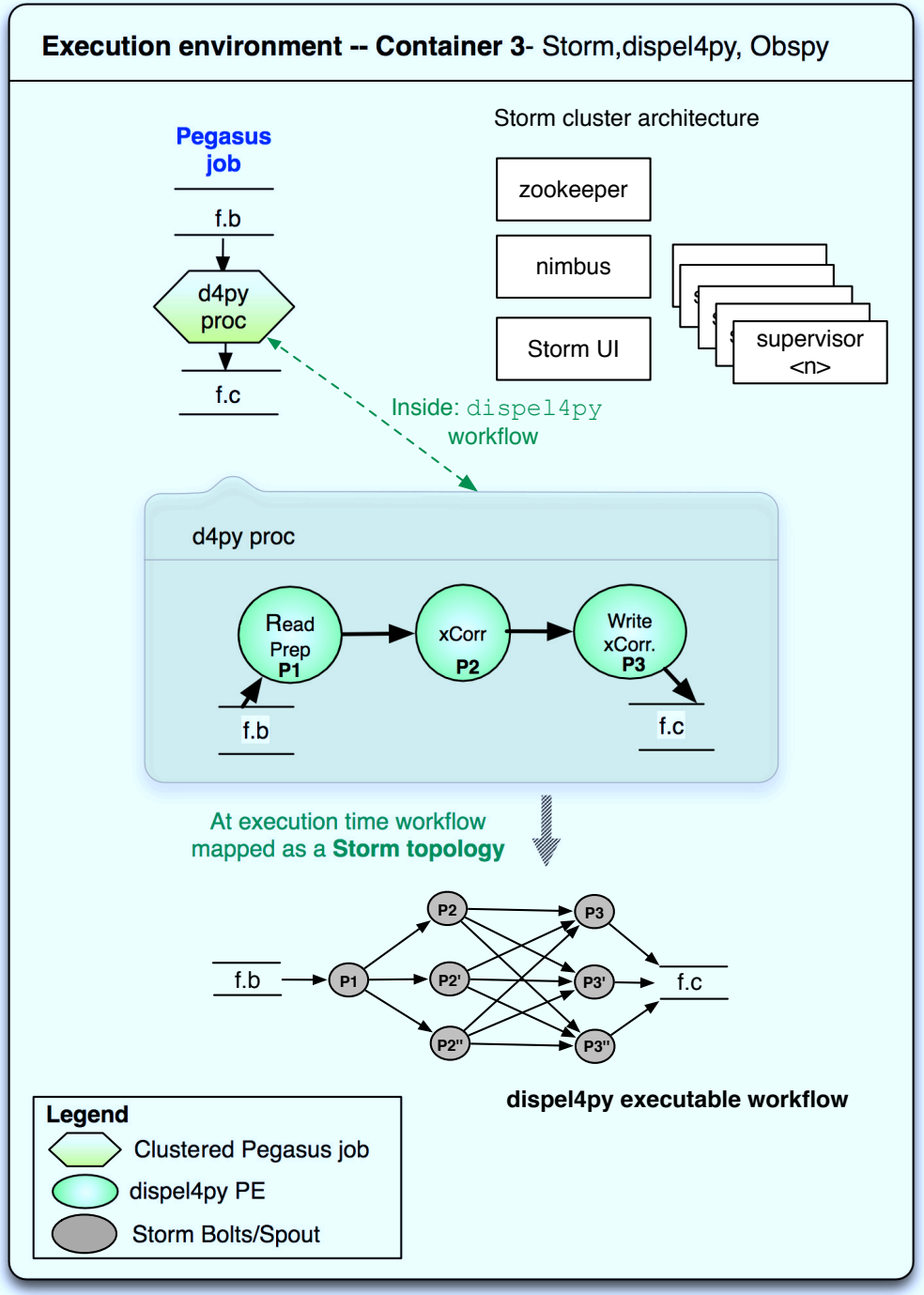
16 instances as Container 2 (MPI workers)

Dockerfiles to configure containers **images** → Stored in our **GitHub** → Linked to **DockerHub** → stored/share/download images

Reminder





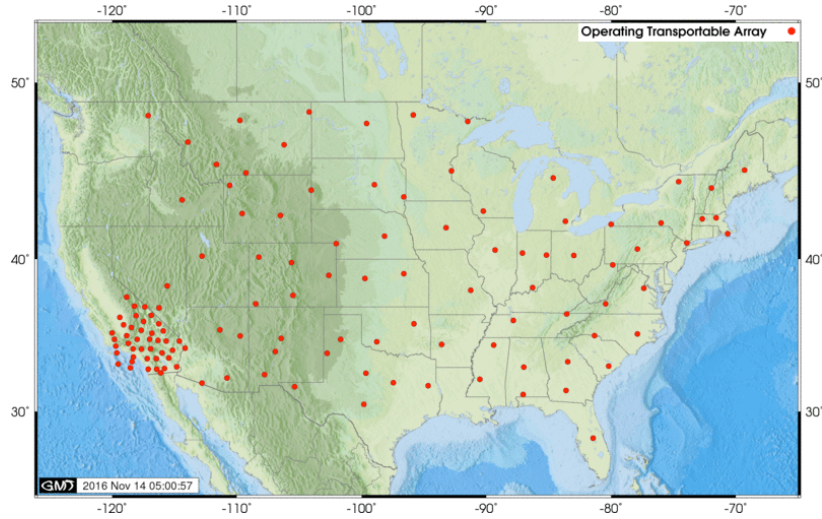


3 instances as Container 3 (zookeper, nimbus, Storm UI)

16 instances as Container 3 (Supervisors)

Dockerfiles to configure containers **images** → Stored in our **GitHub** → Linked to **DockerHub** → stored/share/download images

Asterism Evaluations



Experiment 1: Data from IRIS services (394 stations)

Time

Phase 1 – 8 minutes
Phase 2 – 2 hours
Moving data < 1 minute

Data size

Input data 150MB
Output data 39GB

Experiment 2: Workflow for 3 days requesting data every 2 hours

Scope of this work

Executing & paralyzing automatically data-intensive applications

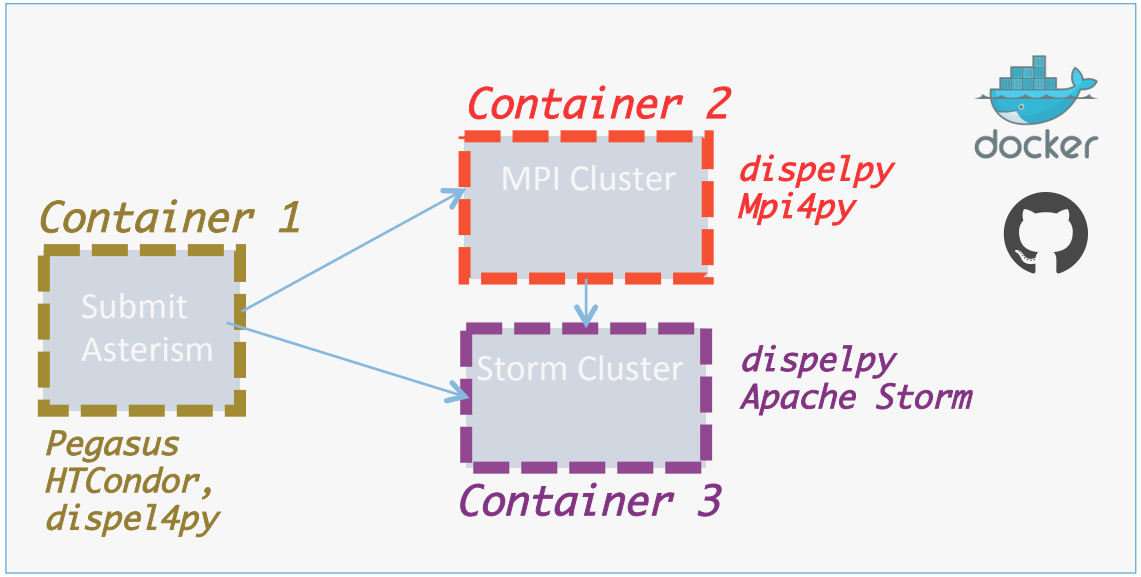
in heterogeneous systems with different enactment engines

Maximizing performance

- both phases in the MPI cluster
- increasing the number of Storm Supervisors

DIaaS: Data-Intensive workflow as a services

- Integrated, complete, easy-to-use, portable approach to run data-intensive workflow
- Packed specialized software
- Reduce the time to build such systems
- Containers turned on when they are needed
- Containers are interchangeable
- Images can be extended



+

ASTERISM framework

=

DIaaS

Conclusions and Future works

Asterism: Easy-to-use system to empower data-driven research

New framework that automatically

- manage the entire workflow, monitor its execution
- handle data transfers between different platforms
- map to different enactment engines at runtime

DIaaS: Data Intensive Workflows as Service

Easy composition & deployment of data-intensive workflows on clouds

Real domain application on the NSF-Chameleon cloud

Future works

More e-Infrastructures and mappings

Asterism via management tool

Asterism: Pegasus and dispel4py hybrid workflows for data-intensive science

Thank You

Questions?

Rosa Filgueira, Ph.D.
rosa@bgs.ac.uk
http://www.rosafilgueira.com
https://github.com/rosafilgueira

Relevant Information

Pegasus Website

http://pegasus.isi.edu

dispel4py GitHub

https://github.com/dispel4py/

Research Object

https://scitech.isi.edu/ro/asterism/