Pegasus Workflow Management System

Gideon Juve
USC Information Sciences Institute
Scientific Workflows

- Orchestrate complex, multi-stage scientific computations
- Often expressed as directed acyclic graphs (DAGs)
- Can execute in parallel on distributed resources
- Capture analysis pipelines for sharing and reuse
Large, Data-intensive Workflows

- **Montage Galactic Plane Workflow**
  - 18 million input images (~2.5 TB / wf)
  - 1000 output images (2.5 GB each, 2.4 TB / wf)
  - 10.5 million tasks (34,000 CPU hours)

John Good (Caltech)
Pegasus Workflow Management System (WMS)

- A collaboration between USC/ISI and the Condor Team at UW Madison
  - Pegasus plans the workflow
  - DAGMan is the workflow engine
  - Condor schedules jobs and manages resources

- Started in 2001

- Actively used by many applications in a variety of domains
  - Earth science, physics, astronomy, bioinformatics, others
  - http://pegasus.isi.edu/applications
Pegasus WMS

API Interfaces
- Python
- Java
- Perl

Portals
- hubzero
- Grayson
- Triana
- Wings

Pegasus WMS
- Mapper
- Engine
- Scheduler
- Monitoring
- Logs

Clouds
- Cloudware
  - OpenStack
  - Eucalyptus
  - Nimbus

Compute
- Amazon EC2
- RackSpace
- FutureGrid

Storage
- S3

Distributed Resources
- Campus Clusters
- Local Clusters
- Open Science Grid
- XSEDE

Other Workflow Composition Tools:
- GRAM
- PBS
- LSF
- SGE

MIDDLEWARE

COMPUTE

STORAGE

USC Viterbi
School of Engineering
DAGMan – Directed Acyclic Graph Manager

- Enables dependencies between jobs
- Provides reliability
  - Retries for job-level failures
  - Rescue DAGs for workflow-level failures
- Other features
  - Throttling
  - Priorities
  - Pre- and Post-Scripts
  - Workflow of workflows

Flowchart:

A → B → C → D

Jobs:
- JOB A a.submit
- JOB B b.submit
- JOB C c.submit
- JOB D d.submit

Parent/Child relationships:
- PARENT A CHILD B C
- PARENT B C CHILD D
Pegasus Planner

- Pegasus planner compiles abstract workflows into executable workflows
  - Adds data management jobs (transfer, cleanup, registration)
  - Performs optimizations (task clustering, reduction, etc.)
  - Generates executable artifacts (DAG, submit scripts, etc.)

- Enables portability and optimization
Other Pegasus WMS Features

- Monitoring and Troubleshooting
  - Web GUI and command-line tools for reporting and investigating workflow progress and failures

- Hierarchical Workflows
  - “Workflow of Workflows”

- Provenance
  - Generates a database with information about when and where jobs were executed

- Others
  - MPI task clustering (pegasus-mpi-cluster)
  - Script generation
  - Notifications
  - …
Distributed Workflow Challenges/Requirements

- **Security and identity management**
  - Different credentials on different sites / clusters
  - Complexity of grid security, X.509
  - Firewall and network issues, e.g. when using Glideins

- **Job submission**
  - Local: Dedicated login node / submit host
  - Remote: Middleware, e.g. Globus GRAM

- **Data transfer**
  - Need high-performance transfer tools, e.g. GridFTP
  - Many different protocols/tools (scp, GridFTP, http, irods, srm, s3)
  - Complex storage configurations
Job Submission in Pegasus

- Personal Condor
- Local Condor pool
- Local Batch (PBS, SGE, etc.)
- Remote Condor pool (Condor-C, flocking)
- Batch using GRAM, UNICORE, ARC
- Glideins with BOSCO or glideinWMS
- Batch using SSH w/ BOSCO (scalability?)
Distributed Data Management in Pegasus

- Pegasus supports many different data configurations
  - Complex data flows
  - Many protocols and tools
  - Push, pull, and 3rd party transfers
  - Inter-site transfers
  - Squid, get http, put !http

- Site Types
  - Local site: Pegasus WMS
  - Storage site: inputs and outputs
  - Staging site: intermediate
  - Compute site: compute jobs
Data Management Configurations

Shared File System

- Submit Host
- WN
- WN
- Shared FS
- Compute Site

Non-shared File System

- Submit Host
- WN
- WN
- Compute Site
- Staging Site

Condor IO

- Submit Host
- Local FS
- WN
- WN
- Compute Site

- Jobs
- Data

Examples:
- e.g. HPC Cluster
- e.g. Amazon EC2 w/ S3
- e.g. Open Science Grid
Workflows and Science Gateways

- HubZero
- Pegasus as a Service

Credit: Frank McKenna, UC Berkeley, NEES, HUBzero
More Information

- **Website:** [http://pegasus.isi.edu](http://pegasus.isi.edu)

- **Email:**
  - pegasus-users@isi.edu
  - pegasus-support@isi.edu

- **Pegasus Team @ ISI**
  - Ewa Deelman, Karan Vahi, Gideon Juve, Mats Rynge, Rajiv Mayani
CC-NIE: ADAMANT

Pegasus Workflow

1. Start workflow
2. Dynamically create compute nodes
3. Compute intensive workflow step
4. Dynamically destroy compute nodes
5. End workflow

ORCA Dynamic Slice

1. Few compute nodes for beginning steps
2. Add compute nodes for parallel compute intensive step
3. Dynamically provision network between cloud sites
4. Free unneeded compute nodes after compute step

Jeff Chase, Duke
Ewa Deelman, USC/ISI
Ilya Baldin, UNC CH/RENCI
Charles Schmitt, UNC CH/RENCI