Wrangler: Virtual Cluster Provisioning for the Cloud

Gideon Juve and Ewa Deelman
USC Information Sciences Institute
{gideon,deelman}@isi.edu

Problem Statement

- Unlike traditional HPC platforms, such as clusters and grids, clouds are not pre-configured to run science applications.
- Science applications require many different cluster-level services, such as schedulers, parallel and distributed file systems, metadata services, file transfer services, and distributed caches.
- Configuring cluster-level services in a dynamic way is difficult. For clusters with more than a few nodes, manual configuration is not practical because of the time required and the risk of errors.
- In order to make it feasible to run large-scale science applications in the cloud, we need tools to automate the provisioning and configuration of virtual clusters.

Virtual Clusters

- Dynamic clusters created on-demand by provisioning resources from a cloud and configuring them with services that can be used to run parallel and distributed applications.
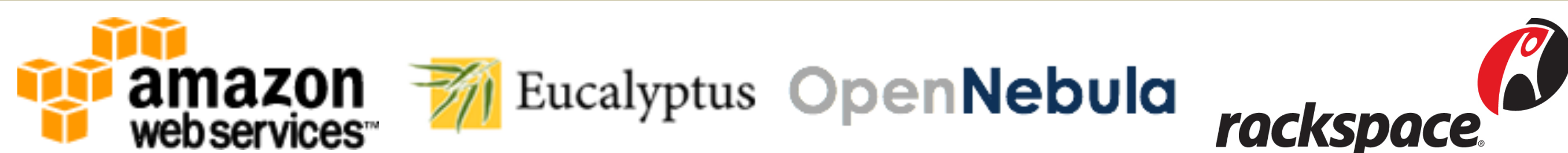
Wrangler

- A service for provisioning and configuring virtual clusters
- User specifies the virtual cluster configuration, and Wrangler provisions the nodes and configures them according to the user's requirements
- Users can specify custom roles for nodes by writing simple scripts
- **Features:** XML format for describing virtual clusters, support for multiple cloud providers, node dependencies and groups, automatic distribution of configuration files and scripts
- **Interfaces:** REST+JSON, Python API, command-line utilities

Nodes, Groups and Roles

- Each virtual cluster in Wrangler is defined in terms of **nodes**, **groups**, and **roles**
- **Nodes** are individual hosts in the virtual cluster. Each node has a resource type, and a configuration, which can specify multiple roles.
- **Groups** are collections of nodes that depend on each other. Nodes within a group are all configured at the same time. This is useful for configuring services that require every node to know about all other nodes, such as some parallel file systems and peer-to-peer services.
- **Roles** define the scripts, configuration files, and parameters that need to be applied to a node to configure it for a given service. Users can define custom roles by writing simple scripts that are *automatically distributed* to the nodes during provisioning. Role scripts can be written in shell, Perl, Python or any other scripting language.
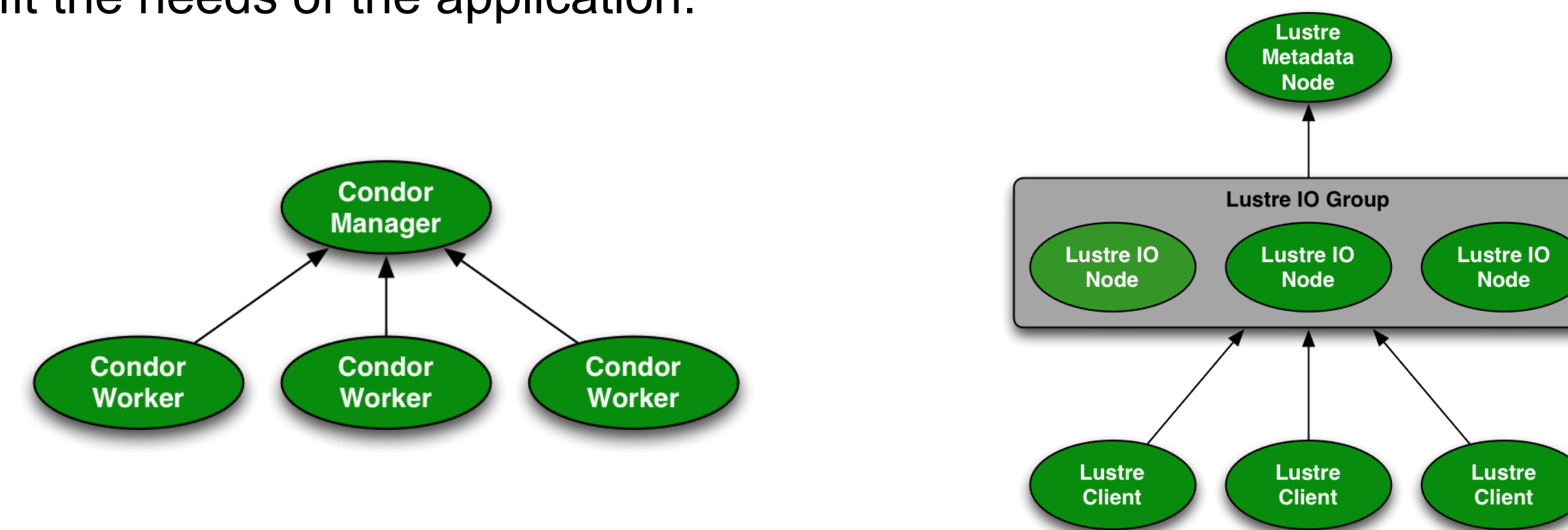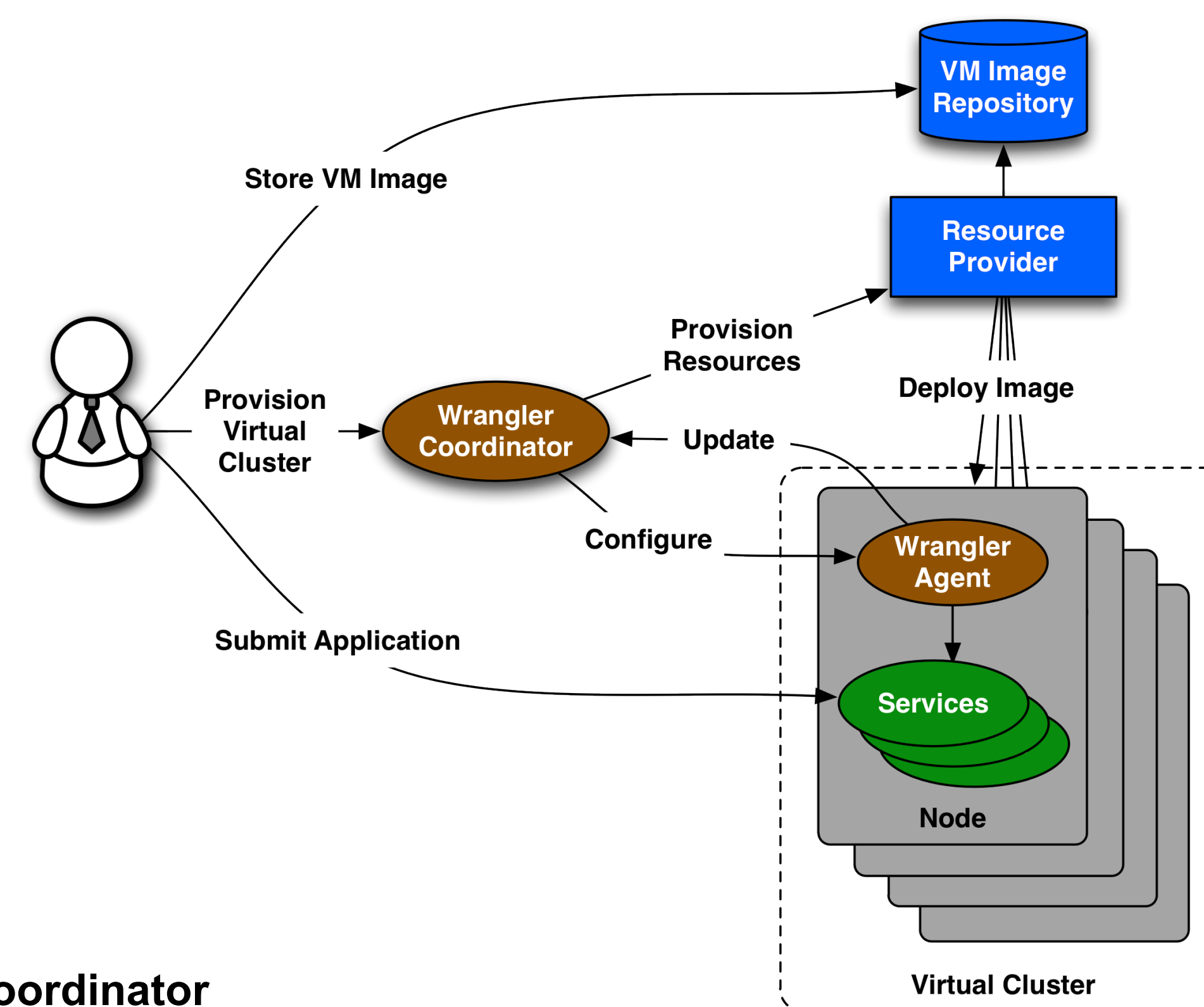
Resource Providers

- Each node in Wrangler is associated with a resource configuration which specifies the resource provider and virtual machine configuration for that node.
- Wrangler supports several commercial and open-source resource provider interfaces, including EC2, Eucalyptus, Open Nebula, and Rackspace.

Dependencies

- Nodes and groups can depend on one another.
- Wrangler will not configure a node until all the other nodes that it depends on have been configured. This enables services to be started in the correct order.
- Nodes can be added if their dependencies are configured or removed if they have no dependencies. This allows the virtual cluster to be dynamically sized to fit the needs of the application.

Architecture

Wrangler Coordinator
- Accepts requests from users to create virtual clusters
- Contacts resource providers to provision nodes
- Stores and distributes user-defined role scripts, configuration files and parameters
- Acts as an information broker to help nodes discover information about each other and about their roles

Wrangler Agent
- Contacts the resource provider's information service to discover its network addresses and other pertinent information
- Reports node status, metadata, and monitoring information to the Coordinator
- Fetches information about the node's roles and peers from Coordinator
- Applies roles to node according to user-defined role scripts
- Periodically sends status updates and monitoring information to Coordinator

Example Roles

- **condor_manager**: Configures a node to run a Condor scheduler.
- **condor_worker**: Configures a node to connect to a Condor manager and execute jobs.
- **nfs_server**: Export an NFS file system and ensure that all the services required for NFS are running.
- **nfs_client**: Mount a remote NFS file system.
- **glusterfs_peer**: Configure node to act as both a storage node and a client in a GlusterFS file system.
- **ec2_init**: A provider-specific role for Amazon EC2 that mounts the ephemeral storage devices on an EC2 VM.

Virtual Cluster Description


Running Scientific Workflows

- We have used Wrangler to set up virtual clusters for running scientific workflow applications in the cloud.
- In these configurations, we use Wrangler to provision nodes to add to our existing Condor pool, which is hosted on a submit node outside the cloud.
- With Wrangler it is easy to combine local resources with cloud resources. It is even possible to create virtual clusters across several clouds.

GlusterFS Configuration
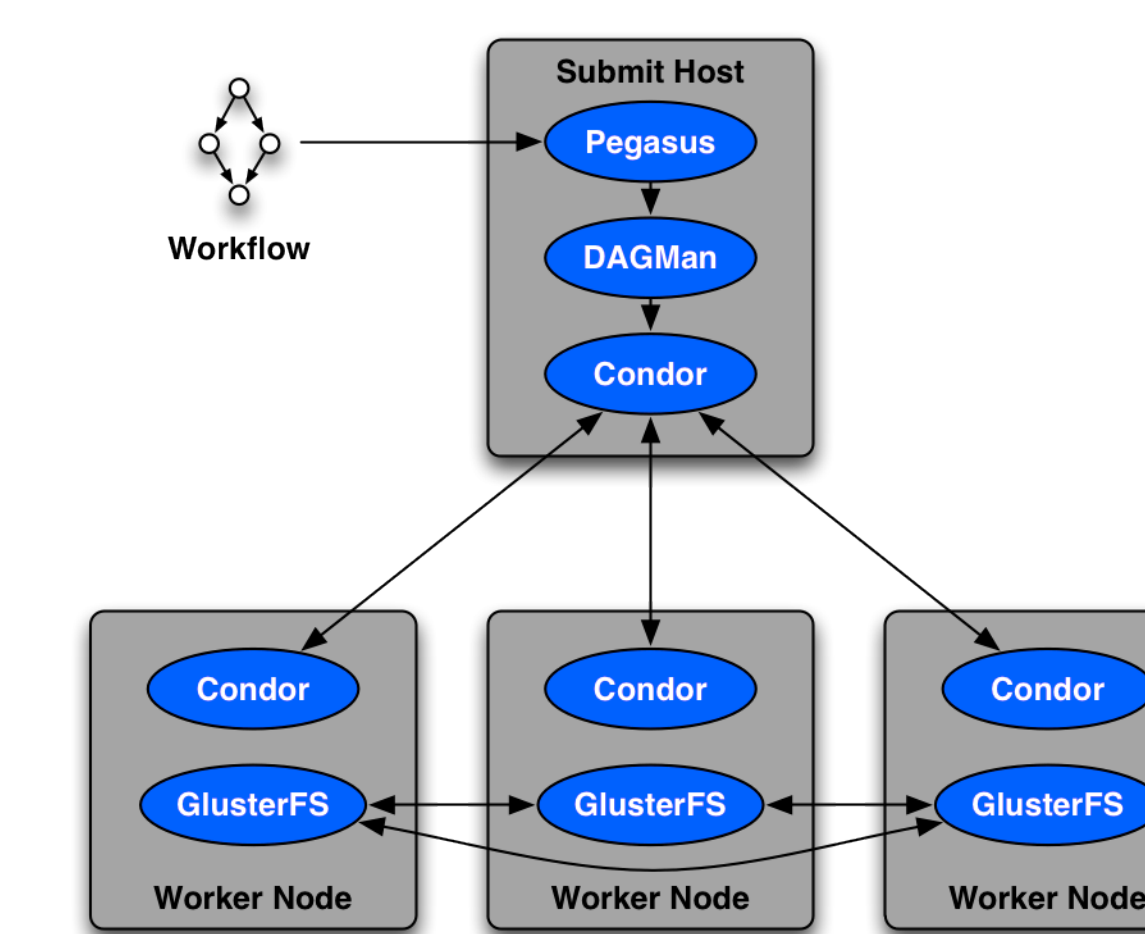
In this configuration we provision several worker nodes in the cloud and start Condor worker processes on them. In addition, we set up a shared GlusterFS file system that spans the nodes.

NFS Configuration
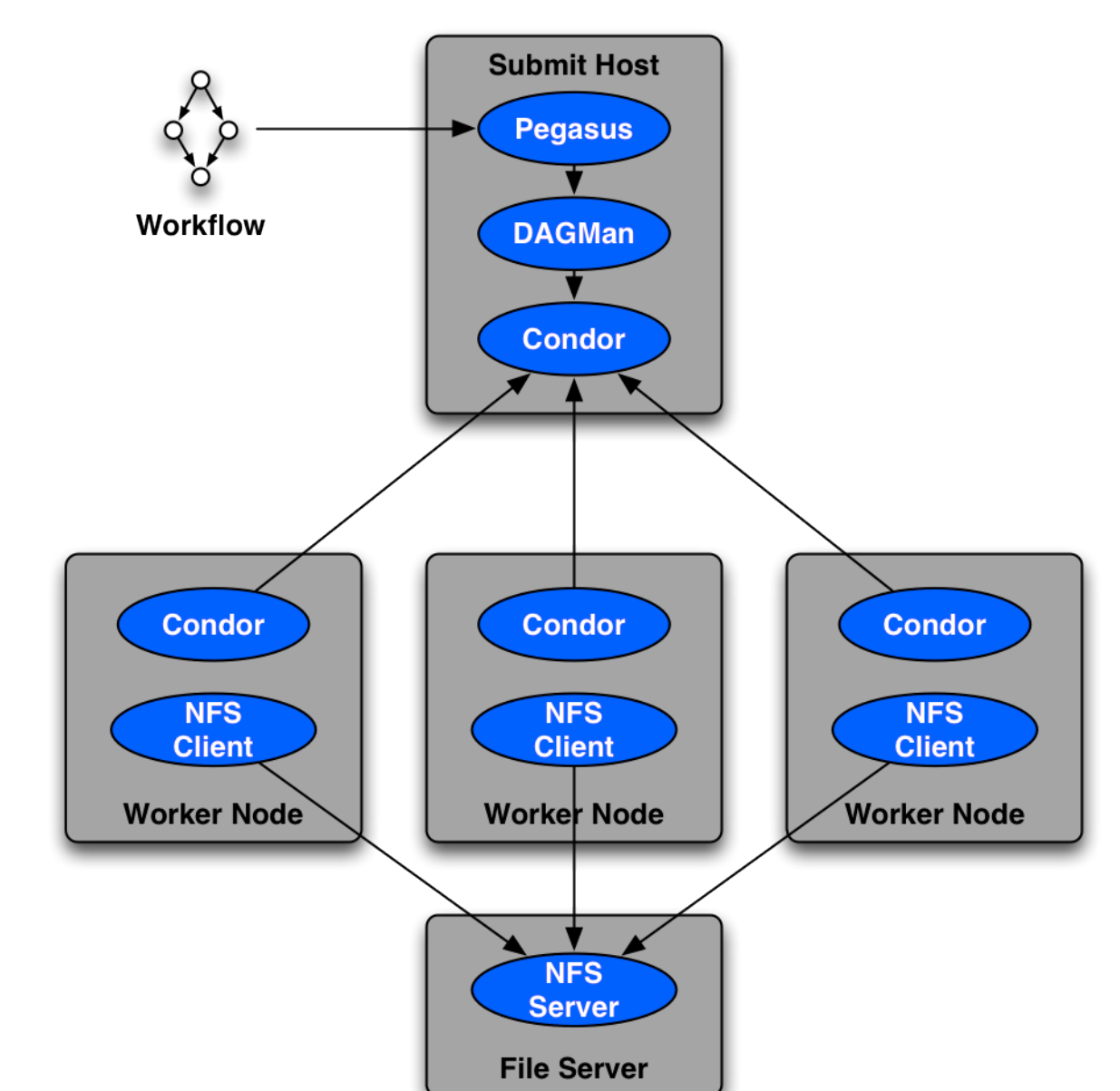
In the case of NFS, the worker nodes mount a shared NFS file system that is hosted by another node. Wranger makes it easy for nodes to depend on one another.

Acknowledgements