Harvester

Tadashi Maeno (BNL)

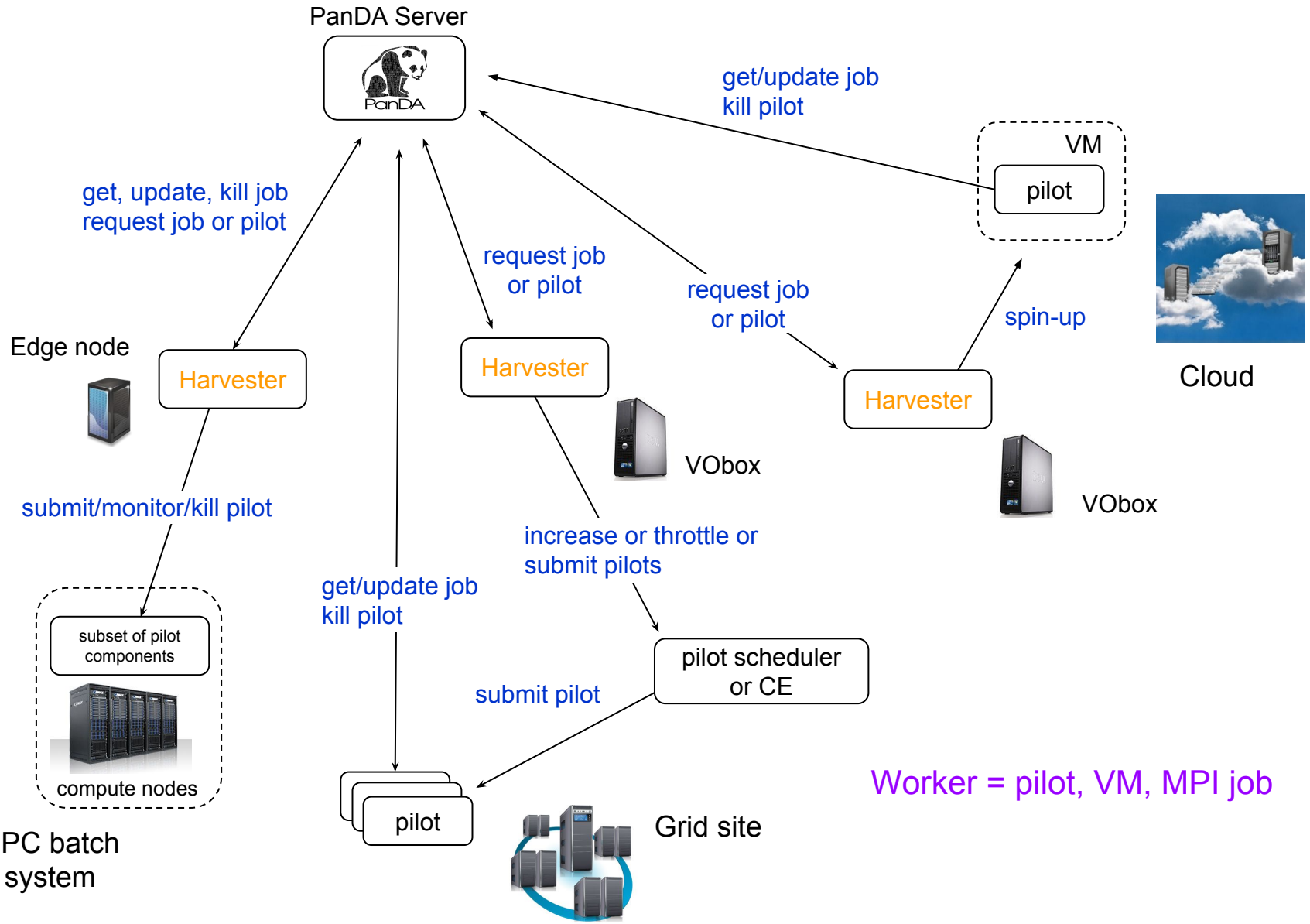
Introduction

- PanDA currently relies on server-pilot paradigm
  - PanDA server maintains state and manages workflows with various granularities, such as task, job, and event
  - Pilots are job-centric and independently run on worker nodes (WNs) with limited view of local resource
- Works well for the grid with 250k cores 24x7 as underlying resources are not very heterogeneous
  - But missing capability to dynamically optimize resource allocation among queues (score, mcore, himem, long, ...)
- Not very well for HPC or large-scale clouds
  - Each HPC has a different edge service and operational policy, leading to over-stretched pilot architecture and incoherence in implementation at different HPCs
  - PanDA itself has no means of managing and monitoring cloud utilization by using native cloud API

Introduction

- New model : server-harvester-pilot
  - Harvester is a resource-facing service between PanDA server and collection of pilots
  - Stateless service with knowledge of resource
  - Modular design for different resource types and workflows
  - Intermediate communication channels between PanDA server and pilots for some workflows
  - Many harvester instances running in parallel

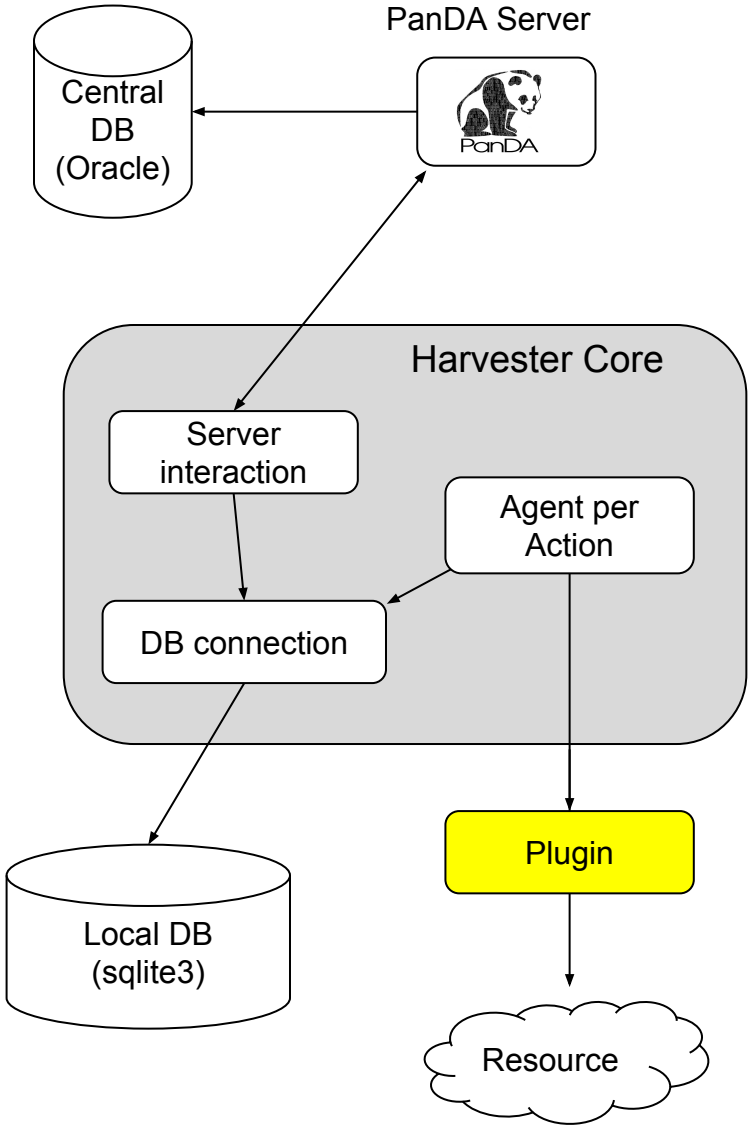
Schematic View



Objectives

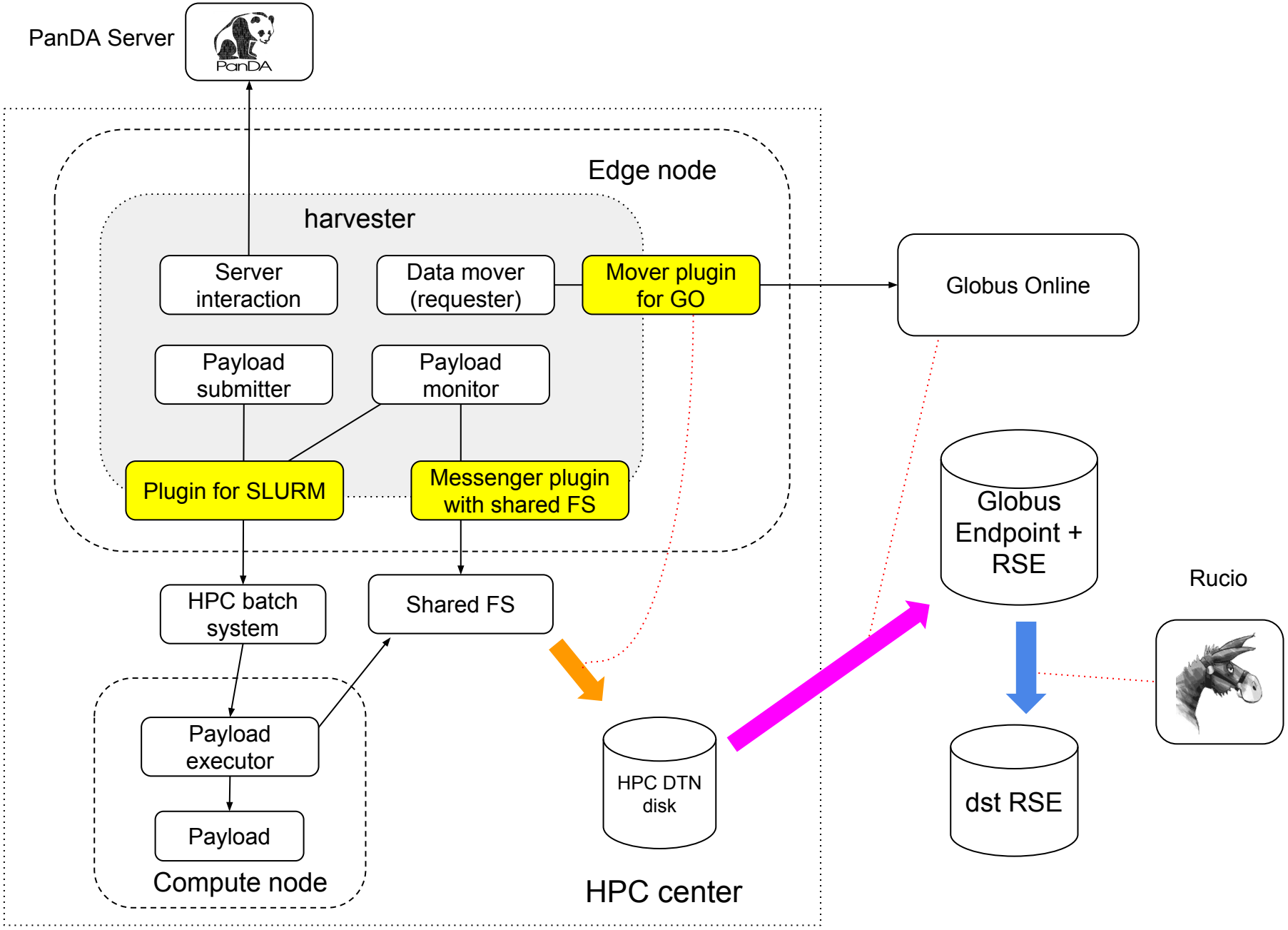
- To have a common machinery for all computing resources
- To provide a commonality layer in bringing coherence to HPC implementations
  - Core + plugins for Yoda/Droid, multi-job pilot, SAGA, synchronous/asynchronous staging, ...
- To add a capability for timely optimization of CPU allocation among various resource types
  - Dynamic resource partitioning
  - Awareness of association between multiple resources such as MCORE and SCORE
- To have better resource monitoring
  - Almost blind right now until the pilot gets a payload
- To have better integration between PanDA system and resources for new workflows
  - Job/event-level late-binding and jumbo jobs

Harvester Architecture

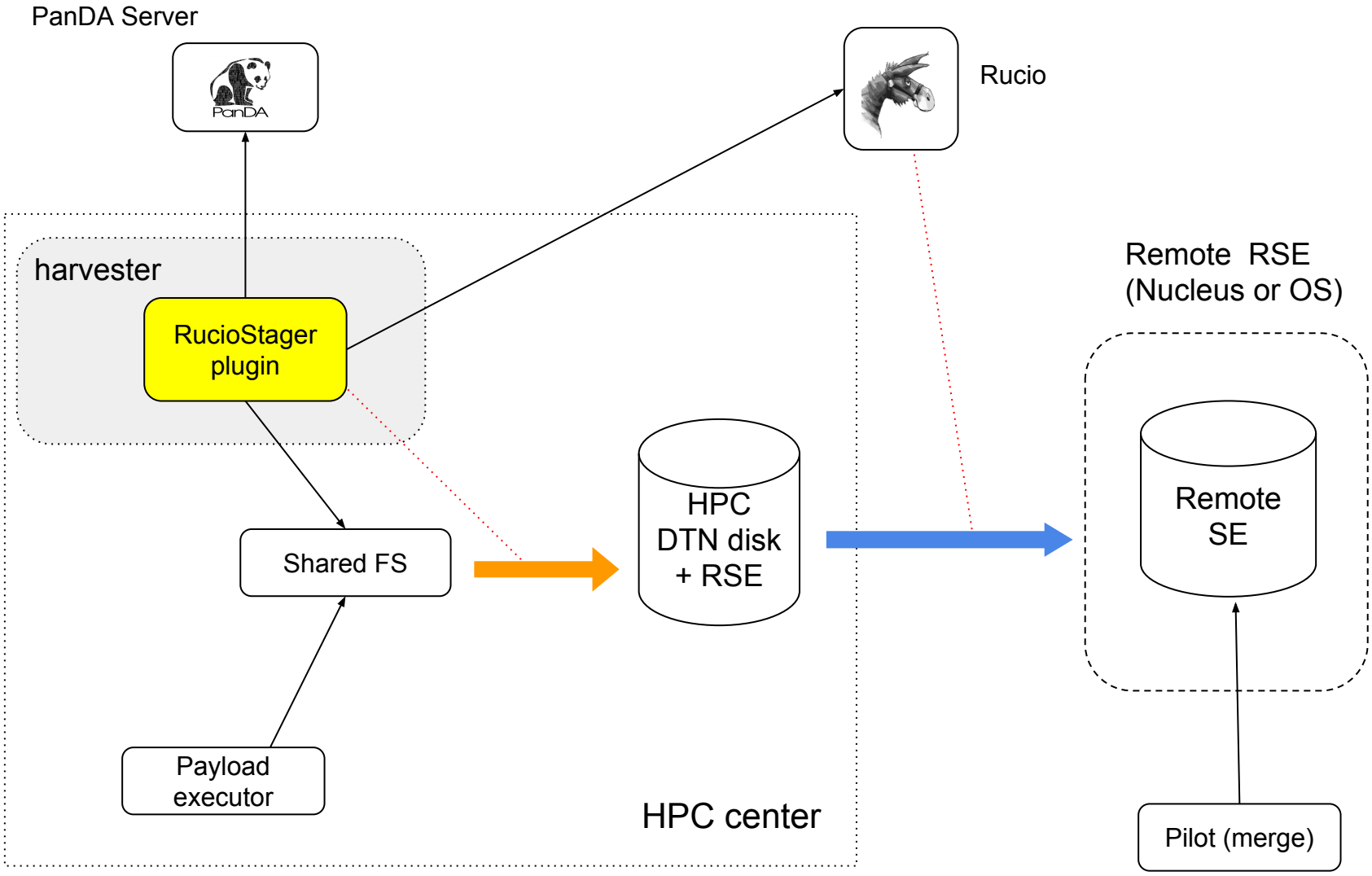


- Central + local DBs for scalability
- One core component (agent) per action
  - Action : job fetching, pre-stating, worker submission, …
- Multiple agent threads for each action
- Taking actions based on transition of job/worker state in local DB
- No direct communication (messaging) between agents
- Plugins developed by each resource expert

Example of Plug-ins (HPC with Globus Online)



Another Example with RucioStager for AES File Stage-out at HPC



Highlights of technical meeting 1/4

- https://indico.cern.ch/event/604083/
- ~20 participants
- Purely technical brainstorming
- Topics
  - General
    - Priorities of resources
      - Grid and HPC : high. Cloud : medium
    - Sub managers
      - Grid : AleDS, HPC : Danila/Taylor, Cloud: Will?/Fernando
    - Manpower
      - HPC: Danila, Taylor, Doug, David
      - Grid: David, Will, John, AleDS, Wen (AES)
      - Cloud: Will?, Fernando, ...

Highlights of technical meeting 2/4

- HPC
  - Current status
    - Prototype is fully functional
    - Tested multi-node AES/Yoda jobs at NERSC
  - Tasks
    - Generalization of access to HPC batch systems using SAGA
    - To identify pilot functions which can be used by harvester or payload
    - To make a common data management framework at HPC centers
      - With 3rd party service like Globus Online
      - With Rucio mover

Highlights of technical meeting 3/4

- Tasks (cntd)
  - ➢ Deployment and commissioning
    - ➢ NERSC, OLCF, ALCF, ASGC
  - ➢ Tests to be done at NERSC
    - ➢ Scalability tests, jumbo jobs, data transfers

– Grid/Core

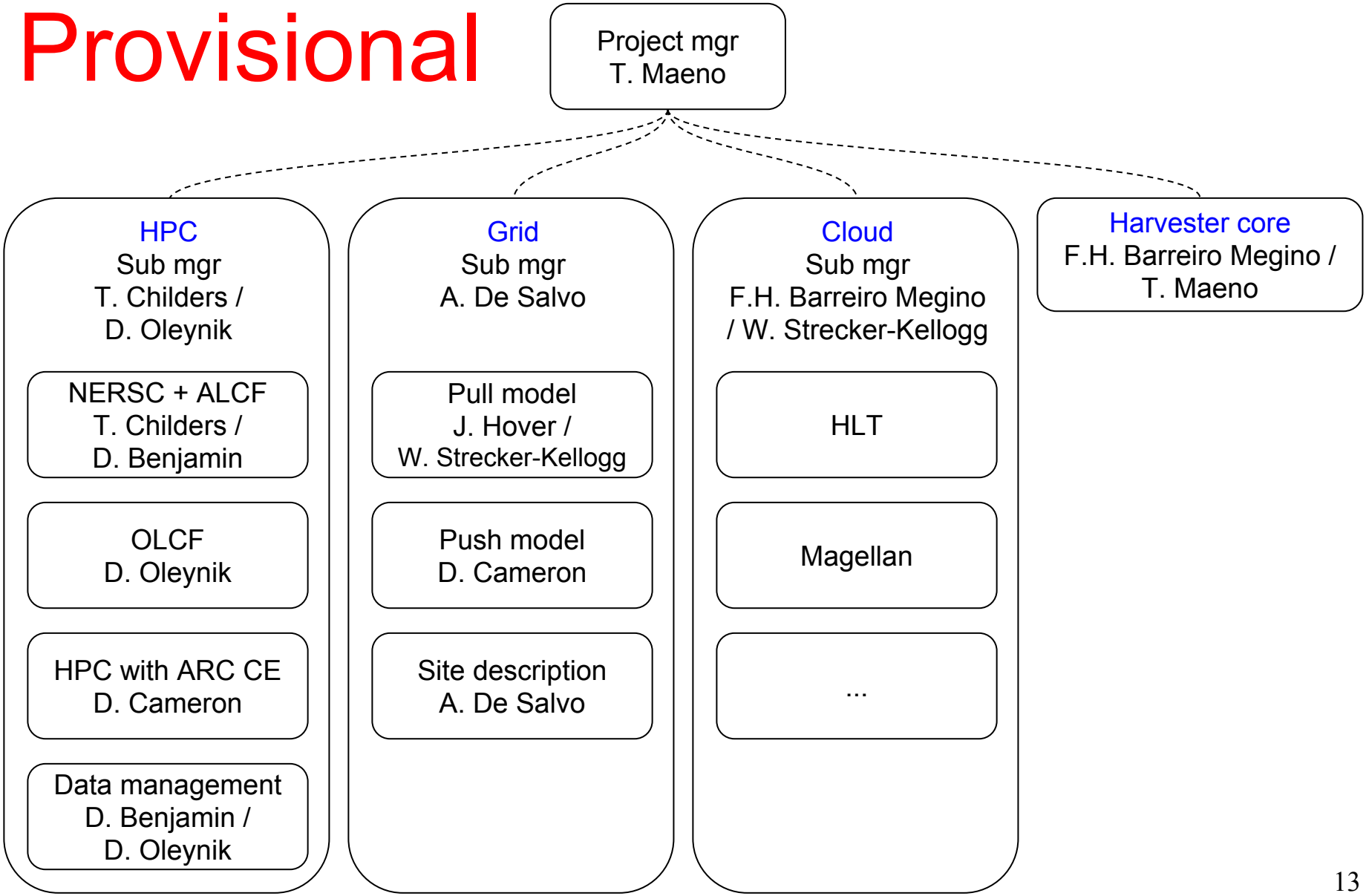
- Current status
  - ➢ Some core developments are required for the grid since the prototype implemented only a minimum function set for HPC
- Tasks
  - ➢ Site description
  - ➢ Centralization of instance configuration

Highlights of technical meeting 4/4

- Tasks (cntd)
  - Assessment of APF and aCT
    - To write architectural description and identify current issues
    - To fit harvester workflow
    - To develop plugins
  - Propagation of local resource information to central database
  - Harvester instance monitoring
  - Definition and implementation of communication protocol between panda and harvester
  - Evaluation of whole-node pilots, pilot stream control and push model for dynamic resource partitioning

Project Organization

Provisional



Contributing to the project

- Git repository
  - https://github.com/PanDAWMS/panda-harvester
  - Going to establish procedures for pull-requests, code reviews, and release
- Meeting
  - Harvester slot in every WFMS meeting (Thurs 4pm)
- Visualization of project status tracking

Summary

- Development project has been launched
  - Tasks have been identified and assigned to developers provisionally
    - More discussion and consultation needed for finalization
  - Priority of each task and time estimation will be defined in next WFSM meetings
- For HPC, commissioning at NERSC and OLCF is urgent and ALCF will follow
- For the grid, some core function to be developed. Assessment of APF and aCT is crucial
- For cloud, new manpower should be confirmed and increased