Resource Usage Estimation and Performance Prediction of Scientific Workflow Applications

Ewa Deelman

USC Information Sciences Institute

http://pegasus.isi.edu

USC Viterbi School of Engineering
Information Sciences Institute

Funded by the Department of Energy

Goal: "make it easier for scientists to execute large-scale computational tasks that use the power of computing resources they do not own to process data they did not collect with applications they did not develop"

Challenges: Little know about the application, dynamic, heterogeneous computing environment

Approach:

- Estimate the application resource needs
- Allocate the needed resources
- Model the performance of the application on the allocated resource
- Manage applications and resources during run
- Compare the actual behavior to the predicted behavior
- Discover anomalies and diagnose them
- Adapt application, resources

dV/dt: Accelerating the Rate of Progress towards Extreme Scale Collaborative Science (2012- ..)

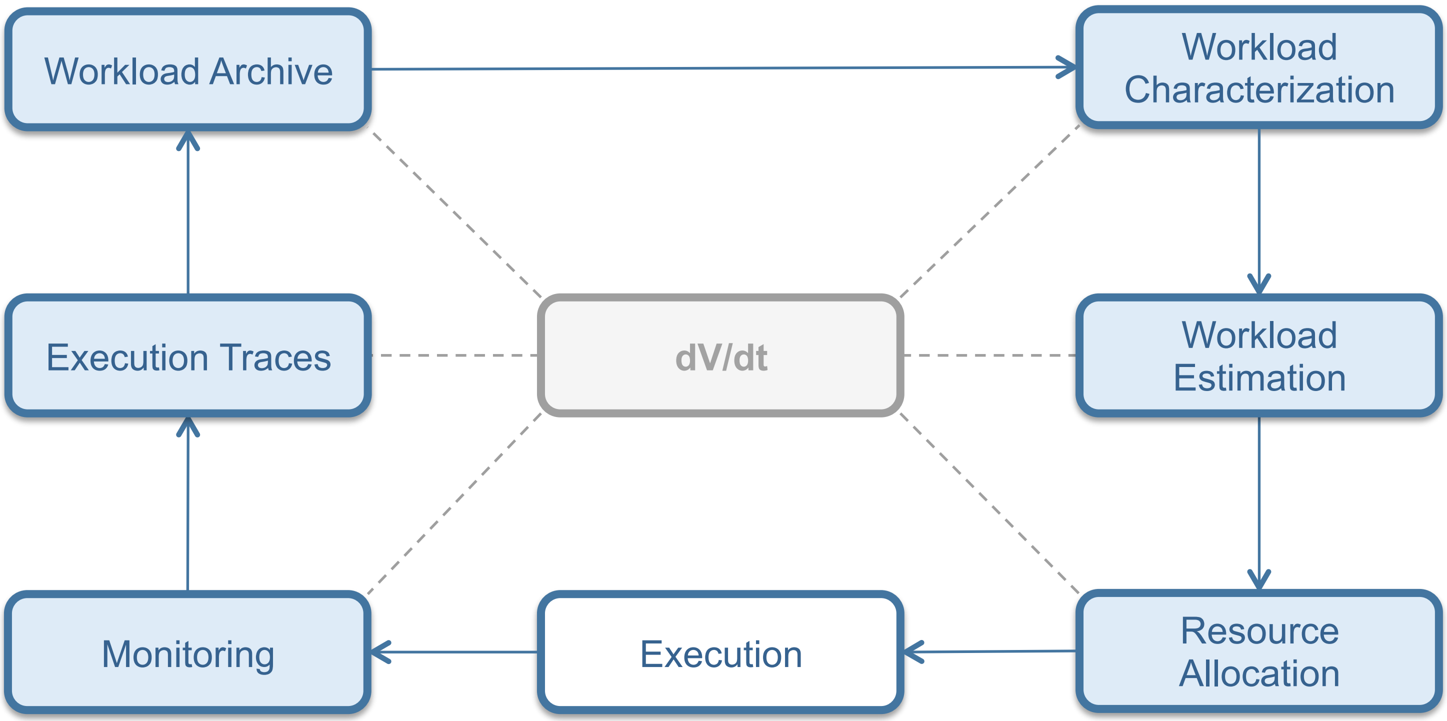
Miron Livny, Greg Thain (UWM), Bill Allcock (ANL), Douglas Thain, Ben Tovar (UND), Frank Wuerthwein, James Letts (UCSD), Ewa Deelman, Gideon Juve, Rafael Ferreira da Silva (USC)

Estimate the application resource needs
Allocate the needed resources
Model the performance of the application on the allocated resource
Manage applications and resources during run
Discover anomalies and diagnose them
Compare the actual behavior to the predicted behavior
Adapt application, resources – re-provision

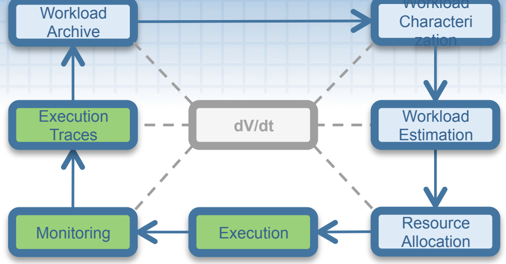
Experimental Foundation

- Real-world applications
  - Sets of tasks and workflows managed by HTCondor, HPC schedulers, workflow management systems (Makeflow, Pegasus)
- State of the art computing capabilities
  - Argonne Leadership Computing Facility and Open Science Grid
  - Campus resources at ND, UCSD and UW
  - Commercial cloud services
- Experimentation from the point of view of a scientist: "submit locally and compute globally"
- Pay attention to the cost involved in acquiring the resources and the human effort involved in software and data deployment and application management
  - Automate as much as possible

Approach


Workload Archive
Workload Characterization
Execution Traces
dV/dt
Workload Estimation
Monitoring
Execution
Resource Allocation

HTC Monitoring



- Job wrappers that collect information about processes
  - Runtime, peak disk usage, peak memory usage, CPU usage, etc.
- Mechanisms
  - Polling (not accurate, low overhead)
  - ptrace() system call interposition (accurate, high overhead)
  - LD_PRELOAD library call interposition (accurate, low overhead)
- Kickstart (Pegasus) and resource-monitor (Makeflow) also HTCondor logs

Error (Accuracy)

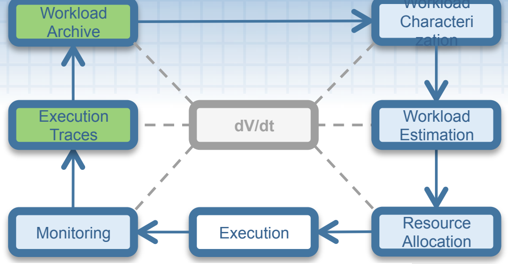
	Polling	LD_PRELOAD	Ptrace (syscalls)
CPU	0.5% - 12%	0.5% - 5%	< 0.2%
Memory	2% - 14%	< 0.1%	~ 0%
I/O	2% - 20%	0%	0%

Overhead

	Polling	LD_PRELOAD	Ptrace (syscalls)
CPU	low	low	low
Memory	low	medium	medium
I/O	low	low	high

Gideon Juve, et al., Practical Resource Monitoring for Robust High Throughput Computing, USC, Technical Report 14-950, 2014.

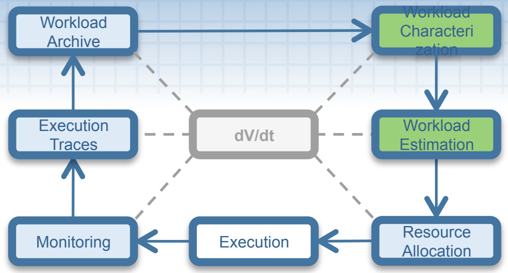
Workload Archive



- The workload summary archive captures the information gathered by our monitoring tools
- The archive is publicly readable at http://dvdt.crc.nd.edu .
  - Drupal and custom PHP and python code
  - Database backend running MySQL.
- Users of the archive can submit sets of resources summaries through a web interface, or with a batch job using ssh keys for authentication
- The archive can be queried to produce task summaries that match conditions, such as task name, monitoring tool used, and resource values comparisons

command	start	end	wall time (s)	cpu time (s)	concurrent processes	virtual memory (MB)	resident memory (MB)	swap memory (MB)	bytes read	bytes written	files	footprint (MB)
./distributed.script 0	2013-06-28 01:42:34	2013-06-28 02:26:52	2658.065628	2647.76	3	5075	2424	0	5015945881	835584	53	8549
./distributed.script 1	2013-06-28 01:01:54	2013-06-28 02:05:42	3827.227723	3825.77	3	5070	2418	0	10010974054	700416	53	8549

Workload Characteristics using HTCondor Logs



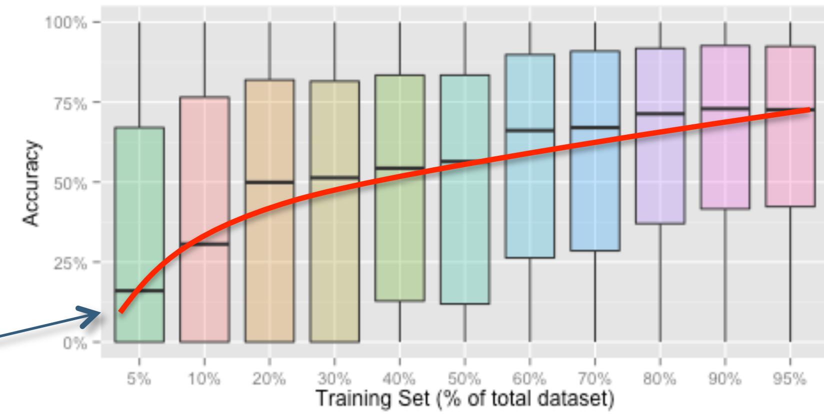
Characteristics of the CMS workload for a period of a month (Aug 2014)

Characteristic	Data
General Workload	
Total number of jobs	1,435,280
Total number of users	392
Total number of execution sites	75
Total number of execution nodes	15,484
Jobs statistics	
Completed jobs	792,603
Preempted jobs	257,230
Exit code (!= 0)	385,447
Average job runtime (in seconds)	9,444.6
Standard deviation of job runtime (in seconds)	14,988.8
Average disk usage (in MB)	55.3
Standard deviation of disk usage (in MB)	219.1
Average memory usage (in MB)	217.1
Standard deviation of memory usage (in MB)	659.6

Job Estimation: Experimental Results

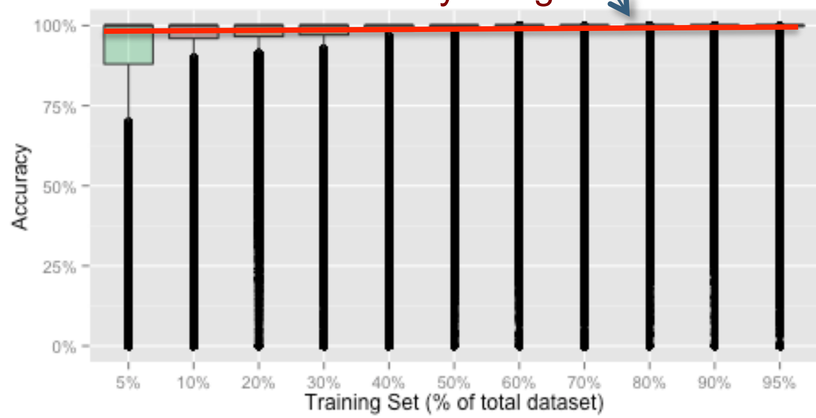
- Based on the regression trees
  - We built a regression tree per user
  - Estimates are generated according to a distribution (Normal or Gamma) or a uniform distribution

Job Runtime

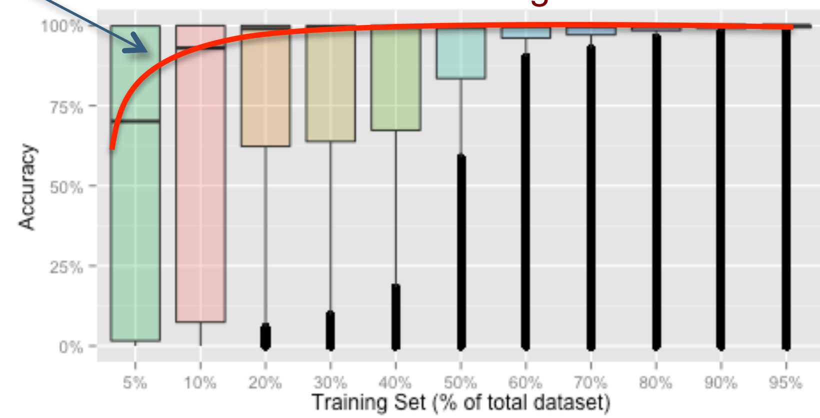


The median accuracy increases as more data is used for the training set

Memory Usage

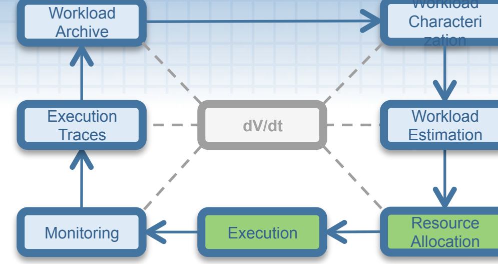


Disk Usage



Average accuracy of the workload dataset
The training set is defined as a portion of the entire workload dataset

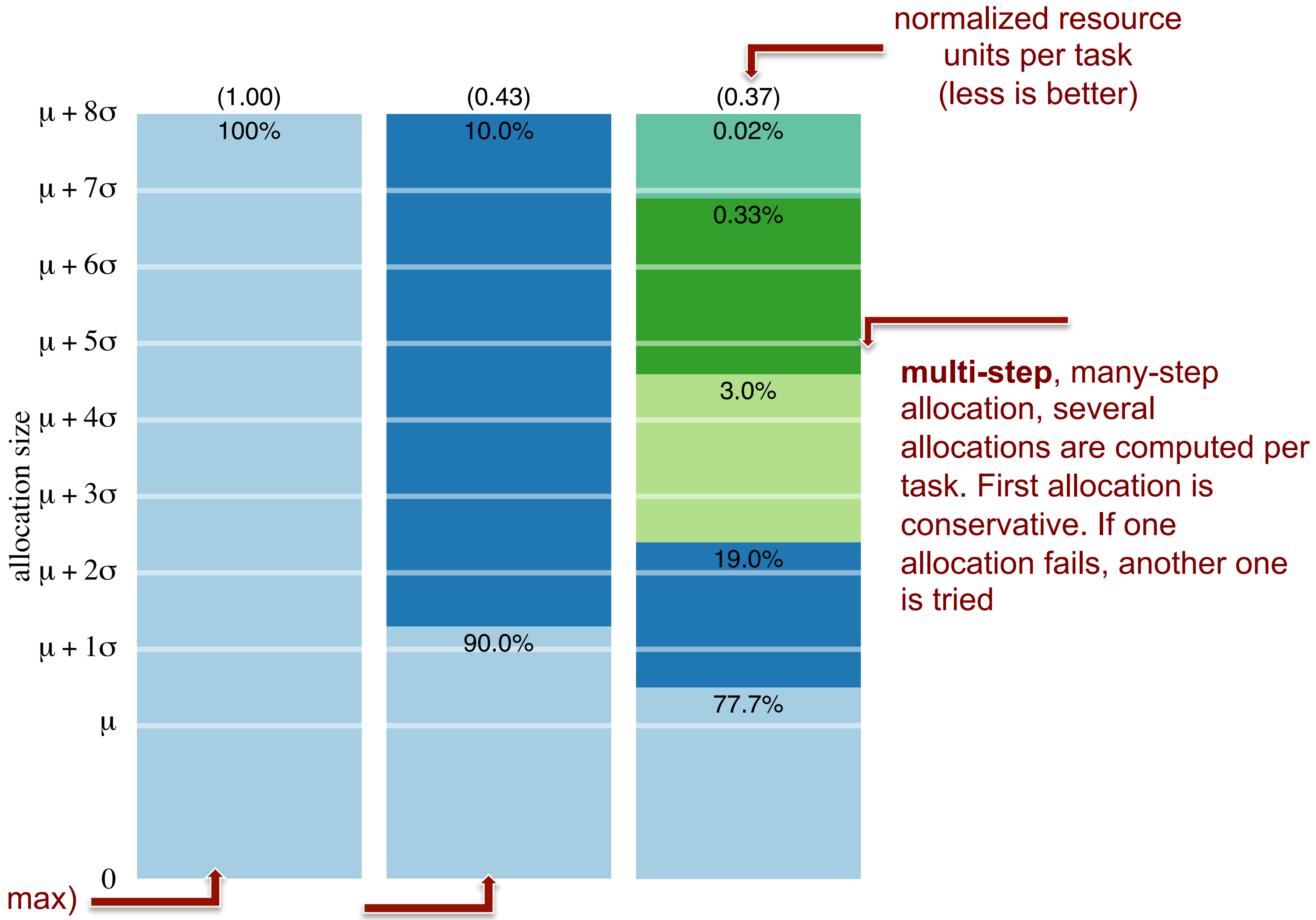
Resource Allocation



- Tasks have different sizes (known at runtime) while computation nodes have fixed sizes

- Resource allocation strategies
  - One task per node
    - Resources are underutilized
    - Throughput is reduced
  - Many tasks per node
    - Resources are exhausted
    - Jobs fail
    - Throughput is reduced

Example: One, Two and Multi-step allocations



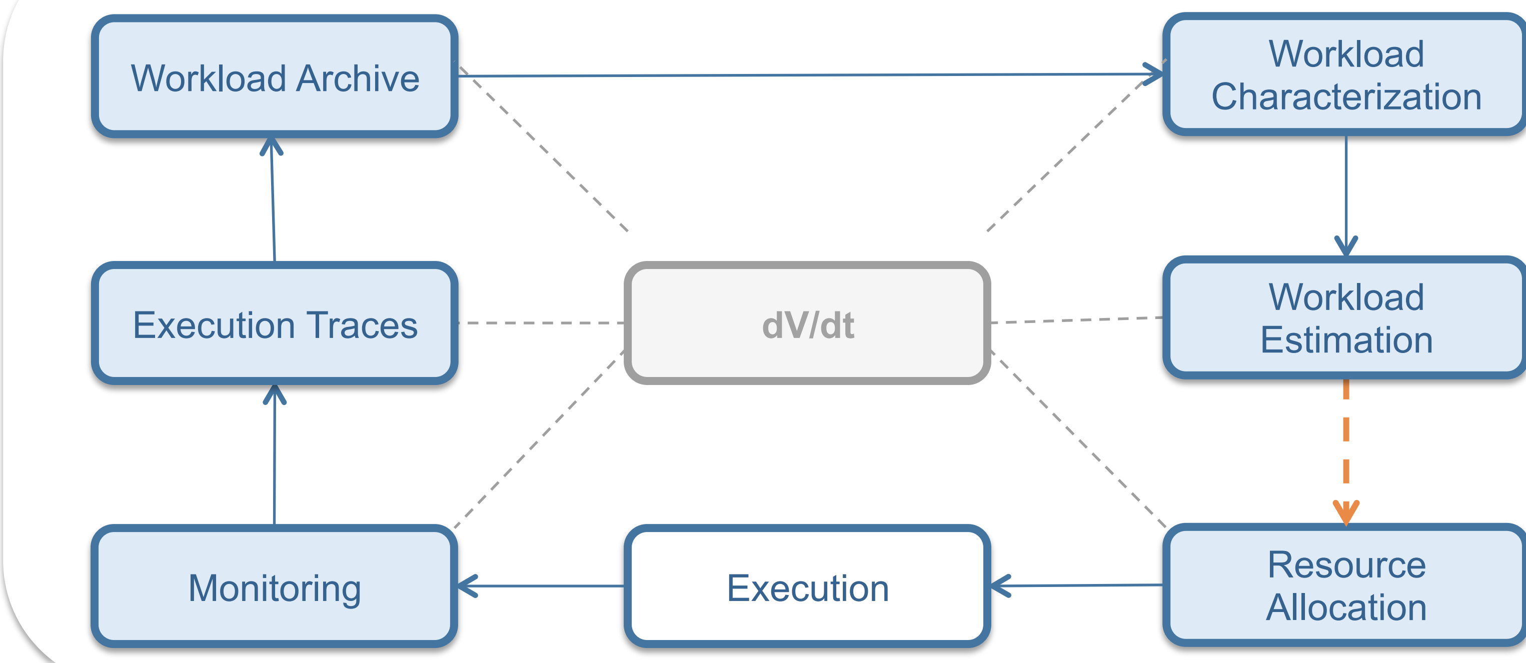
two-step, each task first runs with some computed allocation (aggressive). If the task fails because of resources exhaustion, it is rerun with the maximum allowed.

dV/dt Products

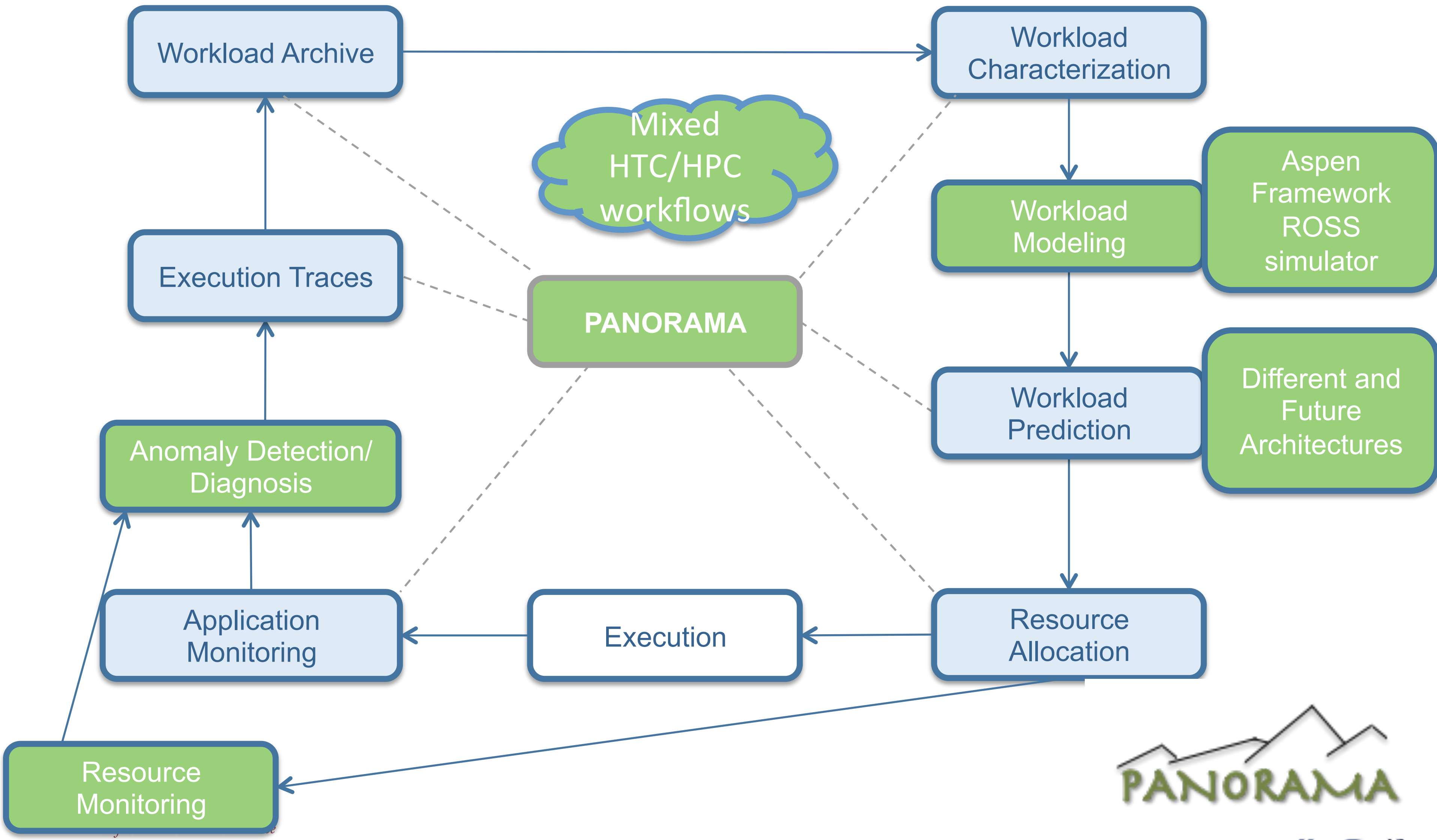
- **Monitoring tools:**
  - *kickstart* and *resource-monitor*, support different monitoring methods: ptrace system call interposition, library interposition, polling, support different levels of monitoring information, workflow system independent
- **Workflow archive:**
  - Sets of various types workflows with detailed performance information
  - Ongoing data collection effort
- **Methods:**
  - Online resource need estimation using regression trees and data clustering techniques
  - Dynamic resource allocation using runtime behavior information

Next Steps

- **Enhance monitoring and profiling**
  - Extend profiling to HPC applications
  - Investigate energy consumption
- **Close the loop**
  - Use resource predictions for provisioning and scheduling
  - Improve automation of entire loop
  - Conduct end-to-end experiments with real workloads
- **Productize tools**
  - Turn workload estimation software into a service

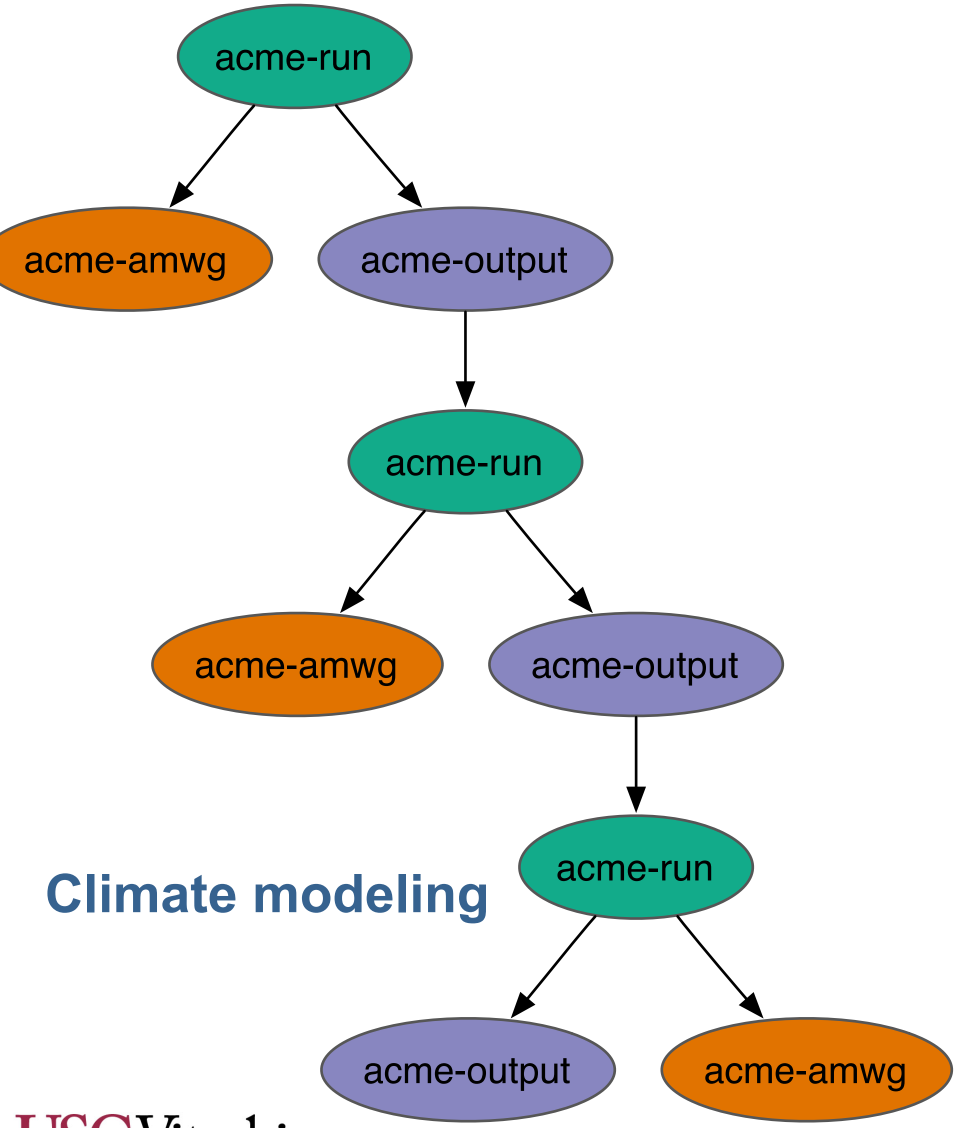


PANORAMA: Predictive Modeling and Diagnostic Monitoring of Extreme Science Workflows (est. 2014)

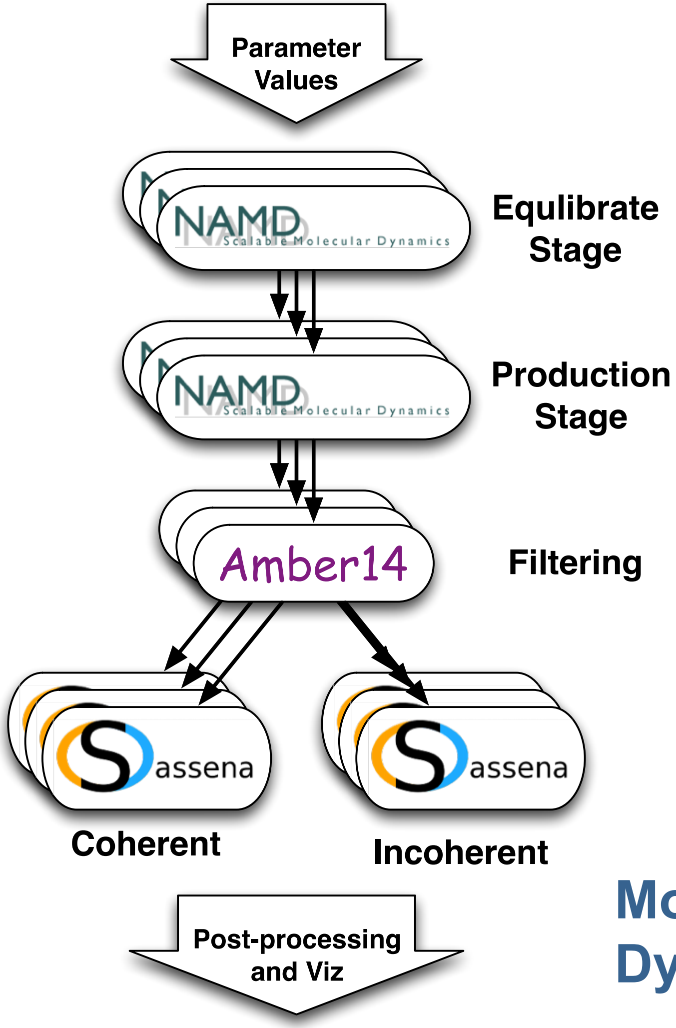


PANORAMA

PANORAMA Applications



Climate modeling



Molecular Dynamics

MANTID

PANORAMA

USC Viterbi
School of Engineering
Information Sciences Institute

PANORAMA next steps

- **Data Collection (Climate, SNS, synthetic workloads)**
- **Analytical Modeling with Aspen extending HPC modeling to wide area workflows**
- **Analytical Model refinement**
  - Integration of Aspen and Simulation
- **Automated Modeling**
  - Integration of Pegasus and Aspen (workflow + infrastructure -> resource needs, scheduling, predictions)
- **Correlation of application and infrastructure-level monitoring**
  - First step in anomaly detection

Participants:

- USC: Ewa Deelman, Gideon Juve, Dariusz Krol, Rafael Ferreira Da Silva,
- LBNL: Brian Tierney
- ORNL: Jeff Vetter, Vickie Lynch, Ben Mayer, Jeremy Meredith, Thomas Proffen
- RENCI: Anirban Mandal, Ilya Baldin, Paul Ruth
- RPI: Chris Carothers

https://sites.google.com/site/panoramaofworkflows/