Data Management Challenges of Large-Scale Data Intensive Scientific Workflows

Ewa Deelman

University of Southern California
Information Sciences Institute

Types of Workflow Applications

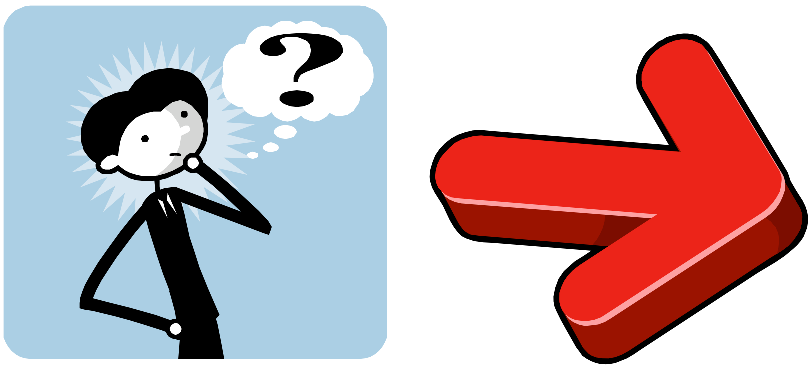
- **Providing a service to a community (Montage project)**
  - Data and derived data products available to a broad range of users
  - A limited number of small computational requests can be handled locally
  - For large numbers of requests or large requests need to rely on shared cyberinfrastructure resources
  - On-the fly workflow generation, portable workflow definition
- **Supporting community-based analysis (SCEC project)**
  - Codes are collaboratively developed
  - Codes are “strung” together to model complex systems
  - Ability to correctly connect components, scalability
- **Processing large amounts of shared data on shared resources (LIGO project)**
  - Data captured by various instruments and cataloged in community data registries.
  - Amounts of data necessitate reaching out beyond local clusters
  - Automation, scalability and reliability

Issues Critical to Scientists

- Reproducibility of scientific analyses and processes is at the core of the scientific method
  - Scientific versus Engineering reproducibility
  - Workflows give us the opportunity to provide reproducibility
- Scientists consider the "capture and generation of provenance information as a critical part of the workflow-generated data"
- "Sharing workflows is an essential element of education, and acceleration of knowledge dissemination."

NSF Workshop on the Challenges of Scientific Workflows, 2006, www.isi.edu/nsf-workflows06
Y. Gil, E. Deelman et al, Examining the Challenges of Scientific Workflows. IEEE Computer, 12/2007

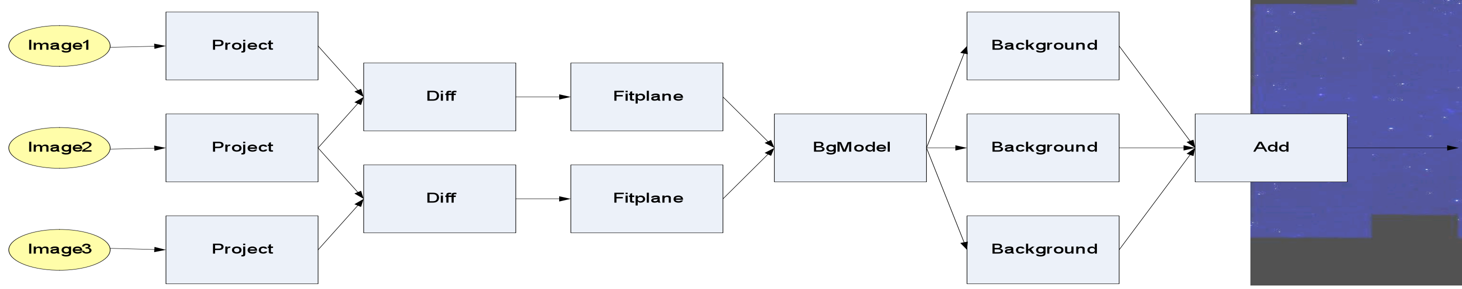
Science-grade Mosaic of the Sky

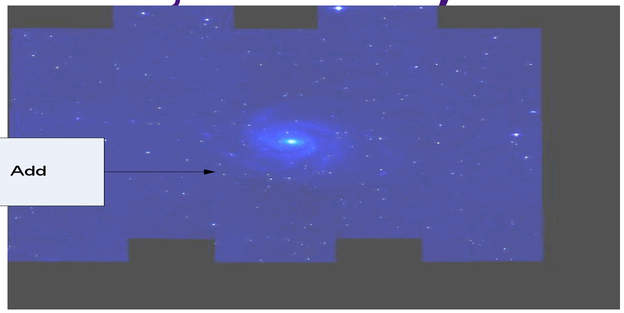


Point on the sky, area

Image Courtesy of IPAC, Caltech

Generating mosaics of the sky (Bruce Berriman, Caltech)

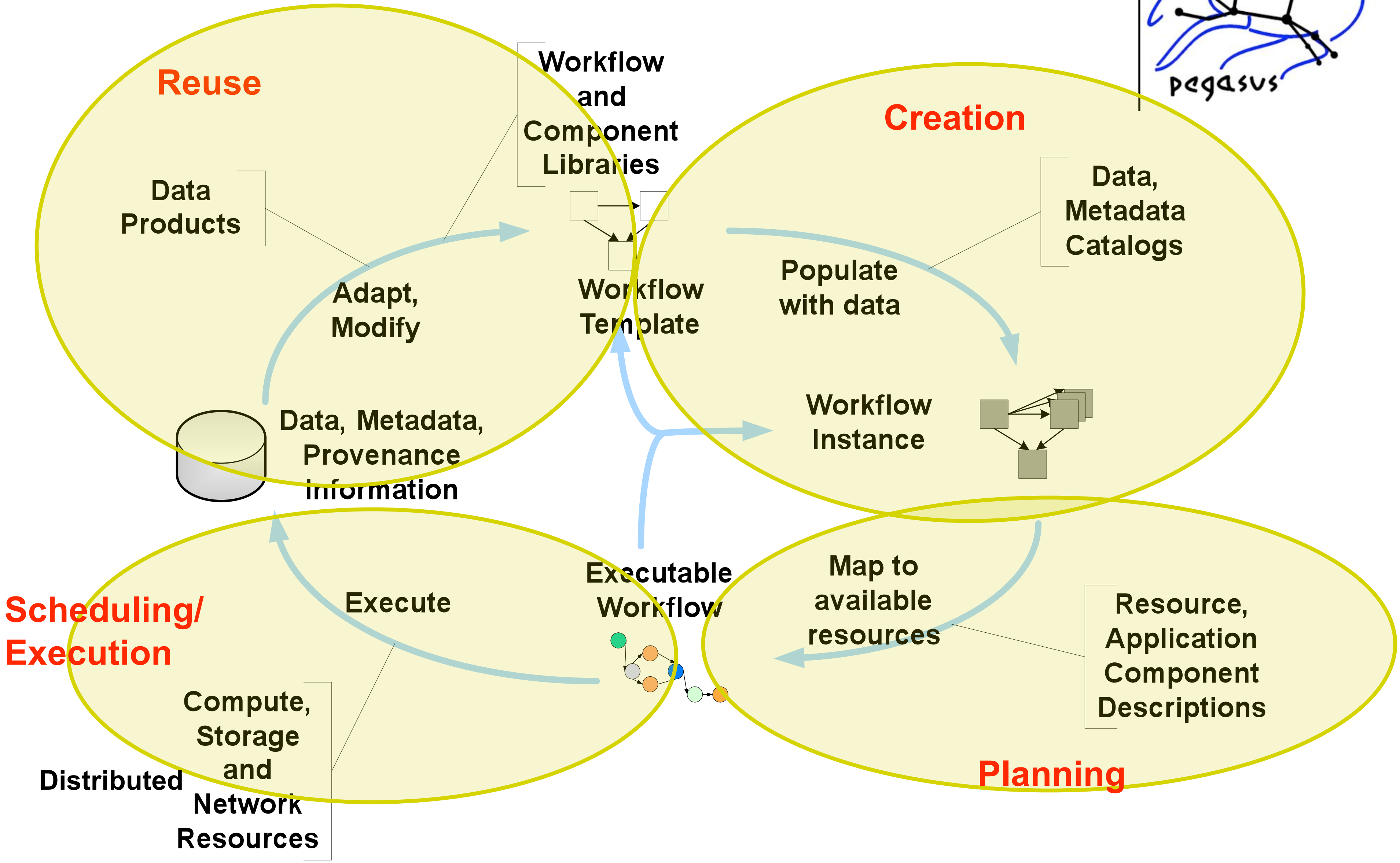




Size of the mosaic is degrees square*	Number of jobs	Number of input data files	Number of Intermediate files	Total data footprint	Approx. execution time (20 procs)
1	232	53	588	1.2GB	40 mins
2	1,444	212	3,906	5.5GB	49 mins
4	**4,856**	**747**	**13,061**	**20GB**	**1hr 46 mins**
6	8,586	1,444	22,850	38GB	2 hrs. 14 mins
10	20,652	3,722	54,434	97GB	6 hours

**The full moon is 0.5 deg. sq. when viewed form Earth, Full Sky is ~ 400,000 deg. sq.*

Workflow Lifecycle



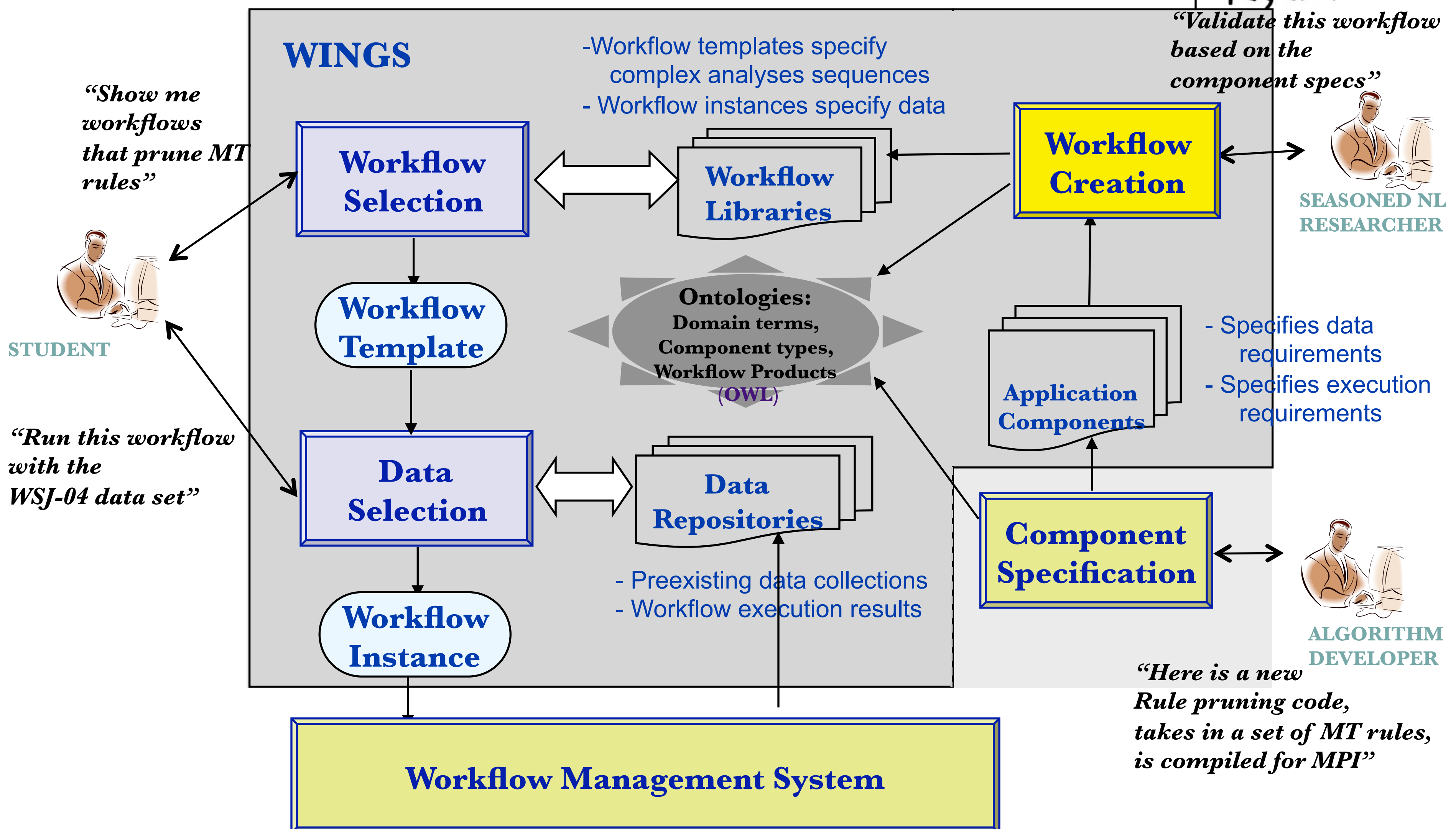
Workflow Creation

- Design a workflow (semantics info needed)
  - Find the right components
  - Set the right parameters
  - Find the right data
  - Connect appropriate pieces together
  - Find the right fillers
- Support both experts and novices

Challenges in user experiences

- Users' expectations vary greatly
  - High-level descriptions
  - Detailed plans that include specific resources
- Users interactions can be exploratory
    - Or workflows can be iterative
  - Modifying portions of the workflow as the computation progresses
- Users need progress, failure information at the right level of detail
- There is no ONE user but many users with different knowledge and capabilities

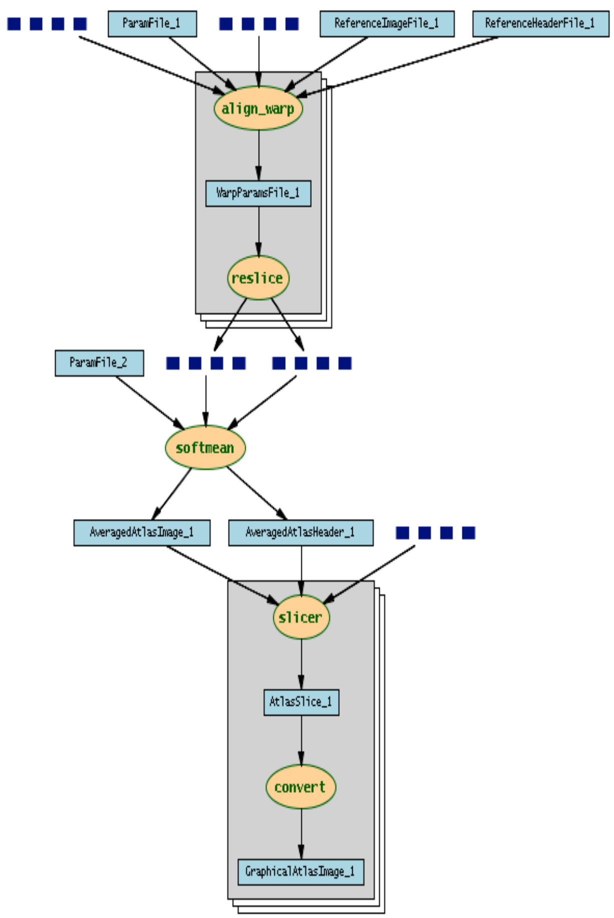
Wings: Workflow Instance Generation and Selection (Y. Gil, USC/ISI)



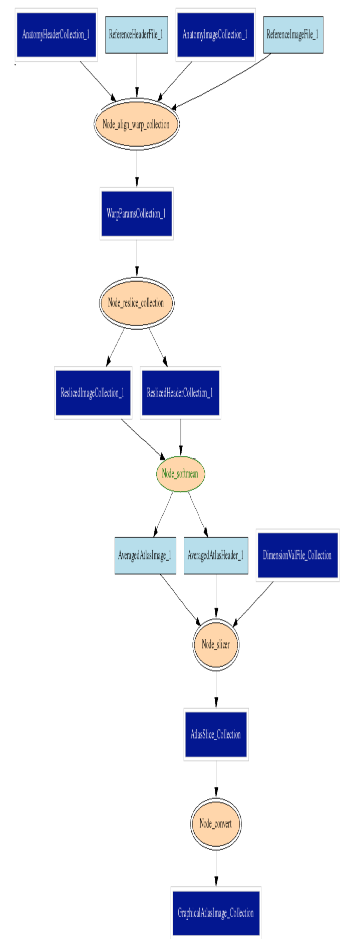
Editing and Creating Workflows

Wings Editor

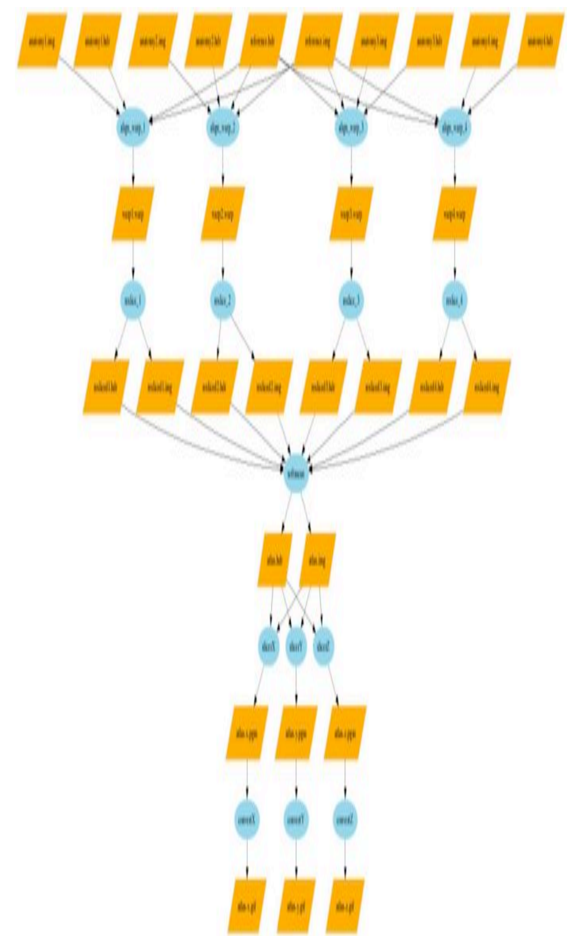
Users get feedback and suggestions



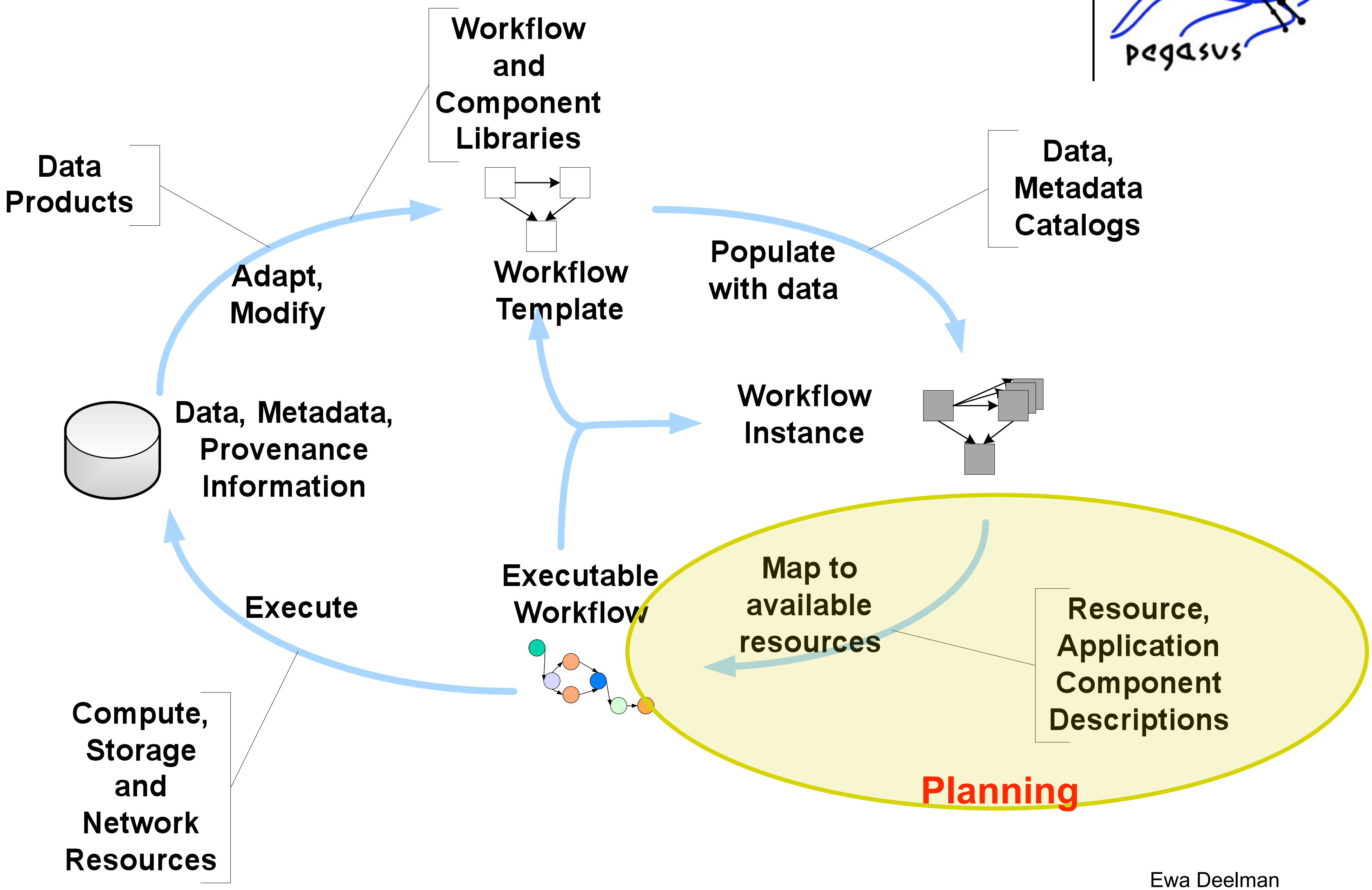
Workflow template



Workflow instance



Workflow Lifecycle



Ewa Deelman
www.isi.edu/~deelman

Specification: Place Y = F(x) at L
Execution Environment: Distributed

- Find where x is--- {S1,S2, …}
- Find where F can be computed--- {C1,C2, …}
- Choose *c* and *s* subject to constraints (performance, space availability,….)
- Move x from *s* to *c*
  - *Move F to c*
- Compute F(x) at *c*
- Move Y from *c* to L
- Register Y in data registry
- Record provenance of Y, performance of F(x) at c

Error! *x* was not at *s!*

Error! *F(x)* failed!

Error! *c* crashed!

Error! *there is not enough* space at L!

Some challenges in workflow mapping

- Automated management of data
- Efficient mapping the workflow instances to resources
  - Runtime Performance
  - Data space optimizations
  - Fault tolerance (involves interfacing with the workflow execution system)
    - Recovery by replanning
    - plan “B”
  - Scalability
- Providing feedback to the user
  - Feasibility, time estimates

Pegasus-Workflow Management System

- Leverages abstraction for workflow description to obtain ease of use, scalability, and portability
- Provides a compiler to map from high-level descriptions (workflow instances) to executable workflows
  - Correct mapping
  - Performance enhanced mapping
- Provides a runtime engine to carry out the instructions (Condor DAGMan)
  - Scalable manner
  - Reliable manner

In collaboration with Miron Livny, UW Madison, funded under NSF-OCI SDCI

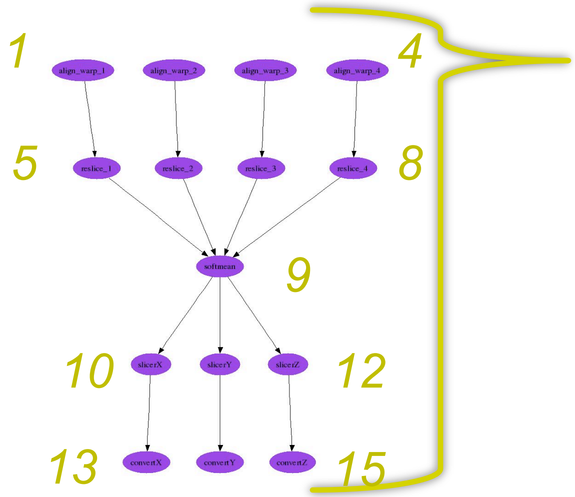
Mapping Correctly

- Select where to run the computations
  - Apply a scheduling algorithm
    - HEFT, min-min, round-robin, random
    - Schedule in a data-aware fashion (data transfers, amount of storage)
    - The quality of the scheduling depends on the quality of information
  - Transform task nodes into nodes with executable descriptions
    - Execution location
    - Environment variables initializes
    - Appropriate command-line parameters set
- Select which data to access
  - Add stage-in nodes to move data to computations
  - Add stage-out nodes to transfer data out of remote sites to storage
  - Add data transfer nodes between computation nodes that execute on different resources
- Add nodes to create an execution directory on a remote site

Additional Mapping Elements

- Cluster compute nodes in small granularity applications
- Add data cleanup nodes to remove data from remote sites when no longer needed
  - reduces workflow data footprint
- Add nodes that register the newly-created data products
- Provide provenance capture steps
  - Information about source of data, executables invoked, environment variables, parameters, machines used, performance
- Scale matters--today we can handle:
  - 1 million tasks in the workflow instance (SCEC)
  - 10TB input data (LIGO)

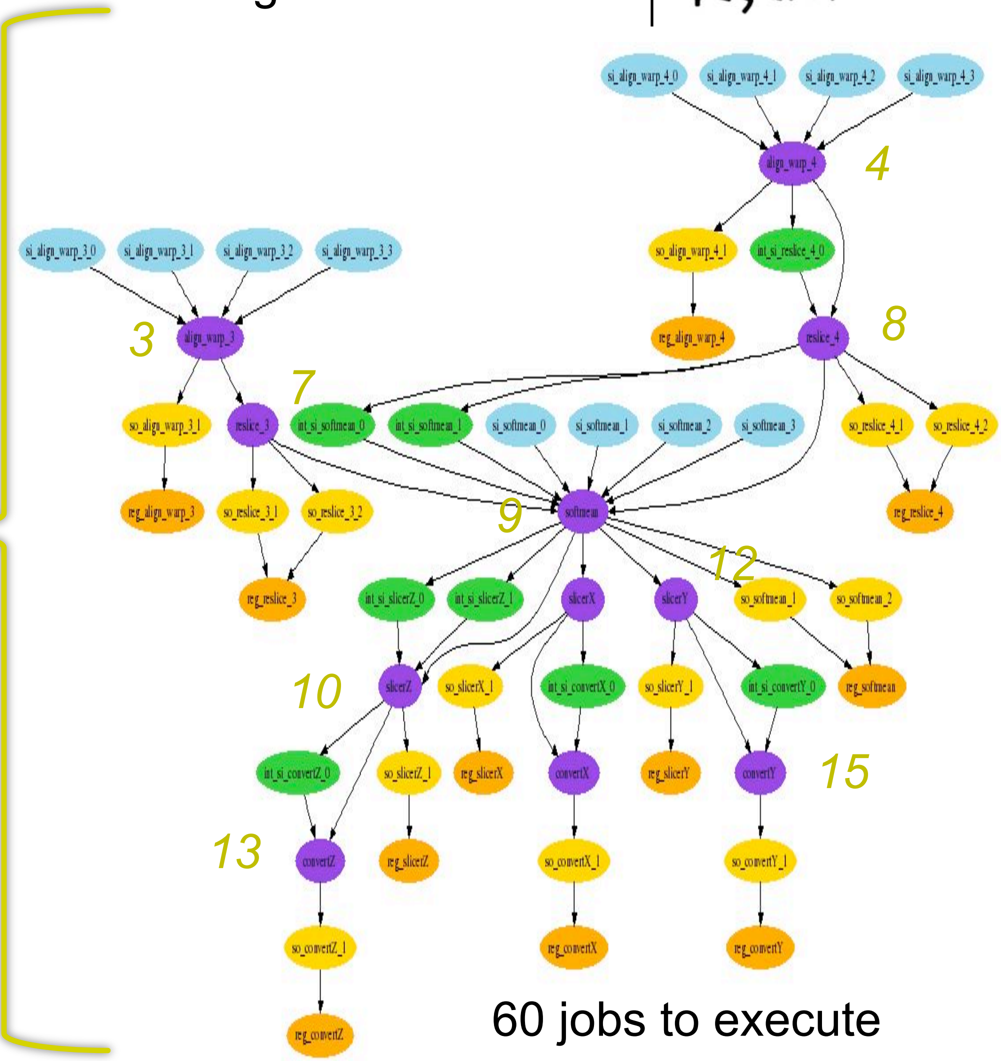
Pegasus Workflow Mapping



Original workflow: 15 compute nodes devoid of resource assignment

Resulting workflow mapped onto 3 Grid sites:

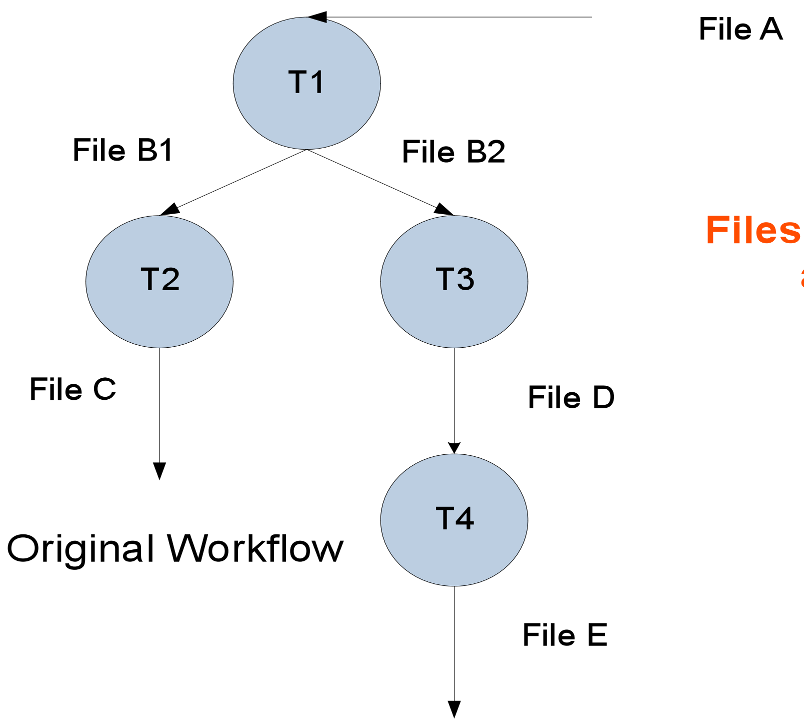
- 11 compute nodes (4 reduced based on available intermediate data)
- 12 data stage-in nodes
- 8 inter-site data transfers
- 14 data stage-out nodes to long-term storage
- 14 data registration nodes (data cataloging)



Data Reuse

Sometimes it is cheaper to access the data than to regenerate it

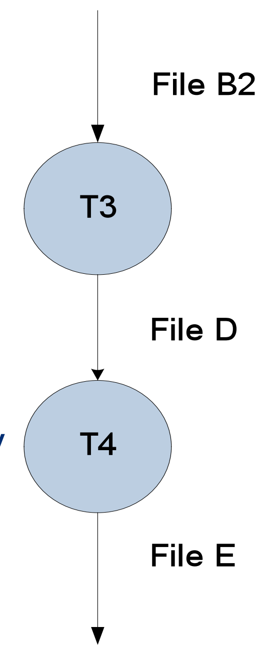
Keeping track of data as it is generated supports workflow-level checkpointing



Files C and B2 are available

Need to be careful how reuse is done

Reduced Workflow



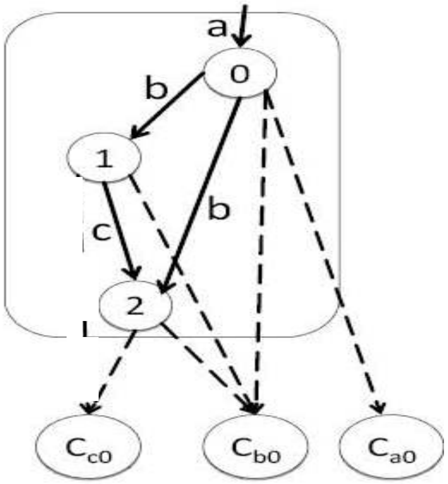
Mapping Complex Workflows Onto Grid Environments, E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, K. Backburn, A. Lazzarini, A. Arbee, R. Cavanaugh, S. Koranda, *Journal of Grid Computing, Vol.1, No. 1, 2003., pp25-39.*

Efficient data handling

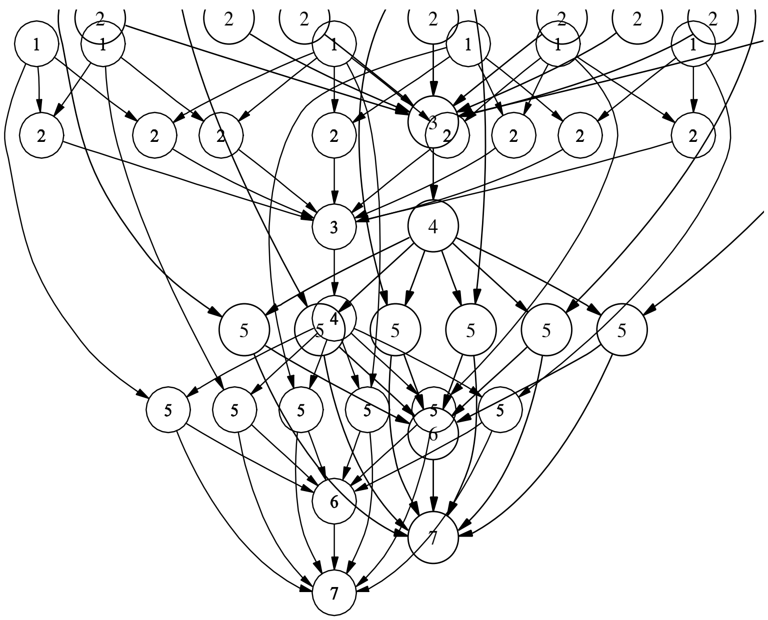
- Input data is staged dynamically
- New data products are generated during execution
- For large workflows 10,000+ files
  - Similar order of intermediate and output files
  - Total space occupied is far greater than available space—failures occur
- Solution:
  - Determine which data are no longer needed and when
  - Add nodes to the workflow do cleanup data along the way
- Issues:
  - minimize the number of nodes and dependencies added so as not to slow down workflow execution
  - deal with portions of workflows scheduled to multiple sites

Joint work with Rizos Sakellariou, Manchester University, CCGrid 2007, Scientific Programming Journal, 2007

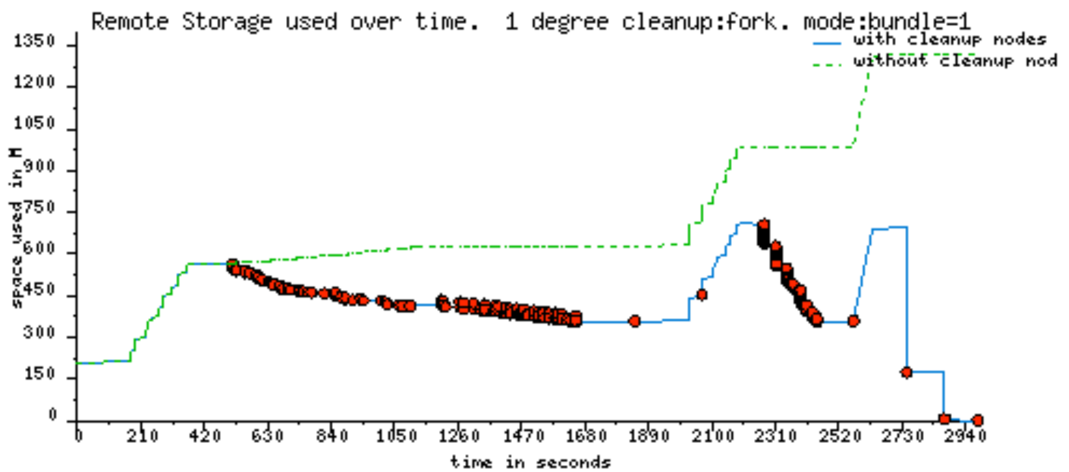
LIGO-a gravitational-wave physics application and Montage



Adding cleanup nodes to the workflow

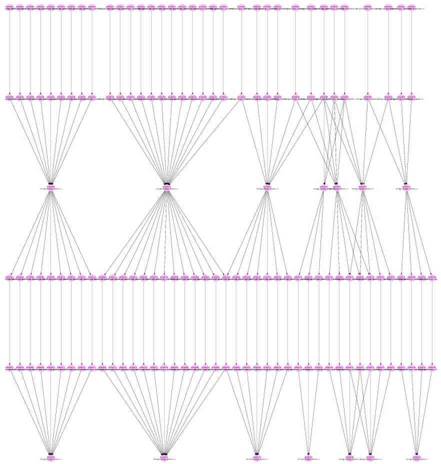


1.25GB versus 4.5 GB

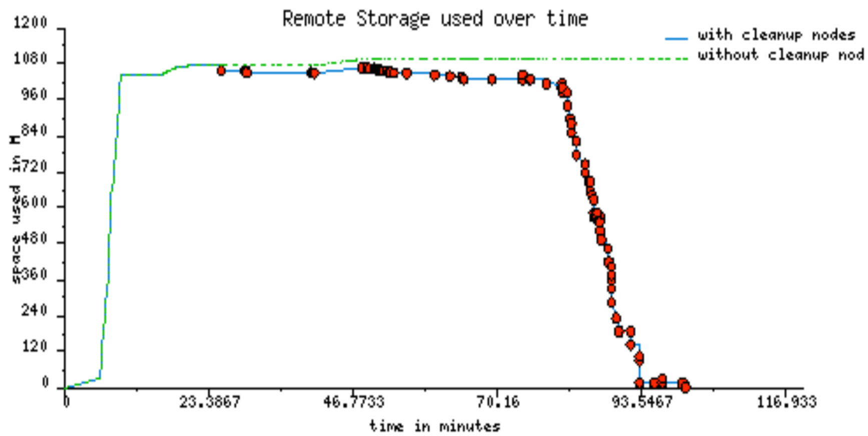


LIGO Scientists Don't like their Workflows Either

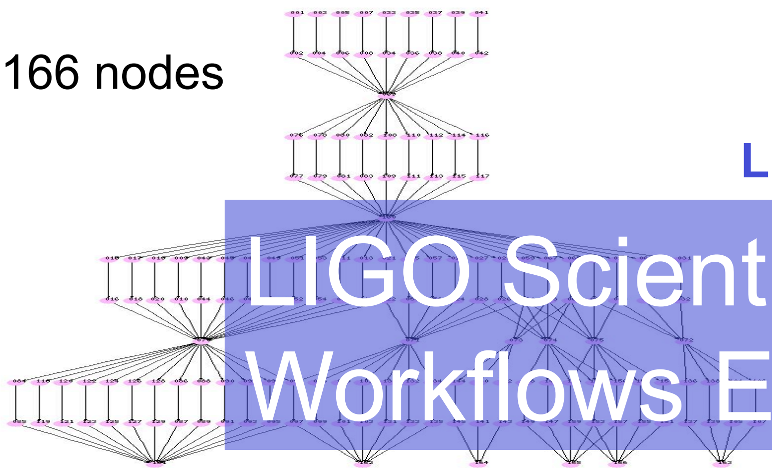
LIGO Workflows

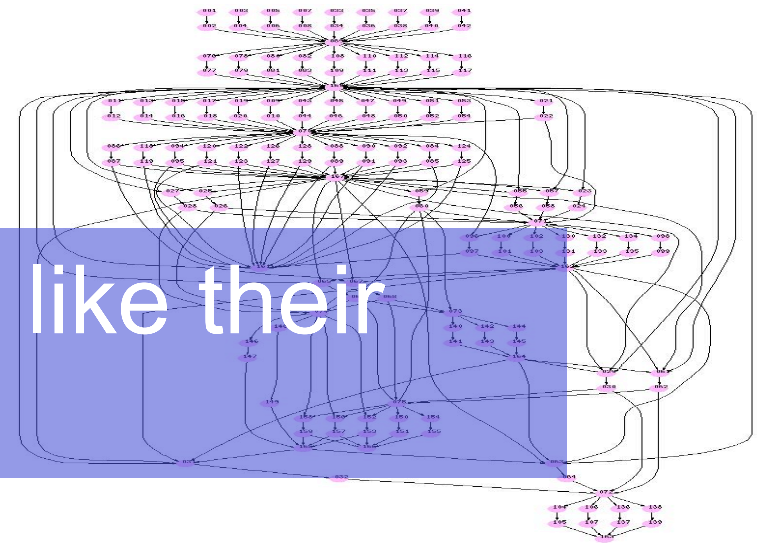


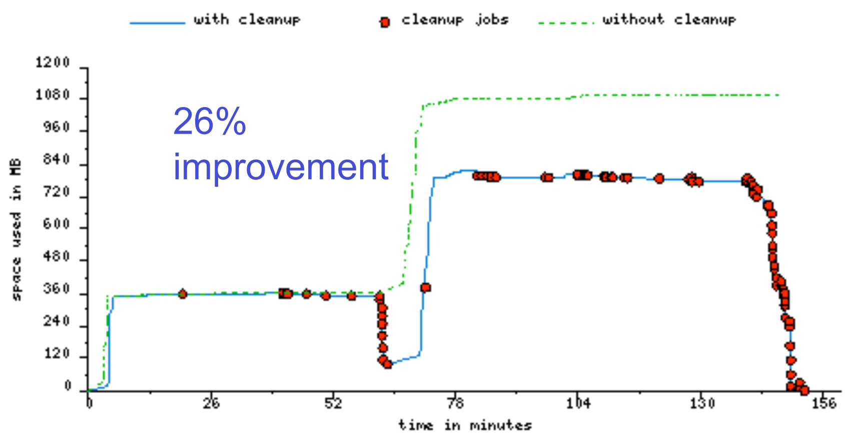
166 nodes



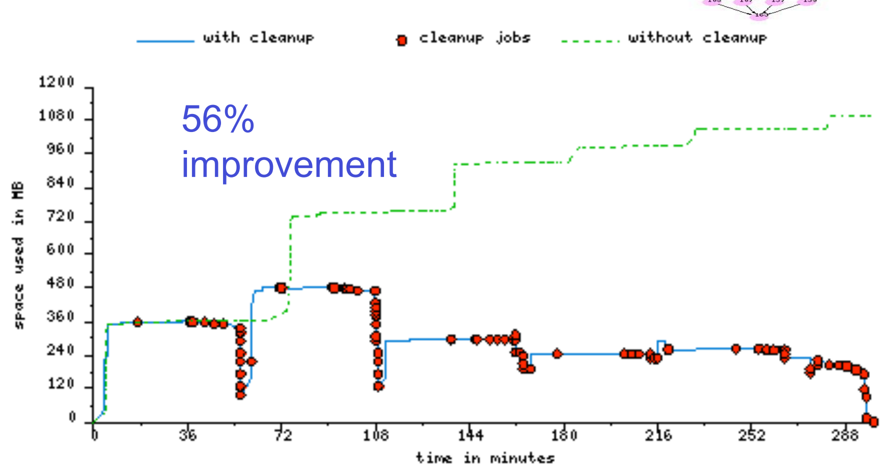
Full workflow:
185,000 nodes
466,000 edges
10 TB of input data
1 TB of output data.







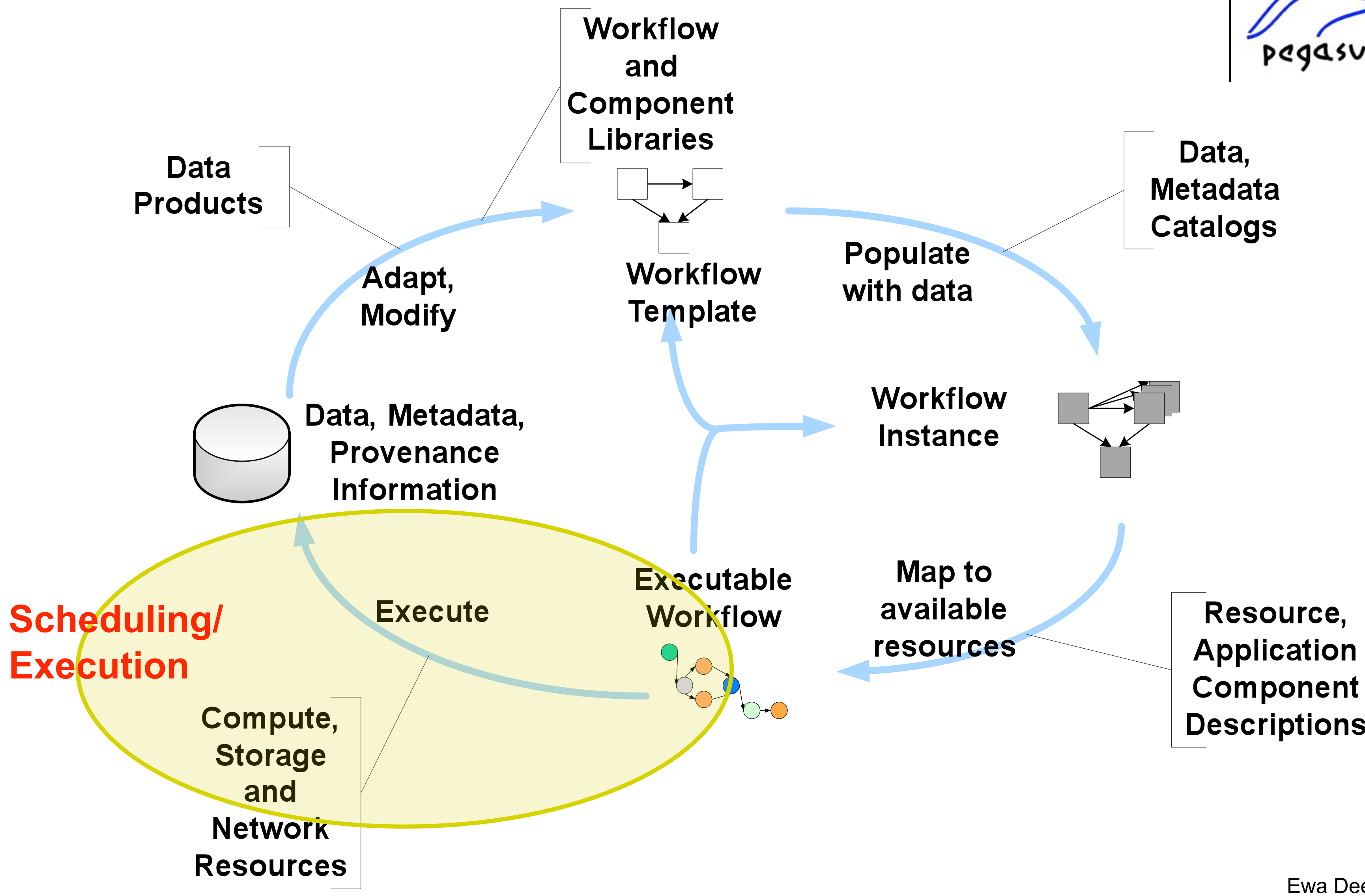
26% improvement



56% improvement

www

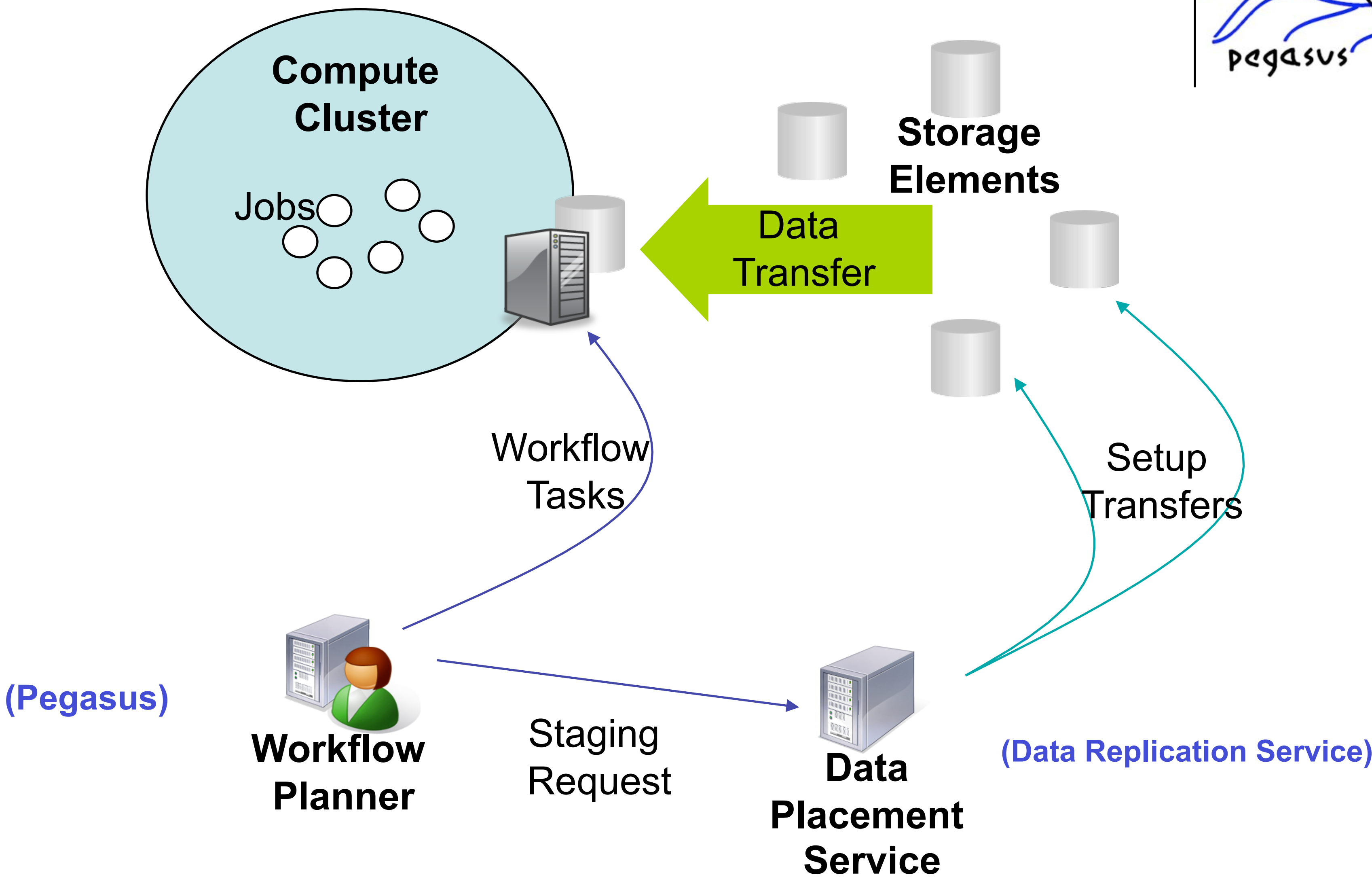
Workflow Lifecycle



Challenges in Workflow Execution

- Resource provisioning
  - Which resources to provision if many possibilities?
  - How many resources to provision?
  - For how long?
- Fault Tolerance
  - How to recognize different types of failures
  - How to recover from failures?
- Efficient collaboration between the data and computation management systems
- Debugging
  - How to relate the workflow result (outcome) to workflow specification

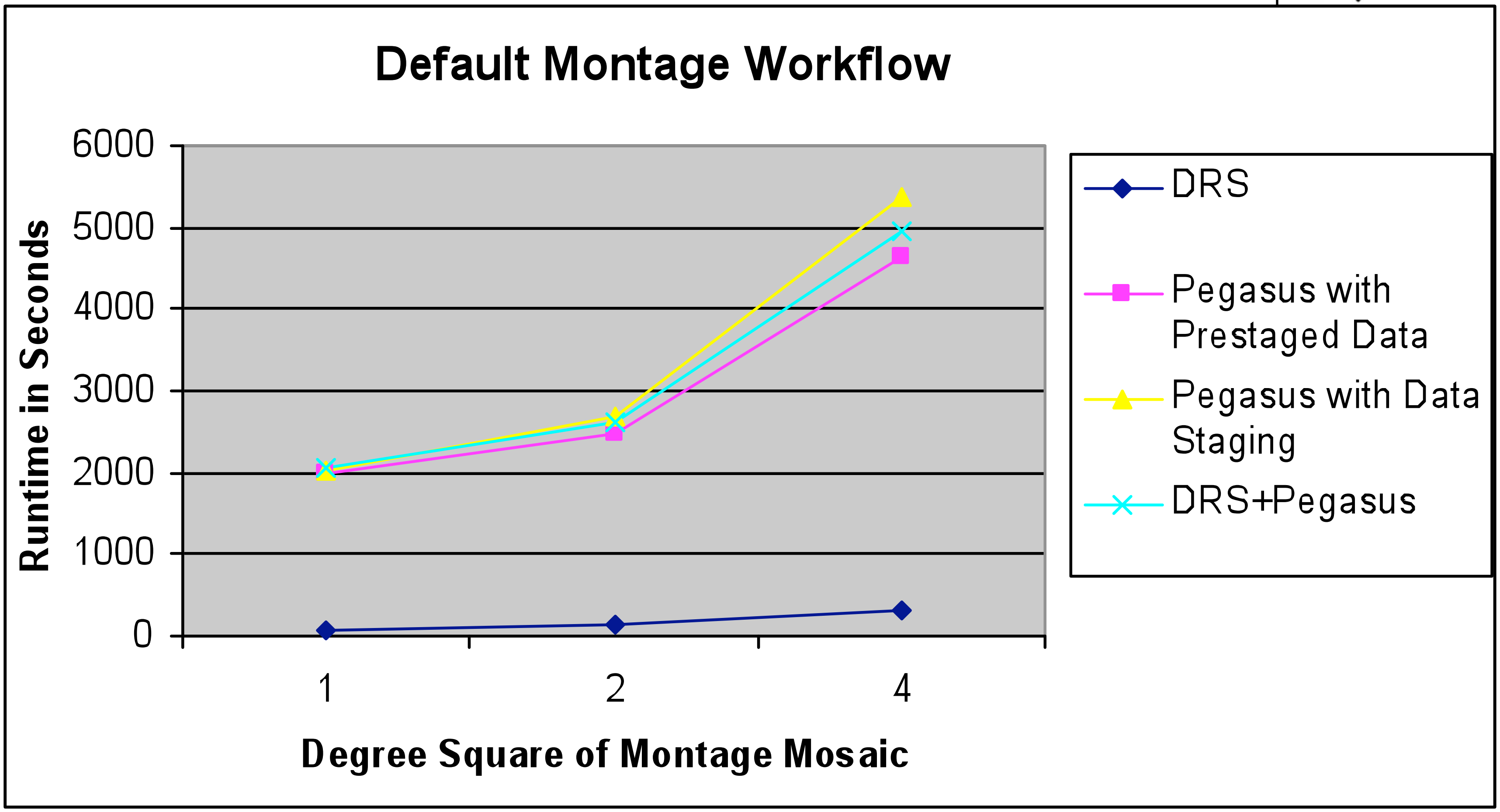
Interaction Between Workflow Planner and Data Placement Service for Staging Data



Joint work with Ann Chervenak, USC/ISI
Significant potential for multiple workflows

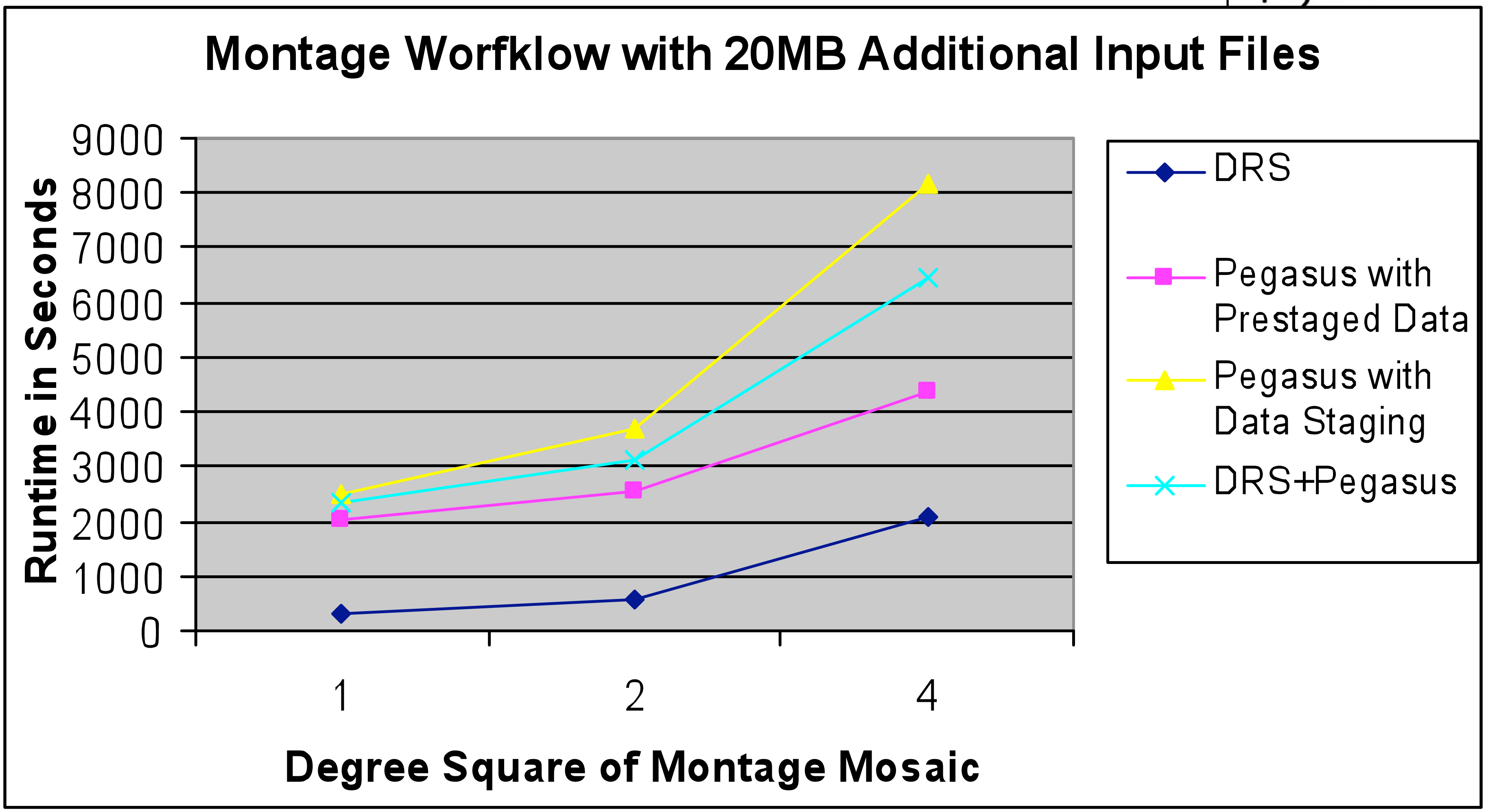
Grid 2007

Execution Times with Default Input Sizes



Combination of prestaging data with DRS followed by workflow execution using Pegasus Improves execution time approximately 21.4% over Pegasus performing explicit data staging

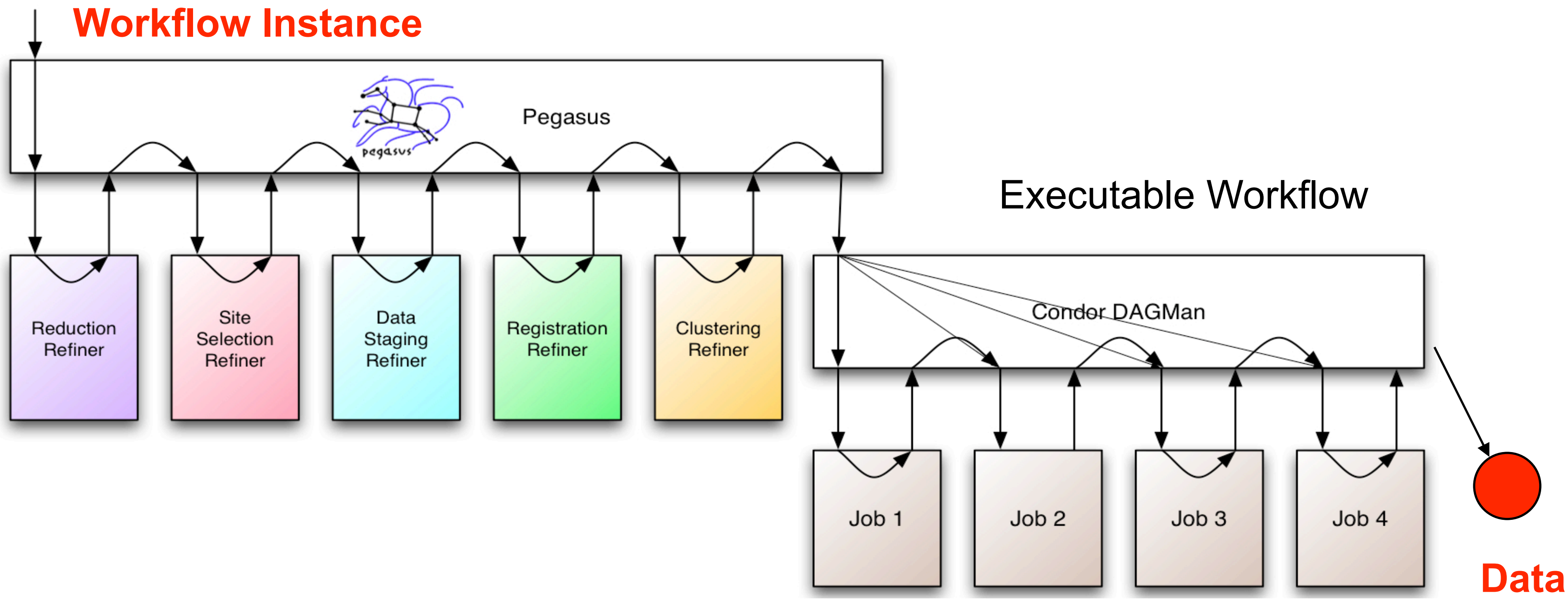
Execution Times with Additional 20 MB Input Files



With asynchronous data staging, execution time is reduced by over 46%

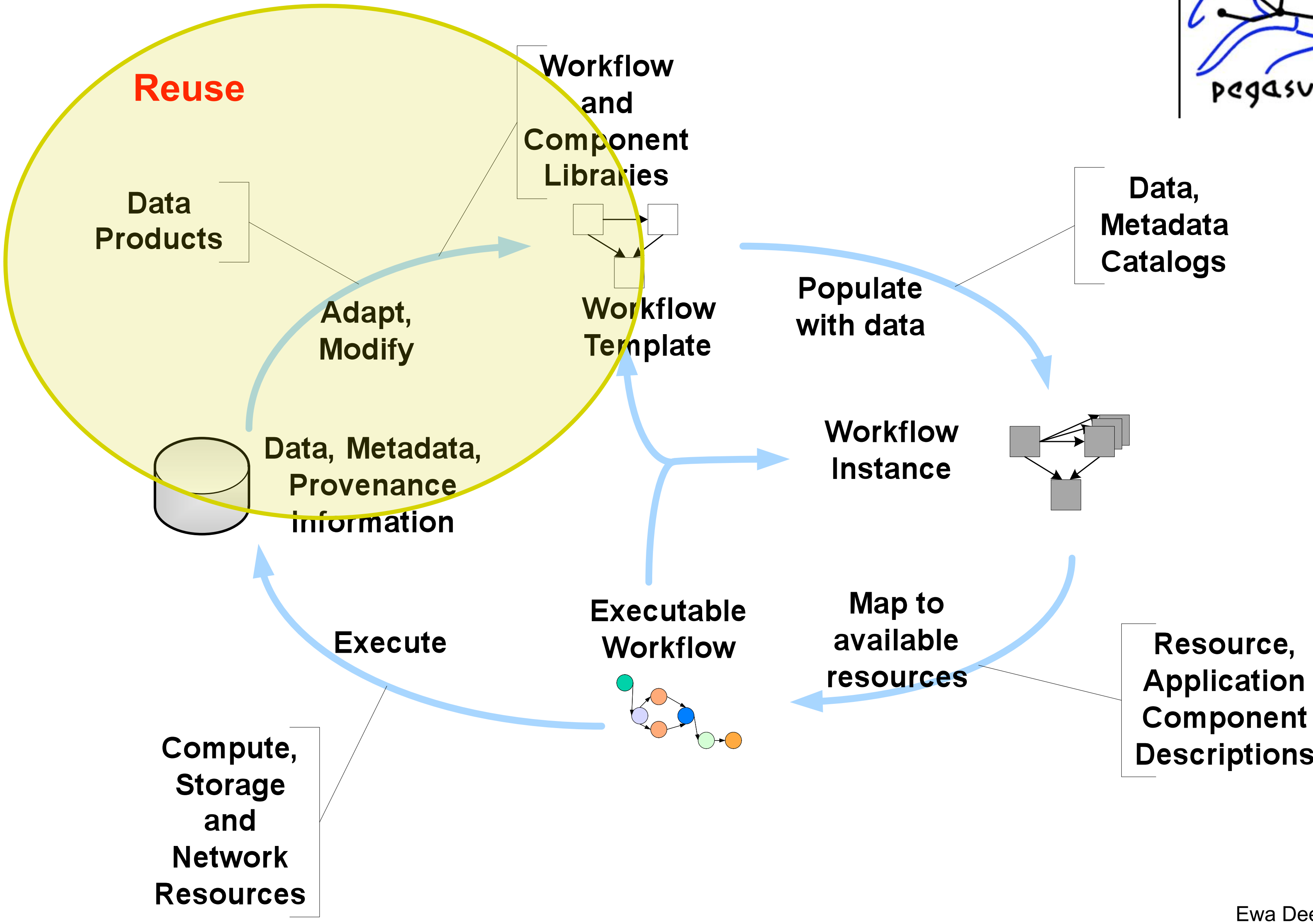
Workflow Mapping and Execution Connected

- For each data item, we can find the executable workflow steps that produced it and other data items that contributed to those steps.
- For each workflow step, we can find its connection to the workflow instance jobs from which it was refined.



Joint work with Luc Moreau, Southampton University e-Science 2007

Workflow Lifecycle



Challenges in reuse and sharing

- How to find what is already there
- How to determine the quality of what's there
- How to invoke an existing workflow
- How to share a workflow with a colleague
- How to share a workflow with a competitor

Sharing: the new frontier

- MyExperiment in the UK (University of Manchester), a repository of workflows http://www.myexperiment.org/

- How do you share workflows across different workflow systems?
  - How to write a workflow in Wings and execute in ASKALON?
  - NSF/Mellon Workshop on Scientific and Scholarly Workflow, 2007 https://spaces.internet2.edu/display/SciSchWorkflow/Home

- How do you interpret results from one workflow when you are using a different workflows system (provenance-level interoperability)
  - Provenance challenge http://twiki.ipaw.info/
  - Open provenance model http://eprints.ecs.soton.ac.uk/14979/1/opm.pdf

Conclusions

- Much work done to date in scientific workflows
- Scientists are buying into the new programming model
- Data handling is critical to the success of workflows
  - Identifying the right data
  - Managing data transfers and execution-side storage
    - Reliability
    - On-time data delivery
    - Timely data offload
  - Keeping track of provenance information

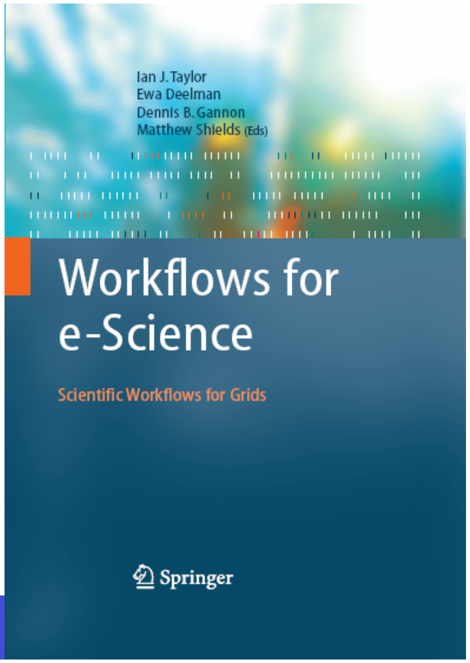
Acknowledgments

- **Pegasus: Gaurang Mehta, Mei-Hui Su, Karan Vahi, Arun Ramakrishnan (USC)**
- **DAGMan (in Pegasus-WMS): Miron Livny, Kent Wenger, and the Condor team (Wisconsin Madison)**
- **Wings: Yolanda Gil, Jihie Kim, Varun Ratnakar, Paul Groth (USC)**
- **LIGO: Kent Blackburn, Duncan Brown, Stephen Fairhurst, Scott Koranda (Caltech)**
- **Montage: Bruce Berriman, John Good, Dan Katz, and Joe Jacobs (Caltech, JPL)**
- **SCEC: Tom Jordan, Robert Graves, Phil Maechling, David Okaya, Li Zhao (USC, UCSD, others)**

Relevant Links

- Pegasus: pegasus.isi.edu
- DAGMan: www.cs.wisc.edu/condor/dagman

- Gil, Y., E. Deelman, et al. *Examining the Challenges of Scientific Workflows.* IEEE Computer, 2007.
- Workflows for e-Science, Taylor, I.J.; Deelman, E.; Gannon, D.B.; Shields, M. (Eds.), Dec. 2006



- Montage: montage.ipac.caltech.edu/
- LIGO: www.ligo.caltech.edu/
- Condor: www.cs.wisc.edu/condor/