From Data to Knowledge: Artificial Intelligence and Scientific Workflows

Yolanda Gil
Ewa Deelman

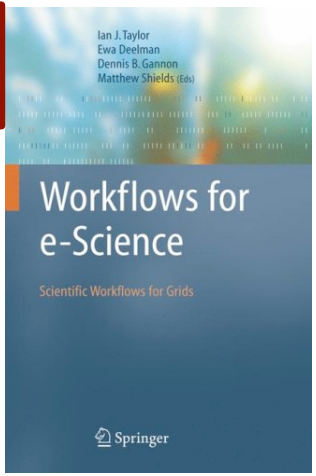
Information Sciences Institute
And Department of Computer Science
University of Southern California
gil@isi.edu

With Jihie Kim, Varun Ratanakar, Paul Groth, Gonzalo Florez, Pedro Gonzalez, Joshua Moody

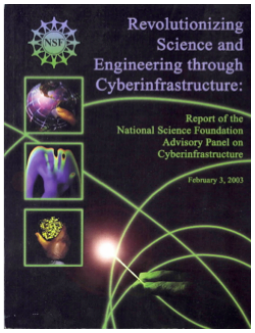
With Raphael Bolze, Rubing Duan, Gideon Juve, Gaurang Mehta, Prasanth Thomas, Karan Vahi

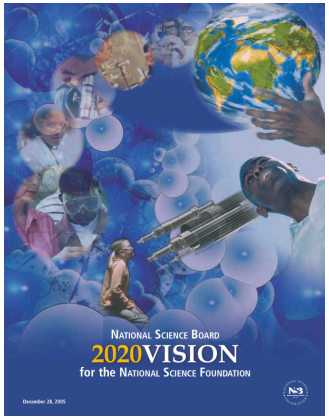
Reading About Scientific Workflows

- **"From Data to Knowledge to Discoveries: Scientific Workflows and Artificial Intelligence."** Yolanda Gil. To appear in Scientific Programming, 2009.
- **"Examining the Challenges of Scientific Workflows"**, Yolanda Gil, Ewa Deelman, Mark Ellisman, Thomas Fahringer, Geoffrey Fox, Dennis Gannon, Carole Goble, Miron Livny, Luc Moreau, and Jim Myers. IEEE Computer, vol. 40, no. 12, pp. 24-32, December, 2007.
- **"Workflows for e-Science: Scientific Workflows for Grids"**, Ian J. Taylor, Ewa Deelman, Dennis B. Gannon, and Matthew Shields (Eds). Springer Verlag, 2007.

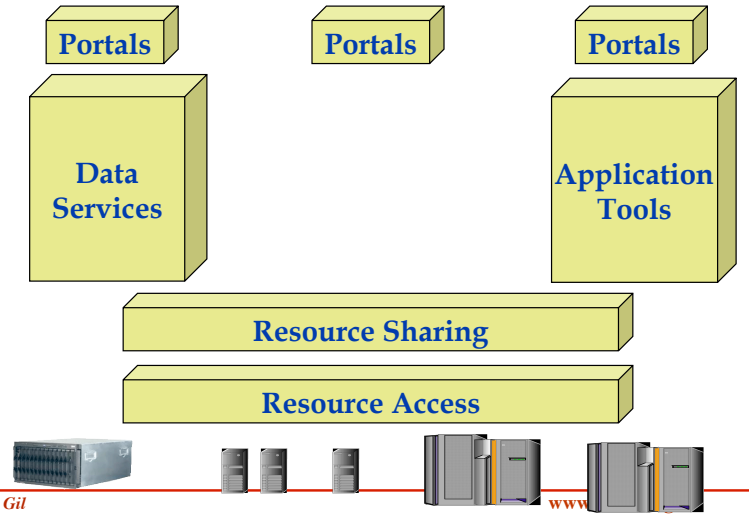


Computing and the Future of Science





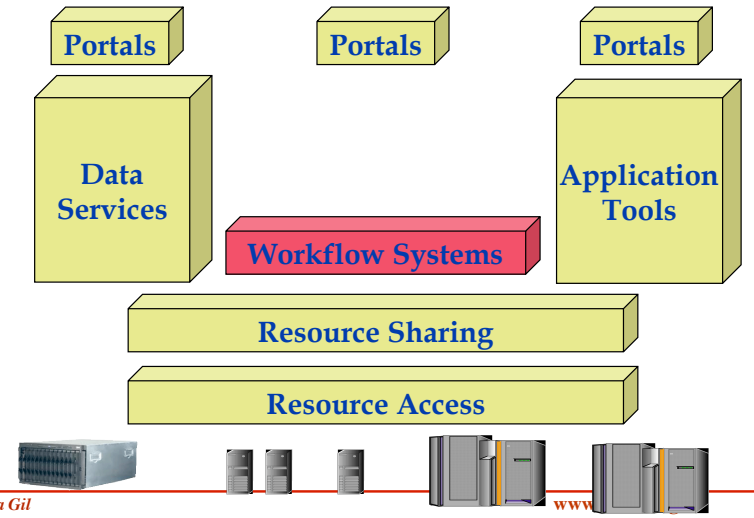
Common Cyberinfrastructure Layers



NSF Workshop on Challenges of Scientific Workflows (2006, Gil and Deelman co-chairs) [Gil et al 07]

- Despite investments on CyberInfrastructure as an enabler of a significant paradigm change in science:
  - **Exponential growth in Compute, Sensors, Data storage, Network BUT growth of science is not same exponential**
  - Reproducibility, key to scientific method, is threatened
- What is missing:
  - Perceived importance of **capturing and sharing process** in accelerating pace of scientific advances
  - Process (method/protocol) is increasingly complex and highly distributed
- **Workflows are emerging as a paradigm for process-model driven science that captures the analysis itself**
- Workflows need to be first class citizens in scientific CyberInfrastructure
  - Enable reproducibility
  - Accelerate scientific progress by automating processes
- Interdisciplinary and intradisciplinary research challenges
- **Report available at `http://www.isi.edu/nsf-workflows06`**

Workflow Systems as Key Cyberinfrastructure Layer



Computational Workflows

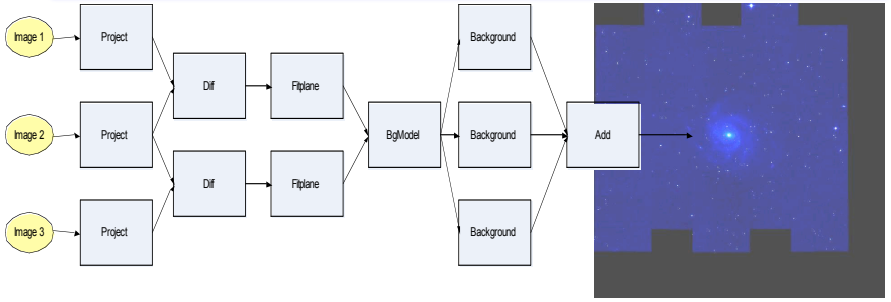
Management of Complex Applications as Scripts

- Scripts that specify the control structure of the application to be executed
  - Generate input values to all application codes from a starting input file
  - Determine the selection of application codes based on starting input file
  - Keep track of where new results come from (provenance)
- Scripts provide a common framework to compose models
- Scripts-based approaches are a first step in managing computation, used by many
- But…

Problems with Script-Based Approaches

- Adding a new requirement affects a lot of scripts
- Adding a new model (or a new version of a model) requires changes to starting input file and going through scripts by hand
  - Error prone process
- Ad-hoc data and execution management
  - Manually check whether intermediate data already exists
  - Metadata generated by scripts and passed around
  - To run the application at other hosts, the scripts have to be changed to have the right file paths
- Includes code to track how new data is generated
- Customized interfaces created for non-experts to ensure the application is run correctly

Complex Science Applications Are Not Monolithic: They Have A Workflow Structure



- **Workflow Components**
  - Standalone computations
  - Data inputs and outputs
- **Explicit data flow among Components**

Managing Scientific Applications as Computational Workflows

- Emerging paradigm for large-scale and large-scope scientific inquiry
  - *Large-scope science* integrates diverse models, phenomena, disciplines
  - "in-silico experimentation"
- Workflows provide a formalization of the scientific analysis
  - analysis routines need to be executed, the data flow amongst them, and relevant execution details
- Workflows provide a systematic way to capture scientific methodology and provide provenance information for their results
- Workflow are structures useful to manage computation
- Collaboratively designed, assembled, validated, analyzed

Abstraction Layers in Computational Workflows

Creation of Computational Workflows in Layers of Increasing Detail

1. Workflow Template (generic known-to-work recipes)
   - Specifies application components and dataflow among them
   - No data specified, just their type
2. Workflow Instance (data-specific)
   - Specifies data files for a given template
   - Logical file names, not physical file replicas
3. Executable Workflow (actual run)
   - Specifies physical locations of data files (may be in data repositories)
   - Assigned hosts/pools for execution of each component
   - Includes data movements among execution sites and data repositories as well as data deposition steps

Creation of Computational Workflows in Layers of Increasing Detail

User guided

1. Workflow Template (generic known-to-work recipes)
   - Specifies application components and dataflow among them
   - No data specified, just their type
2. Workflow Instance (data-specific)
   - Specifies data files for a given template
   - Logical file names, not physical file replicas
3. Executable Workflow (actual run)
   - Specifies physical locations of data files (may be in data repositories)
   - Assigned hosts/pools for execution of each component
   - Includes data movements among execution sites and data repositories as well as data deposition steps

Automated

Workflow Creation then Workflow Mapping

Workflow Creation Functions

Workflow formulation	Workflow assembly	Workflow completion
Workflow discovery	Workflow validation	Workflow sharing
Workflow elaboration	Data/parameter selection	Metadata generation

Workflow Mapping and Execution Functions

Site selection	Replica selection	Workflow optimization
Workflow submission	Failure recovery	Data management

The Wings/Pegasus Workflow System

[Gil et al 07; Deelman et al 03; Deelman et al 05; Kim et al 08; Gil et al forthcoming]

WINGS: Knowledge-based workflow environment
www.isi.edu/ikcap/wings
- Ontology-based reasoning on workflows and data (W3C's OWL)
- Workflow library of useful analyses
- Proactive assistance +automation
- Execution-independent workflows

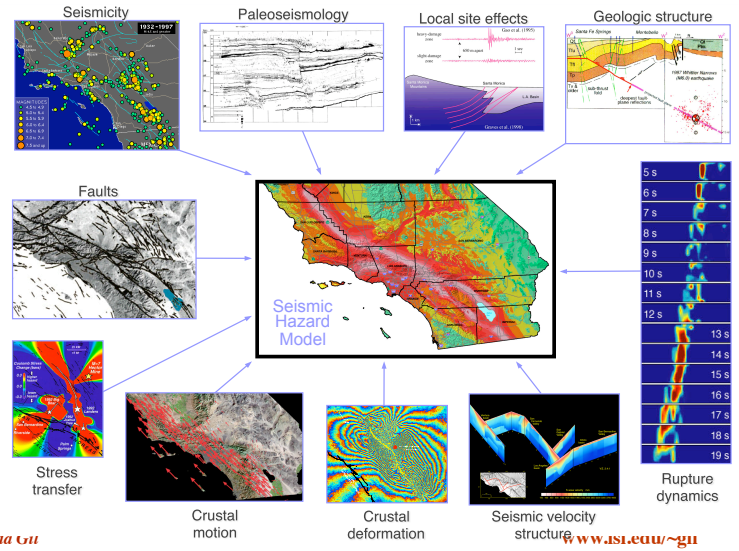
Pegasus: Automated workflow refinement and execution
pegasus.isi.edu
- Optimize for performance, cost, reliability
- Assign execution resources
- Manage execution through DAGMan
- Daily operational use in many domains

Grid services
condor.wisc.edu
www.globus.org
- Sharing of distributed resources
- Remote job submission
- Scalable service-oriented architecture
- Commercial quality, open source

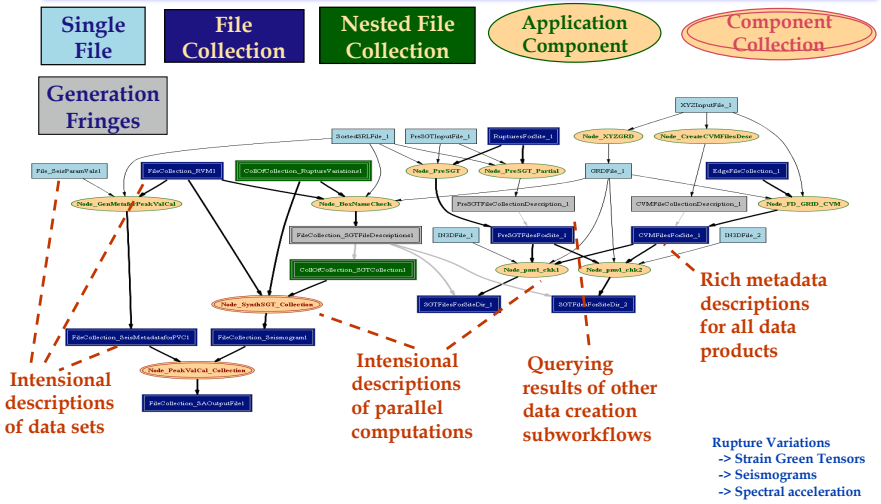
Wings/Pegasus for Workflow Generation and Metadata Propagation in Large-Scale Workflows

In collaboration with David Okaya, Kim Olsen, Tom Jordan, Phil Maechlin, and others at the Southern California Earthquake Center

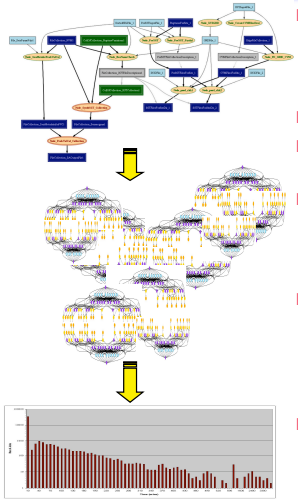
Physics-Based Seismic Hazard Analysis: SCEC's CyberShake [Slide from T. Jordan of SCEC]



A Wings Workflow Template for Seismic Hazard Analysis



Wings/Pegasus Workflows for Seismic Hazard Analysis [Gil et al 07]

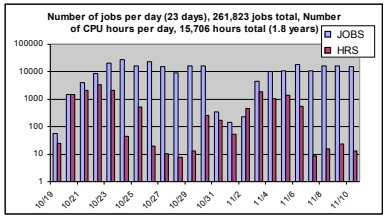


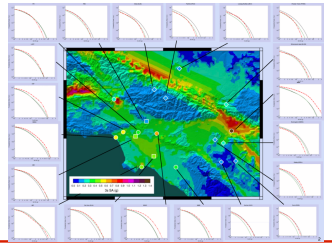
- Input data: a site and an earthquake forecast model
  - thousands of possible fault ruptures and rupture variations, each a file, unevenly distributed
  - ~110,000 rupture variations to be simulated for that site
- High-level template combines 11 application codes
- 8048 application nodes in the workflow instance generated by Wings
- 24,135 nodes in the executable workflow generated by Pegasus, including:
  - data stage-in jobs, data stage-out jobs, data registration jobs
- Executed in USC HPCC cluster, 1820 nodes w/ dual processors) but only < 144 available
  - Including MPI jobs, each runs on hundreds of processors for 25-33 hours
  - Runtime was 1.9 CPU years
- Provenance records kept for 100,000 workflow data products
  - Generated more than 2M triples of metadata

Pegasus: Large Scale Distributed Execution

[Deelman et al 06] Best Paper Award, IEEE Int'l Conference on e-Science, 2006

- Pegasus **managed 1.8 years of computation in 23 days** in NSF's TeraGrid
- Processed **20 TB of data** with **260,000 jobs**
- Daily operations of NSF's TeraGrid shared resources managed using Globus services
- Pegasus **managed ~840,000 individual tasks** in a workflow over a period of three weeks





Benefits of Computational Workflows

Workflow Creation then Workflow Mapping

Workflow Creation Functions

Workflow formulation	Workflow assembly	Workflow completion
Workflow discovery	Workflow validation	Workflow sharing
Workflow elaboration	Data/parameter selection	Metadata generation

Workflow Mapping and Execution Functions

Site selection	Replica selection	Workflow optimization
Workflow submission	Failure recovery	Data management

Benefits of Workflow Approaches [Gil 09]

Mapping and Execution:

- Automation of workflow execution
- Managing distributed computation
- Managing parallel computations
- Systematic exploration of parameter space
- Managing the evolution of an application
- Provenance recording
- Low-cost high fidelity reproducibility

Semantics and reasoning:

- Automation of workflow generation
- Systematic exploration of design space
- Validation of workflows
- Automated generation of metadata
- Guarantees of data pedigree
- "Conceptual" reproducibility

Benefits of Workflow Approaches [Gil 09]

Execution management:

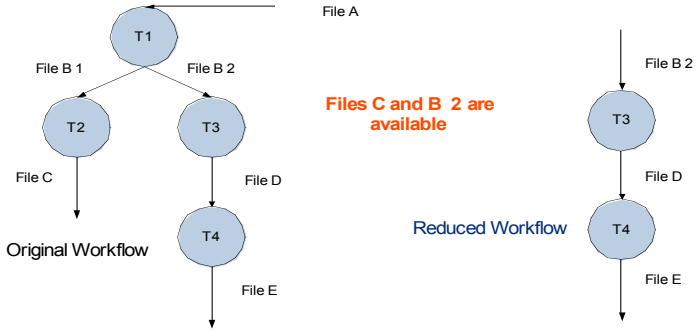
- Automation of workflow execution
- Managing distributed computation
- Managing parallel computations
- Systematic exploration of parameter space
- Managing the evolution of an application
- Provenance recording
- Low-cost high fidelity reproducibility

Semantics and reasoning:

- Automation of workflow generation
- Systematic exploration of design space
- Validation of workflows
- Automated generation of metadata
- Guarantees of data pedigree
- "Conceptual" reproducibility

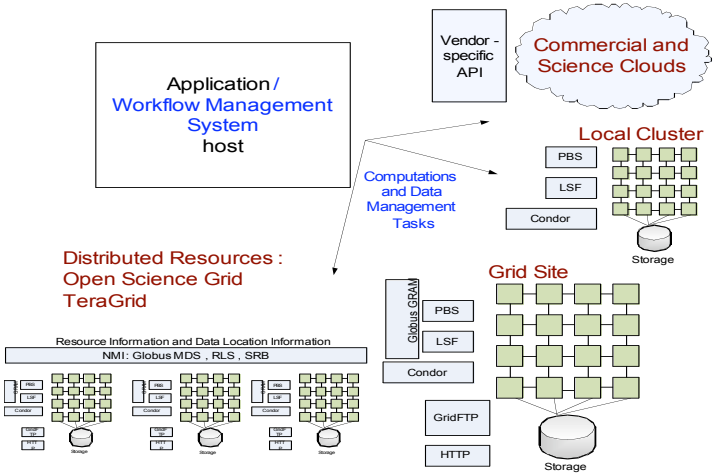
Pegasus: Data Reuse [Deelman et al 03]

- When it is cheaper to access the data than to regenerate it
- Keeping track of data as it is generated supports workflow-level checkpointing

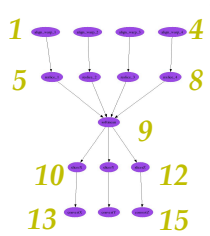


Mapping Complex Workflows Onto Grid Environments, E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, K. Backburn, A. Lazzarini, A. Arbee, R. Cavanaugh, S. Koranda, *Journal of Grid Computing, Vol.1, No. 1, 2003., pp25-39.*

Pegasus: Execution in a Distributed Environment [Deelman et al 08]



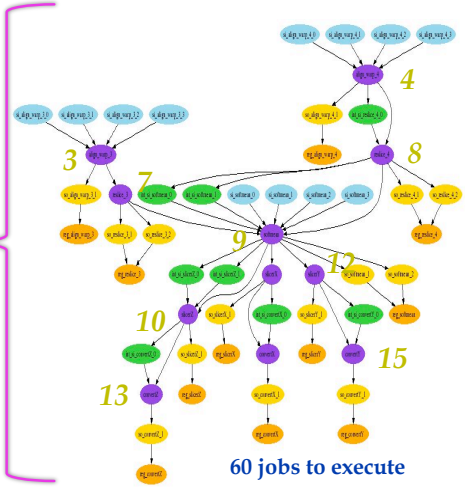
Pegasus: Workflow Mapping [Deelman et al 05]



Original workflow: 15 compute nodes devoid of resource assignment

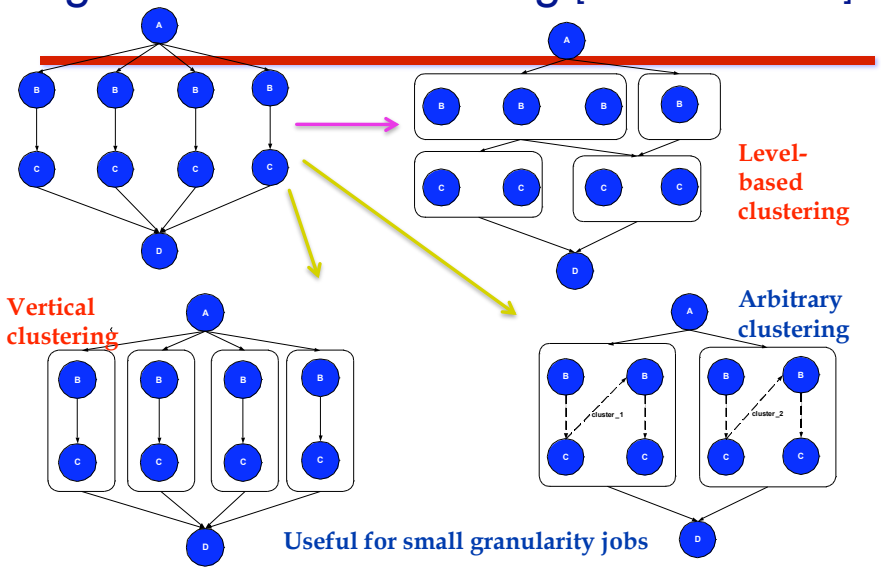
Resulting workflow mapped onto 3 Grid sites:

- 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)

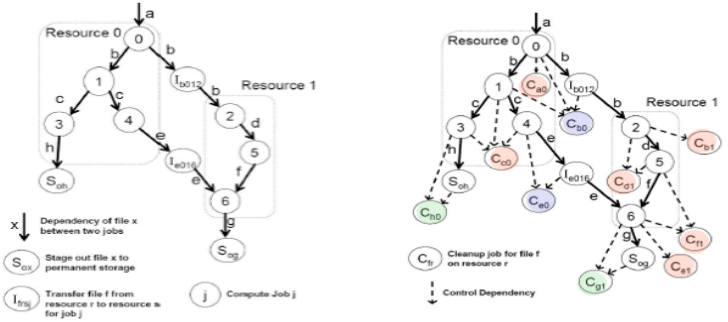


60 jobs to execute

Pegasus: Node clustering [Deelman et al 06]



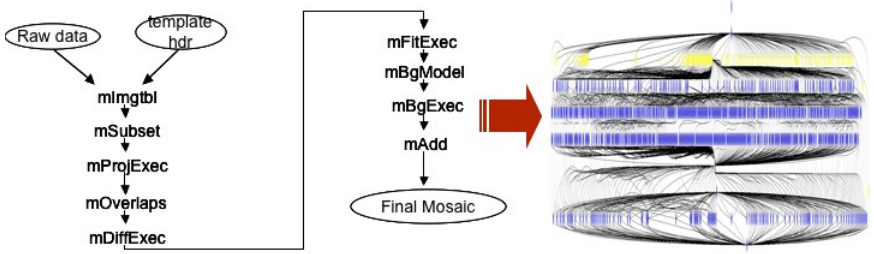
Pegasus: Cleanup Disk Space as Workflow Progresses [Deelman et al 07]

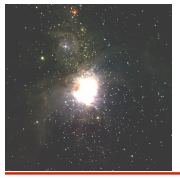


- For each node add dependencies to cleanup all the files used and produced by the node
- If a file is being staged-in from r1 to r2, add a dependency between the stage-in and the cleanup node
- If a file is being staged-out, add a dependency between the stage-out and the cleanup node

Pegasus for National Virtual Observatory and Montage [Berriman et al 06]

NVO's Montage mosaic application: Transformed a single-processor code into a workflow and parallelized





- Pegasus mapped workflow of 4,500 nodes onto NSF's TeraGrid
- Pegasus improved runtime by 90% through automatic workflow restructuring and minimizing execution overhead
- Montage is a collaboration between IPAC, JPL and Caltech

Montage: Composing a large image based on many individual images (Bruce Berriman, Caltech)

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

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

Science View [Katz et al 05]

- Montage is part of the National Virtual Observatory (www.nvo.org) and is used to create science-grade mosaics of the sky from multiple images that may have different characteristics (eg, different coordinate systems, projection, etc). Montage includes several application codes for re-projection into common scale and coordinates, modeling background radiation to minimize inter-radiation differences, rectification into common flux scale, and co-addition into a final mosaic. Montage can process data using two alternative approaches: one is a system that parallelizes computations implemented as a message passing interface (MPI) code that can be executed in a cluster, and the other uses Pegasus workflows to parallelize computations and execute them on distributed resources. Detailed comparisons showed that there is no notable difference in the execution performance of these two approaches, and that Pegasus has the additional advantages of fault tolerance and computation management [Katz et al 05]. Pegasus improved runtime by 90% over the original Montage design through automatic workflow restructuring and minimizing execution overhead [Berriman et al 06].

Benefits of Workflow Approaches [Gil 09]

Execution management:

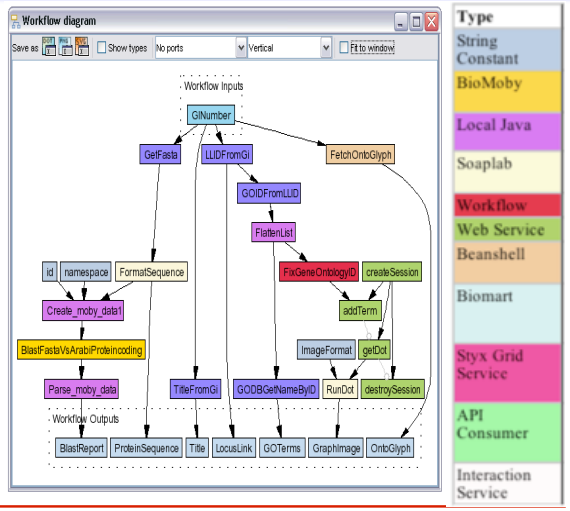
- Automation of workflow execution
- Managing distributed computation
- Managing parallel computations
- Systematic exploration of parameter space
- Managing the evolution of an application
- Provenance recording
- Low-cost high fidelity reproducibility

Semantics and reasoning:

- Automation of workflow generation
- Systematic exploration of design space
- Validation of workflows
- Automated generation of metadata
- Guarantees of data pedigree
- "Conceptual" reproducibility

TAVERNA [Goble et al 07; Hull et al 06; Oinn et al 07]

- Workflows of services
  - Web services, REST services, etc.
- Bioinformatics applications
  - 3,000+ 3rd party services: EMBL-EBI, NCBI, BioMOBY, KEGG, …



Science View [Fischer et al 07]

- A recent result obtained with Taverna is the identification of a candidate gene thought to be responsible for resistance to African tripanosomiasis [Fisher et al 07]. The workflow looks for correlations between phenotype in microarray data to Quantitative Trait Loci (QTL) genotype data. [Fisher et al 07] argues that when this kind of correlation is done manually there is no guarantee of a systematic consideration of hypotheses due to several features:
  1. eliminated datasets prematurely to reduce complexity,
  2. hypothesis-driven research dominates rather than complements data-driven research,
  3. user bias in pursuing hypotheses,
  4. re-analysis of data is hard due to changes in software interfaces and data availability,
  5. errors due to all the above.
- The workflow provides a mechanism to systematically and correctly explore variations of parameter settings. In addition, it is possible to re-analyze data since the provenance of any result is made available and the workflows are easily re-executed.

Benefits of Workflow Approaches [Gil 09]

Execution management:

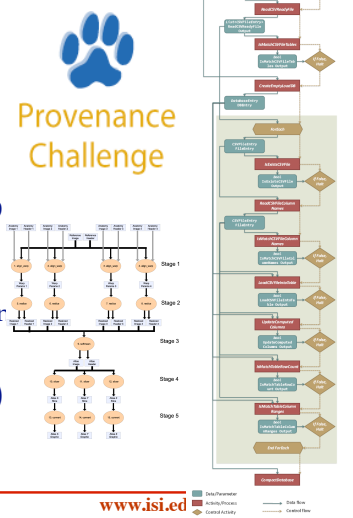
- Automation of workflow execution
- Managing distributed computation
- Managing parallel computations
- Systematic exploration of parameter space
- Managing the evolution of an application
- Provenance recording
- Low-cost high fidelity reproducibility

Semantics and reasoning:

- Automation of workflow generation
- Systematic exploration of design space
- Validation of workflows
- Automated generation of metadata
- Guarantees of data pedigree
- "Conceptual" reproducibility

Provenance (http://twiki.ipaw.info)

- 1st Provenance Challenge – 2006
  - Capabilities, queries, scope
  - fMRI workflow
- 2nd Provenance Challenge – 2008
  - interoperability
- 3rd Provenance Challenge – June 2009
  - Exchange provenance records among workflow systems
  - Answer queries about imported provenar
  - Pan-STARRS workflow
- Open Provenance Model (OPM)
  - http://openprovenance.org



Benefits of Workflow Approaches [Gil 09]

Execution management:

- Automation of workflow execution
- Managing distributed computation
- Managing parallel computations
- Systematic exploration of parameter space
- Managing the evolution of an application
- Provenance recording
- Low-cost high fidelity reproducibility

Semantics and reasoning:

- Automation of workflow generation
- Systematic exploration of design space
- Validation of workflows
- Automated generation of metadata
- Guarantees of data pedigree
- "Conceptual" reproducibility

Creating Analytic Pipelines for Genomic Analysis in GenePattern [http://www.genepattern.org]

1) Select an analysis (eg differential analysis)

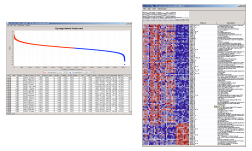
2) Select a module (eg Comparative marker selection)

3) Setup code

4) Choose data & parameter settings

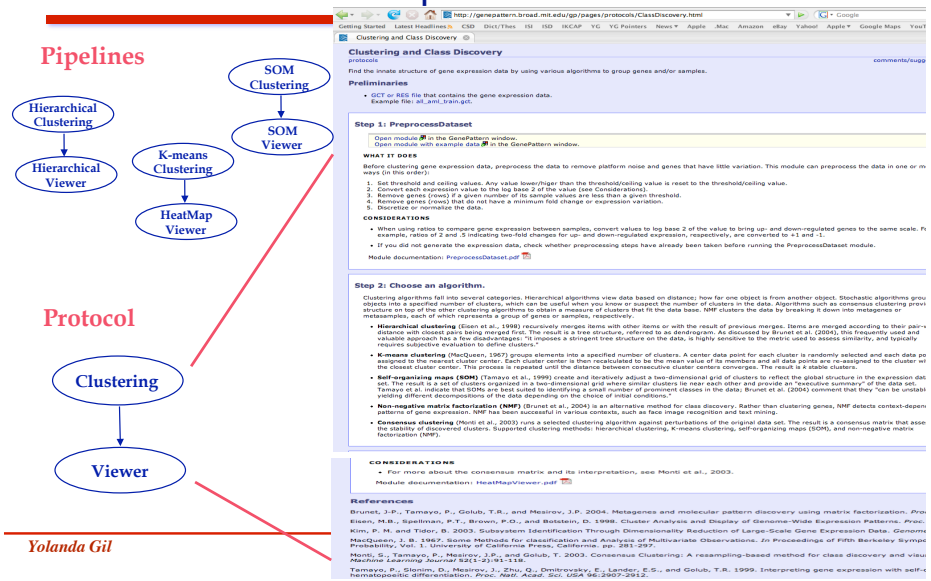
5) Select a viewer module

6) Run pipeline & view results



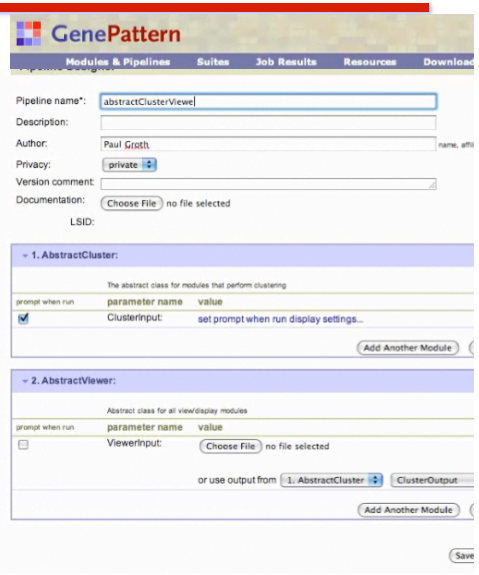
Using Modules: Documenting Constraints and Requirements

GenePattern Protocols: Documenting Constraints and Requirements



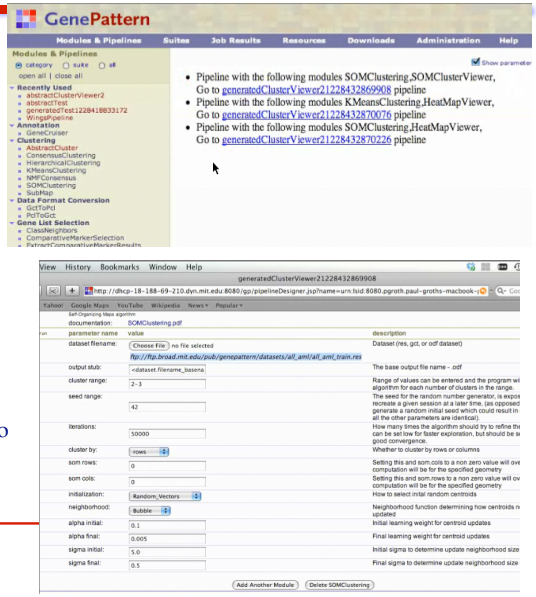
GenePattern / Wings Integration: 1) Defining reusable abstract protocols

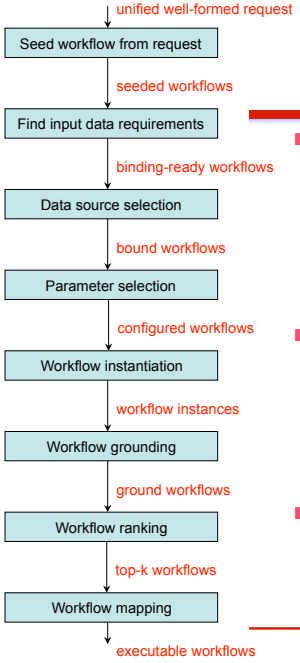
- Represent requirements constraints on modules declaratively
  - Wings can reason about how to instantiate pipelines while respecting those constraints
- Extend GenePattern module repository with **module classes**
- Define protocols as GenePattern pipelines that contain module classes as steps
  - If no links to an input it takes the output of its predecessor (default in GenePattern)
  - Can link an input to the output of an earlier module



GenePattern / Wings Integration: 2) Generating executable pipelines

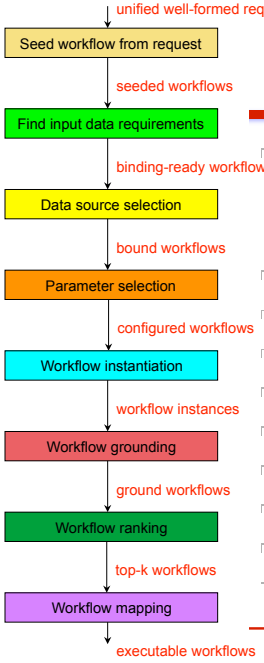
- Wings automatically generates valid pipelines from GenePattern protocols
  - Generates possible combinations of modules
  - Checks constraints of each module
  - Propagates constraints to other modules through dataflow
  - Eliminates invalid pipeline candidates
- Wings automatically sets required and consistent parameter values
  - Can also set default values
- Wings converts workflows to GenePattern execution format





Automatic Template-Based Workflow Generation Algorithm [Gil et al forthcoming]

- How it works: the algorithm automatically finds appropriate data sources, components, and parameter settings appropriate for the request
  - Maintains a pool of possible workflow candidates, each algorithm step elaborates existing candidates
  - Constraints on all workflow components, datasets, and parameters are propagated
  - Invalid workflow candidates are eliminated
- Input: Workflow request
  - Needs to specify:
    - High-level workflow template to be used
    - Additional constraints on data
  - Does not need to specify:
    - Data sources to be used
    - Particular component to be used
    - Parameter settings for the components
- Output: workflow instances ready to submit to Pegasus for mapping and execution
  - May eliminate or add candidates in the process



Why Do We Automate All This? So You Don't Have To

Request ID	Workflow candidates generated + considered (many are eliminated)			Queries about data	Queries about tools		Workflow Generation Time
R1	6	8	8	1	6	8	5 s
R2	6	8	8	7	6	16	4 s
R3	6	24	24	7	6	48	7 s
R4	6	24	24	13	6	72	8 s
R5	18	64	48	7	18	128	22 s
R6	18	288	216	7	18	576	81 s
R7	18	16	12	7	18	32	10 s
R8	6	0	0	1	6	0	1 s

Benefits of Workflow Approaches [Gil 09]

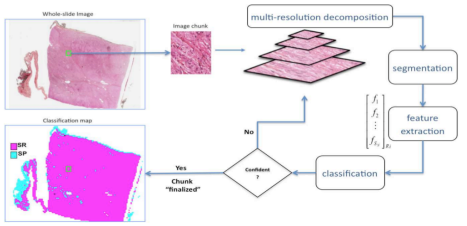
Execution management:

- Automation of workflow execution
- Managing distributed computation
- Managing parallel computations
- Systematic exploration of parameter space
- Managing the evolution of an application
- Provenance recording
- Low-cost high fidelity reproducibility

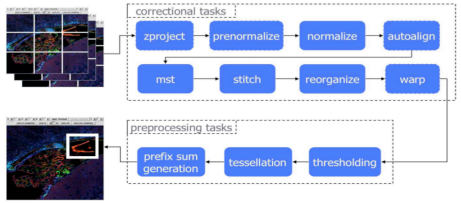
Semantics and reasoning:

- Automation of workflow generation
- Systematic exploration of design space
- Validation of workflows
- Automated generation of metadata
- Guarantees of data pedigree
- "Conceptual" reproducibility

Accuracy/Quality Tradeoffs in Large-Scale Biomedical Image Analysis

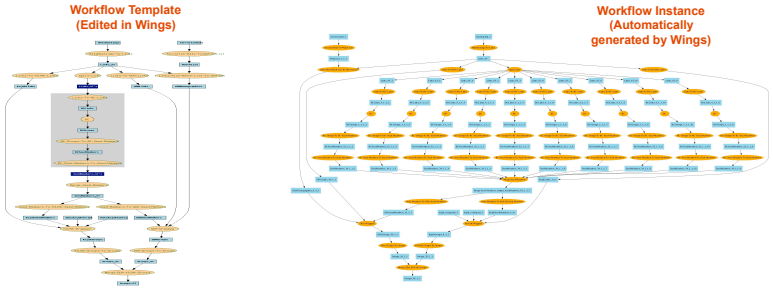


- **NC: Neuroblastoma Classification** (from OSU's BMI [Kong et al 07])
  - Diverse user queries:
    - "Minimize time, 60% accuracy"
    - "Most accurate classification of a region within 30 mins"
    - "Classify image regions with some min accuracy but obtain higher accuracy for feature-rich regions"
  - Easily parallelizable computations: per image chunk



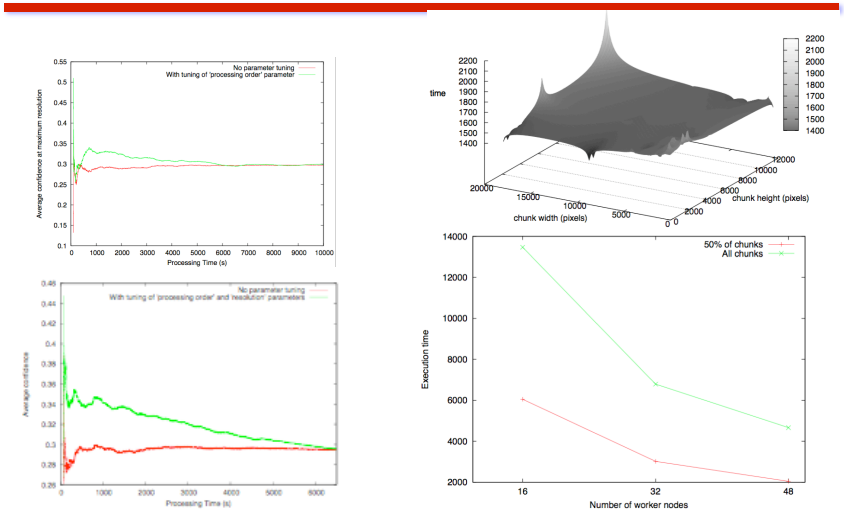
- **PIQ: Pixel Intensity Quantification** (from National Center for Microscopy and Imaging Research [Chow et al 06])
  - Terabyte-sized out-of-core image data
  - Need to minimize execution time while preserving highest output quality
  - Some operations are parallelizable, others must operate on entire images

Systematic Exploration of the Parameter Space



- Many degrees of freedom, many tradeoffs:
  - Processors available: amount and capabilities
  - Data processing operations: reduce I/O and communication overhead
  - Parameters of application components: vary accuracy and performance
  - Workflow parameters: degree of parallel branching based on chunking data into smaller datasets

Exploring Tradeoffs [Kumar et al 09]



Benefits of Workflow Approaches [Gil 09]

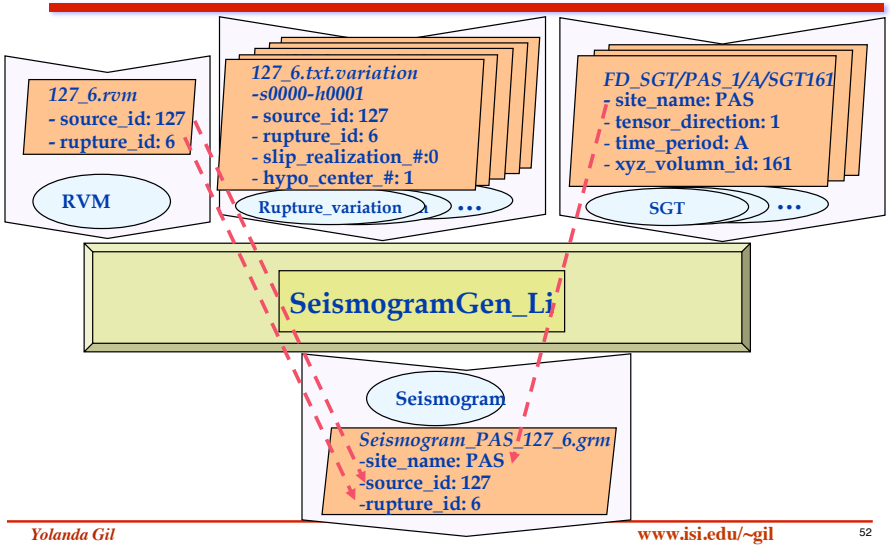
Execution management:

- Automation of workflow execution
- Managing distributed computation
- Managing parallel computations
- Systematic exploration of parameter space
- Managing the evolution of an application
- Provenance recording
- Low-cost high fidelity reproducibility

Semantics and reasoning:

- Automation of workflow generation
- Systematic exploration of design space
- Validation of workflows
- Automated generation of metadata
- Guarantees of data pedigree
- "Conceptual" reproducibility

Propagation of Metadata in WINGS



Benefits of Workflow Approaches [Gil 09]

Execution management:

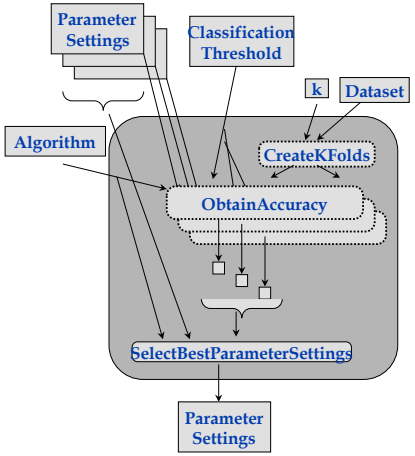
- Automation of workflow execution
- Managing distributed computation
- Managing parallel computations
- Systematic exploration of parameter space
- Managing the evolution of an application
- Provenance recording
- Low-cost high fidelity reproducibility

Semantics and reasoning:

- Automation of workflow generation
- Systematic exploration of design space
- Validation of workflows
- Automated generation of metadata
- Guarantees of data pedigree
- "Conceptual" reproducibility

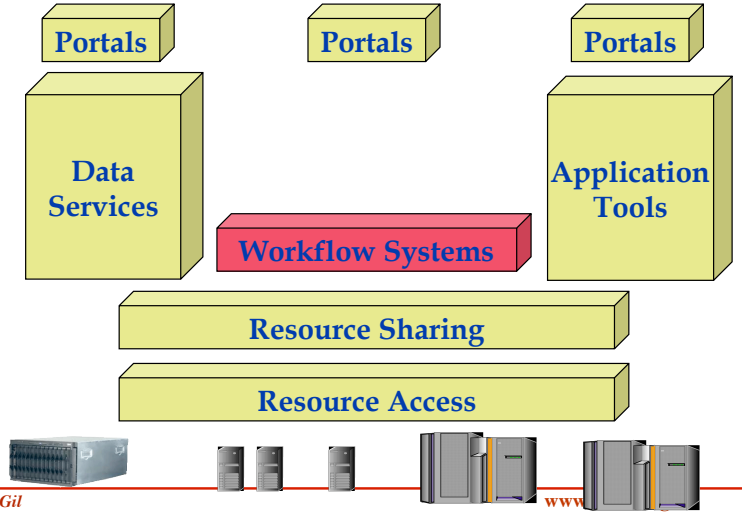
Conceptual Reproducibility

With Foster Provost, NYU

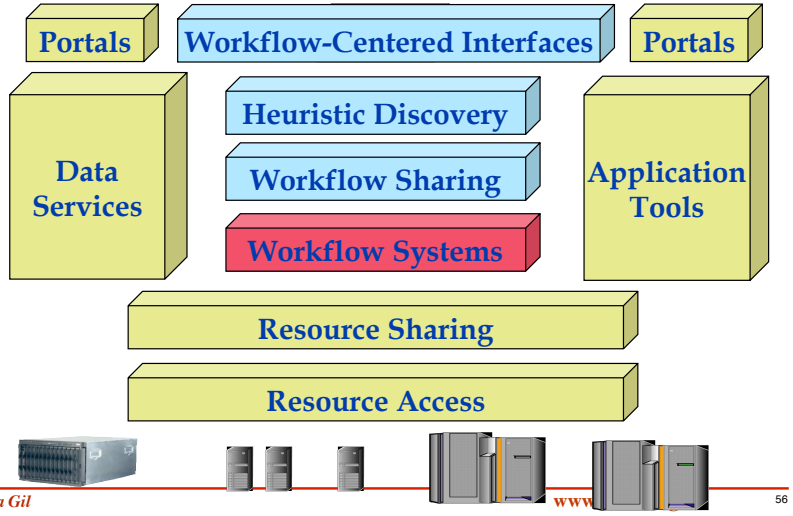


With Chris Mason, Yale

Workflow Systems as Key Cyberinfrastructure Layer



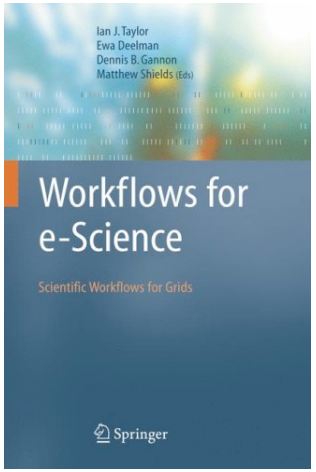
Tomorrow's Cyberinfrastructure Layers Enabled by Knowledge-Rich Workflow Systems [Gil 09]



Goal 5: From Portals to "Cockpits"

Reading About Computational Workflows

- **"From Data to Knowledge to Discoveries: Scientific Workflows and Artificial Intelligence."** Yolanda Gil. To appear in Scientific Programming, 2009.
- **"Examining the Challenges of Scientific Workflows"**, Yolanda Gil, Ewa Deelman, Mark Ellisman, Thomas Fahringer, Geoffrey Fox, Dennis Gannon, Carole Goble, Miron Livny, Luc Moreau, and Jim Myers. IEEE Computer, vol. 40, no. 12, pp. 24-32, December, 2007.
- **"Workflows for e-Science: Scientific Workflows for Grids"**, Ian J. Taylor, Ewa Deelman, Dennis B. Gannon, and Matthew Shields (Eds). Springer Verlag, 2007.



Goal 1: Reduce Setup Cost -> Workflow as First Class Citizen in Scientific Research

- Today: Workflow design and implementation is costly
  - Developed through collaboration
    - Application scientists in several areas, software engineers, distributed systems experts, etc.
  - Developed over many months
    - Must adapt existing code, must create "glue" code
  - Validated and refined over time
- Goal: Must be done by scientists themselves at minimal cost:
  - To create them
  - To understand them
  - To learn to use them for research
  - To adapt them for another purpose or analysis variant
  - To refine/update them over time

Goal 2: Workflows for Cross-Disciplinary Analyses -> Enable Integrative Science

- Today: Workflow systems can generate detailed provenance and metadata for new data products
  - Describe individual datasets so they can be used by others
  - Reuse of new data products by other systems is currently rare
    - Reuse is common within systems/communities
- Goal: Workflows generating data that is used across disciplines
  - Meaningful reuse of data products (results) by other workflows
  - True test of the utility of provenance and metadata information

Goal 3: Using Workflows for Educating New (and Old!) Scientists

- Today: Scientific analyses are less and less accessible to newcomers
  - Steep learning curve that includes a variety of areas of expertise
    - Application science(s), modeling, software engineering, distributed computing, etc.
- Goal: Workflow systems could be configured to enable learning of additional capabilities on-demand
  - Could isolate less proficient users from advanced capabilities while enabling them to learn and practice what they learn
  - Everyone should be able to contribute as they learn

Goal 4: Workflows as Efficient Instruments of Systematic Exploration and Discovery

- Today: Workflows manually selected by user
  - User decides what data/analysis to conduct
  - Not a systematic exploration of space
  - Visualization is only one way to understand results
  - Human is bottleneck, current practice will not scale
- Goal: Workflows conduct automated heuristic discovery and pattern detection
  - Automate systematic exploration of all possible workflows
  - Formulate heuristics for scientific discovery: recurring domain-independent data analysis patterns [Simon 82]
  - Search for patterns (or pattern types)
  - Workflows could include pattern detection and discovery components

Goal 5: From Portals to "Cockpits"

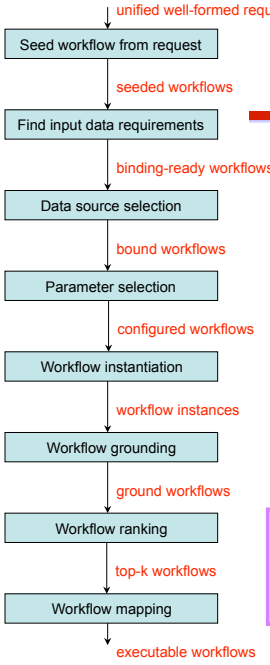
Goal 5: Consider a "Researcher Cockpit"

- "How a Cockpit remembers its speeds", E. Hutchins, *Cognitive Science*, 19, 1995, http://cognitrn.psych.indiana.edu/rgoldsto/cogsci/classics.html , see also [Hutchins and Klausen 95]

Abstract: In earlier research on the organization of work, Hutchins developed a theory of **distributed cognition that takes as its unit of analysis a culturally constituted functional group rather than an individual mind. This theory is concerned with how information is propagated through a system in the form of representational states of mediating structures.** These structures include internal as well as external knowledge representations, (knowledge, skills, tools, etc.). This approach permits us to describe cognitive processes by tracing the movement of information through a system and characterize the mechanisms of the system which carry out the performance, both on the individual and the group level. In this paper we apply this approach to the structure of activity in a commercial airline cockpit. A cockpit provides an opportunity to study the interactions of internal and external representational structure and the distribution of cognitive activity among the members of the crew. Through an analysis of audio and video recordings of the behaviors of real airline flight crews performing in a high fidelity flight simulator **we demonstrate that the expertise in this system resides not only in the knowledge and skills of the human actors, but in the organization of the tools in the work environment as well.** The analysis reveals a pattern of cooperation and coordination of actions among the crew which on one level can be seen as a structure for propagating and processing information and on another level appears as a system of activity in which **shared cognition emerges as a system level property.**

Goal 5: Workflows as "Cockpit Instrument"

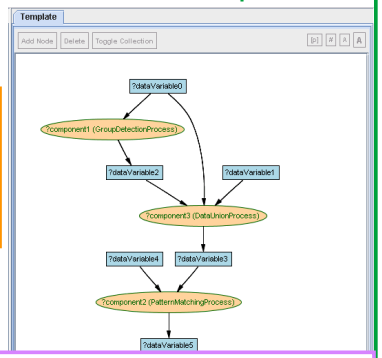
- Workflow template as "flight plan"
- User visibility into the data analysis process
- User steering during execution based on results
- Interleaving generation and execution (data-driven adaptation)
- Recording provenance = "flight log"
- Automation = "automatic pilot"



Automatic Template-Based Workflow Generation Algorithm

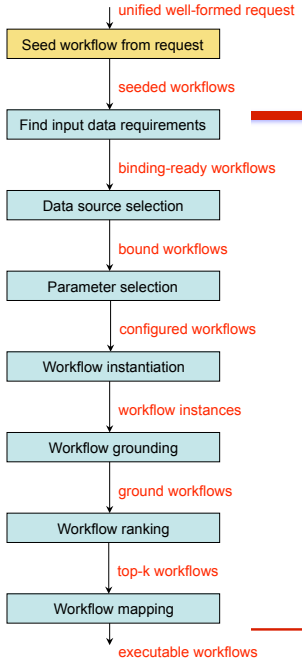
Workflow request =
Workflow Template
+
Seed Constraints

WR0: Workflow Template

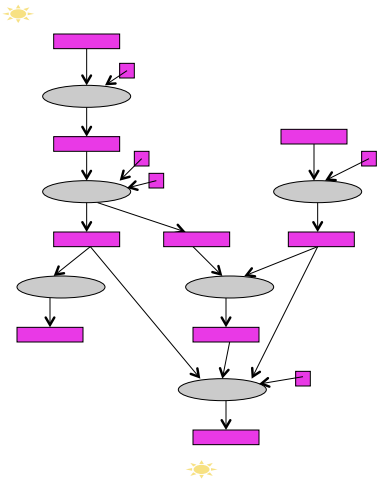


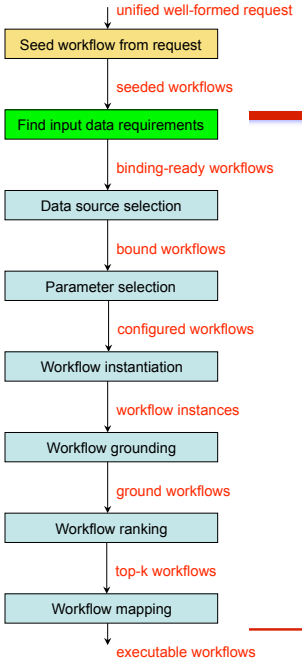
WR0: Seed Constraints

dataVariable5 process:saturatedWith data:ChainedCommunicationEvent
dataVariable0 data:creator 5048
dataVariable1 data:creator 5048

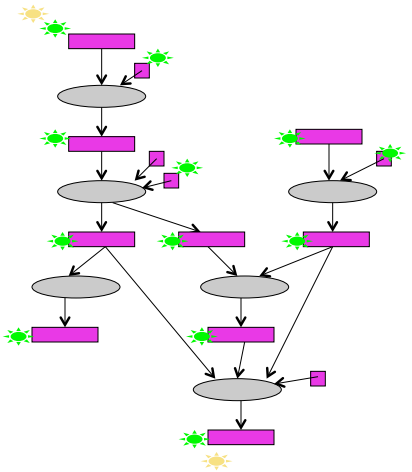


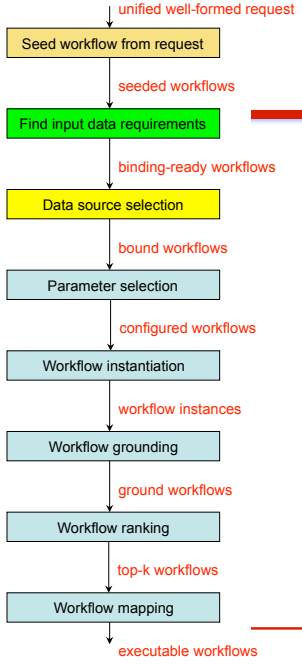
Step 1: Workflow Template is Seeded



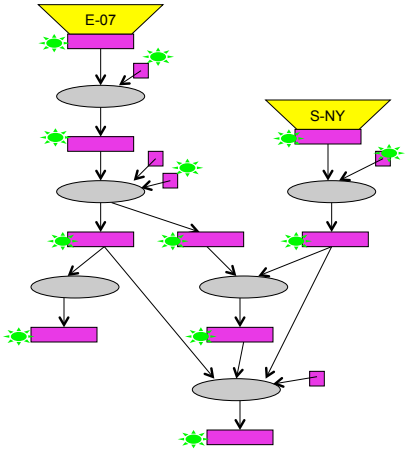


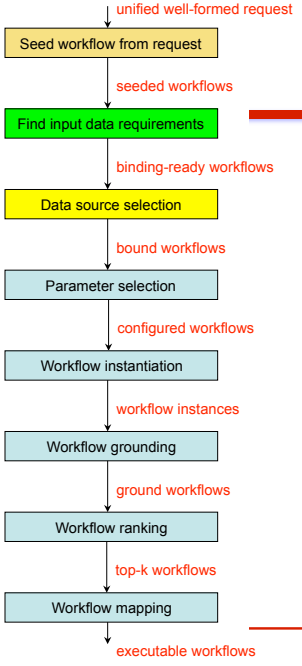
Step 2: Backward Sweep



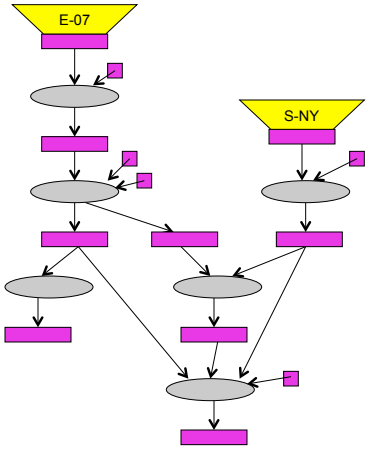


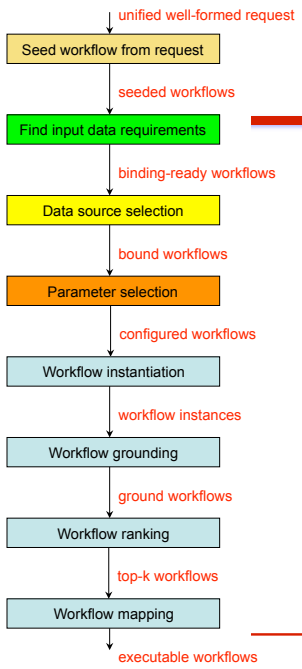
Step 3: Select Data Sources



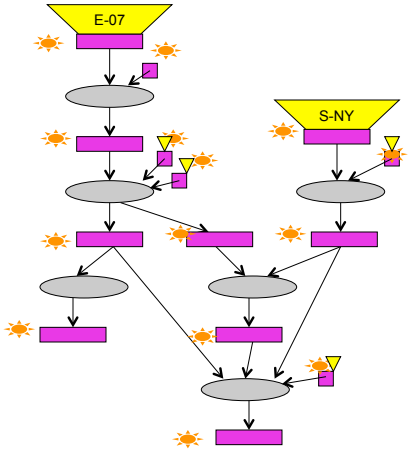


Step 3: Select Data Sources

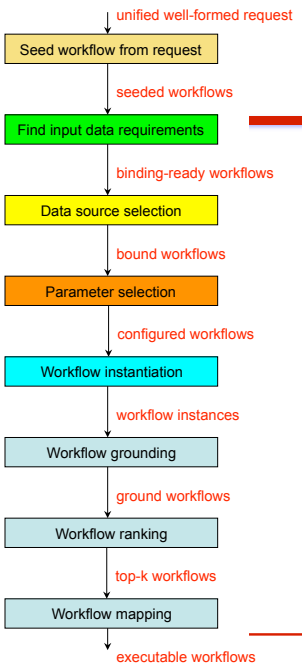




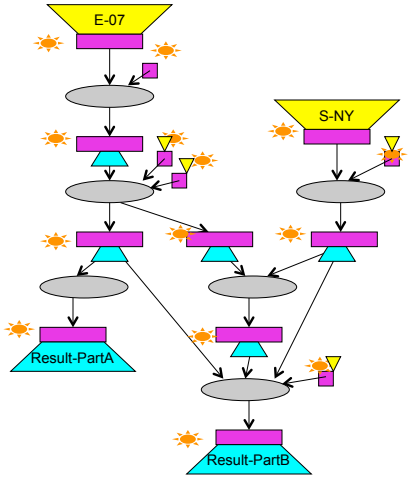
Step 4: Forward Sweep

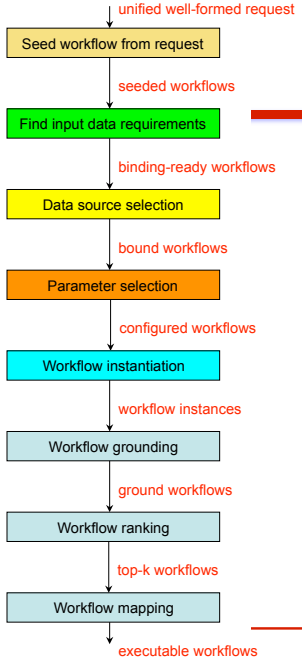


Step 5: Workflow Instantiation

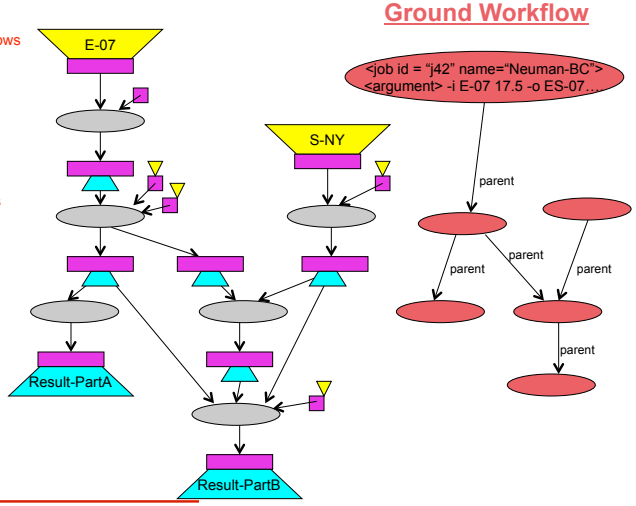


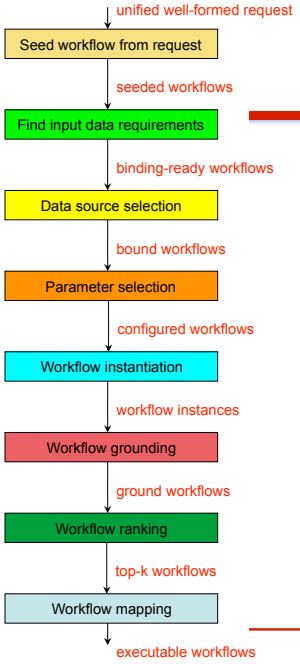
Step 5: Workflow Instantiation



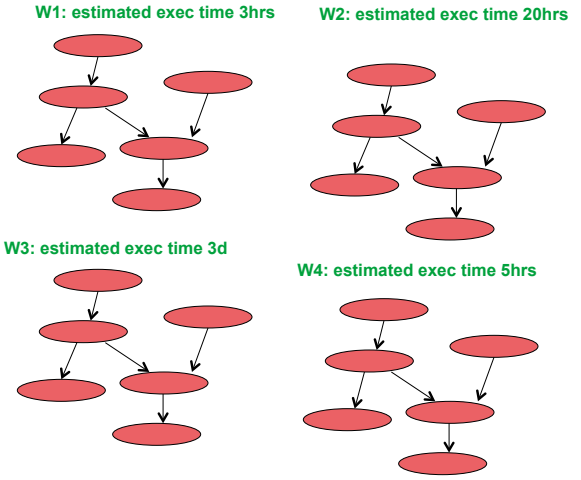


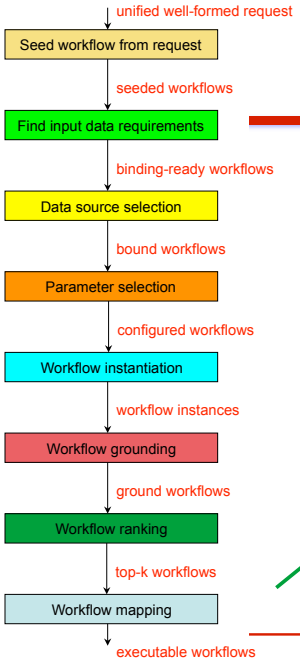
Step 6: Workflow Grounding



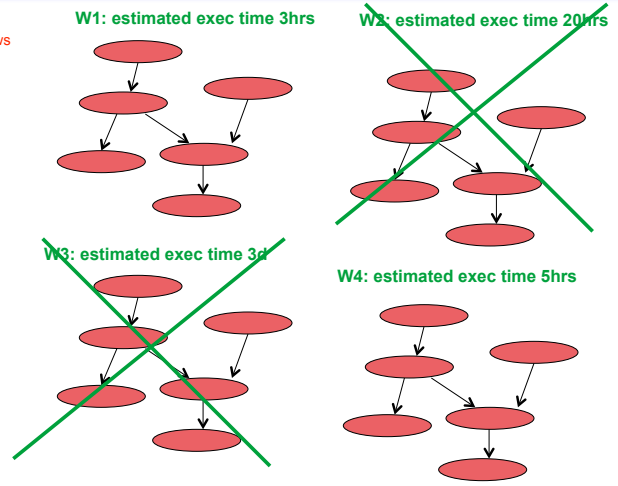


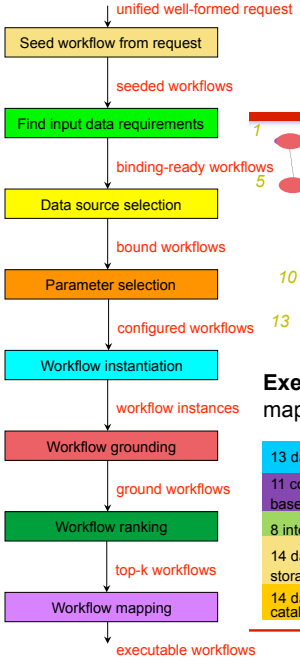
Step 7: Workflow Ranking



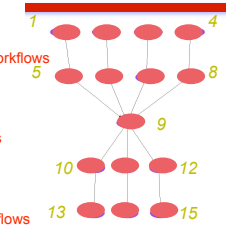


Step 7: Workflow Ranking



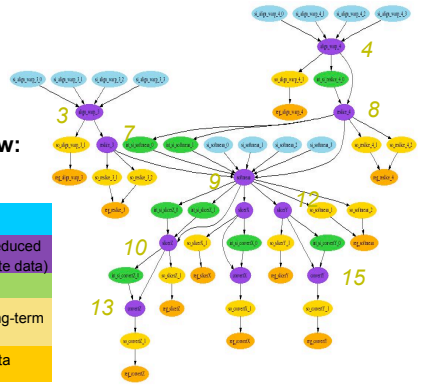


Step 8: Workflow Mapping

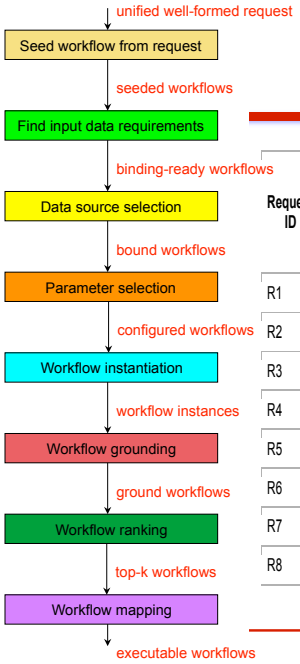


Ground workflow: 15 compute nodes devoid of resource assignment

Executable workflow: mapped to 3 sites



- 13 data stage-in nodes
- 11 compute nodes (1-2&5-6 reduced based on available intermediate data)
- 8 inter-site data transfers
- 14 data stage-out nodes to long-term storage
- 14 data registration nodes (data cataloging)

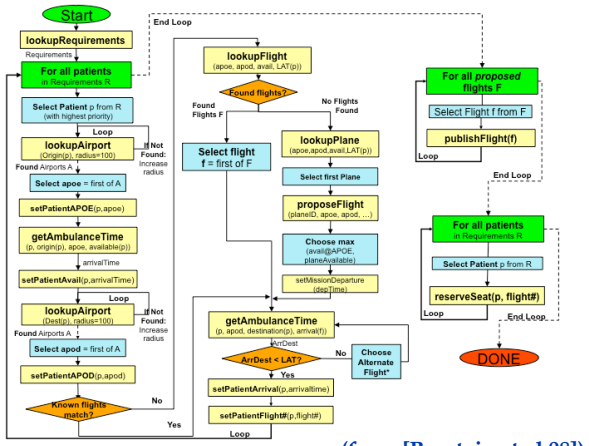


Why Do We Automate All This? So You Don't Have To

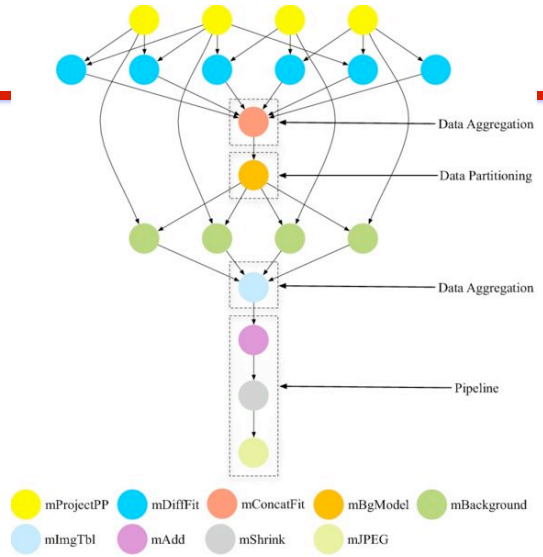
Request ID	Workflow candidates generated + considered (many are eliminated)			Queries about data	Queries about tools		Workflow Generation Time
R1	6	8	8	1	6	8	5 s
R2	6	8	8	7	6	16	4 s
R3	6	24	24	7	6	48	7 s
R4	6	24	24	13	6	72	8 s
R5	18	64	48	7	18	128	22 s
R6	18	288	216	7	18	576	81 s
R7	18	16	12	7	18	32	10 s
R8	6	0	0	1	6	0	1 s

Learning Workflows

1) From a user's demonstration of service invocations [Kim & Gil 08]
2) From tutorial instruction [Groth & Gil 08]



(from [Burstein et al 08])



Ewa Deelman, deelman@isi.edu www.isi.edu/

Montage workflow with ~ 1,200 nodes