

# Simplifying Construction of Complex Workflows for Non-Expert Users of the Southern California Earthquake Center Community Modeling Environment

Philip Maechling (3), Hans Chalupsky (2), Maureen Dougherty (2), Ewa Deelman (2), Yolanda Gil (2), Sridhar Gullapalli (2), Vipin Gupta (3), Carl Kesselman (3), Jihie Kim (2), Gaurang Mehta (2), Brian Mendenhall (1), Thomas Russ (2), Gurmeet Singh (2), Marc Spraragen (2), Garrick Staples (1), Karan Vahi (2) (1) Center for High Performance Computing and Communications, USC, Los Angeles, CA 90089, USA, (2) Information Sciences Institute, USC, Marina del Rey, CA 90292, (3) Southern California Earthquake Center, USC, Los Angeles CA, 90089, USA, {Corresponding Author: maechlin@usc.edu}

## Abstract:

*Workflow systems often present the user with rich interfaces that express all the capabilities and complexities of the application programs and the computing environments that they support. However, non-expert users are better served with simple interfaces that abstract away system complexities and still enable them to construct and execute complex workflows. To explore this idea, we have created a set of tools and interfaces that simplify the construction of workflows. Implemented as part of the Community Modeling Environment developed by the Southern California Earthquake Center, these tools, are integrated into a comprehensive workflow system that supports both domain experts as well as non expert users.*

## Keywords:

Workflow, Grid, Intelligent Assistant, Seismic Hazard Analysis, Data Management, Scheduling, Pegasus, Meta-scheduler, Metadata, Semantic Web

## 1. Introduction:

For many regions of the globe, earthquakes represent a significant hazard to life and property. This has lead geophysicists to study the earth as a system, to gain a better understanding of the complex interactions between crustal stress, fault ruptures, wave propagations through 3D velocity structures, and ultimately, ground motions. While the holy grail of this work is earthquake prediction, a more prosaic goal is to estimate the potential for earthquake damage at specific locations on the earth and this *seismic hazard analysis* (SHA) is one of the main objectives of earthquake geophysics investigation.

Recent advances have resulted in increasingly more accurate models of the different system components: fault models, rupture dynamics, wave propagation, etc. Given this progress, it becomes obvious that attention must turn to combining these diverse elements into an integrated model of the earth. It is with this goal in mind that that the Southern California Earthquake Center (SCEC) created the Community Modeling Environment (SCEC/CME) [14]. The CME is an integrated environment in which a broad user community encompassing geoscientists, civil and structural engineers, educators, city planners, and disaster response teams can have access to powerful physics-based simulation techniques for SHA. The diversity and distribution of the user community combined with the complexity of the problem space imposes

some demanding requirements on any proposed solution. These include:

- The need to deliver complex computational methods to wide range of users, from non-expert (i.e. non-geophysicists) who need hazard information to domain experts who are using the environment as part of their research program.
- The need to support multiple models, and data types. For example, the community has developed alternative earth velocity models each of which has valid uses, but produces different results.
- The distributed nature and specializations of the geophysical research community leads to distributed model development with models being developed by different organizations with differing expertise and computational resources.
- The computational requirements of earthquake models require high performance and high throughput computing techniques and the computational and data resources may be dispersed across institutions and administrative domains.

To address these requirements, the approach we take is to represent computational models as scientific workflows targeted towards execution in a distributed Grid environment. While many options exist for specification and execution of scientific workflows, addressing the SCEC community requirements dictates that existing Grid workflow solutions be combined with new user-centric interfaces in unique ways. The result has been the creation of a system for the specification and execution of scientific workflows with the following features:

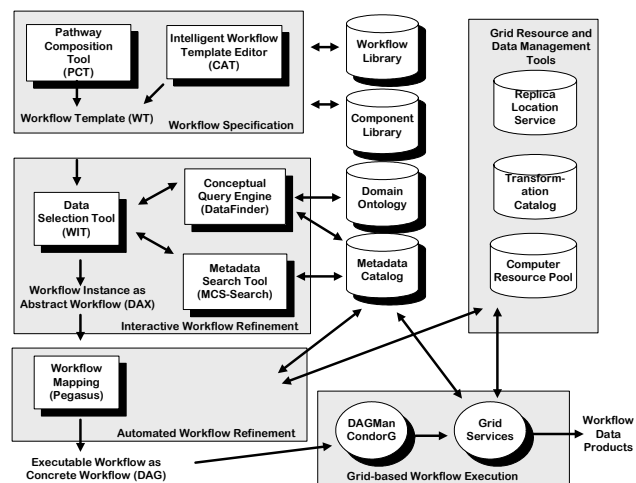
- User-centered knowledge based interface that helps users express geophysical problems as high-level workflow specifications.
- Knowledge-based metadata search tools integrated with data and metadata management tools allowing users to search for data by concept rather than by metadata attribute name and value.
- Workflow refinement tools based on the Virtual Data Systems (VDS) [11] toolkit for mapping workflow specifications into executable form.
- A Grid based execution environment that enables execution of workflows against domain specific software libraries, data sources, information services and national class execution and storage resources such as the National Science Foundation's TeraGrid environment.

We note that while the CME consists of many components and offers a rich set of functionalities,

we focus on the system's workflow specification and management capabilities in this paper.

## 2. SCEC/CME Workflow System Overview

During development of the SCEC/CME workflow system, we have found it useful to analyze the construction and execution of workflows in three phases which are: (1) workflow specification (2) data discovery and workflow refinement, and (3) grid-based workflow execution. Fig. 1 shows the tools and data stores used in SCEC/CME workflow system and how they map to these phases.



**Figure 1: The SCEC/CME system supports workflow specification, refinement, and grid-based execution.**

In the workflow specification phase, the user describes the problem to be solved in high-level geophysical terms and the CME assists in mapping these into high-level workflow descriptions or *workflow templates*. These descriptions define the process but do not yet include information about specific data instances or other detailed configuration information needed to execute the workflow. The data discovery and workflow refinement step fills in this missing detail, identifying data sets, selecting compute resources to execute simulation steps, determining when and where to stage data. In the final phase, a Grid-based workflow execution engine is used to execute the descriptions, keeping track of what tasks have been completed, and executing new tasks as they become ready.

One of our development challenges is to support both expert and non-expert users. A characteristic of our system is to provide the user with a selection of tools in each of these workflow phases. In order to support users with differing levels of domain and computational sophistication our system includes both manual and knowledge-based tools. For example, in the workflow specification phase, we provide a standard graphical workflow specification tool (the Pathway Composition Tool), as well as a knowledge-based intelligent assistant (Composition Analysis Tool) interface that guides users to a

computational solution defined as a workflow. Also, in the data discovery and workflow refinement phase, we provide both text-matching metadata search tools as well as semantic metadata search tools to help users locate existing data.

## 3. Workflow Specification

The first step in generating PSHA data is to identify a simulation workflow that solves the user's problem by generating the desired data. We provide two browser-based workflow specification tools to help map problem specifications in terms of domain semantics into high-level workflow templates that define the basic computational steps in a solution.

To aid non-expert users we provide an *Intelligent Assistant* workflow construction tool called the Composition Analysis Tool (CAT) [15,16]. CAT facilitates interactive construction of computational pathways where users select and connect existing SCEC components, and the system assists in completing a correctly formulated computation. Given an outline of a workflow, CAT analyzes it using the semantic description of components, reports errors and gaps, and generates specific suggestions on how to fix these errors. Users often construct a workflow template using CAT by specifying a starting data type and a target data type and then CAT provides suggestions on which computational modules are needed in the workflow to transform the starting data type to the target data type.

The CAT system is implemented in two components; 1) a browser-based user interface tool that guides the user through the creation of a workflow template, and 2) a web service-based reasoner (called the Composition Analysis Tool Service (CAT-S)) that includes the reasoning and validation algorithms. One of the features of CAT is that it can check workflows for validity, i.e.: (a) all links are consistent with the component descriptions and their input and output data types, (b) all computational steps have linked inputs, (c) an end result has been specified, (d) all computational steps are executable, and (e) all computational steps contribute to the results. CAT uses the Web Ontology Language (OWL) standard as its knowledge representation language so the component ontology used by CAT-S for validation is expressed in OWL.

For expert users we also provide a fairly standard browser-based graphical workflow editing tool called the Pathway Composition Tool (PCT). Both tools (CAT and PCT) output workflow templates in a common XML format. However, in the case of PCT, the output workflow template has not been validated while a workflow template produced by CAT has been validated based on the semantic description of the task and constituent components.

## 4. Interactive Workflow Refinement

Given a specification of the processing steps that need to be performed, the workflow must be refined to identify what input data to apply the processing to, what implementations to use and where to perform the computations. The first step in this process is to identify the input data to be used for the workflow.

Before a workflow template can be used, the user must identify what data sets to which the workflow should be applied. We refer to the process of discovering, selecting, and specifying initial input files as the interactive workflow refinement phase.

In the SCEC/CME system, a user performs interactive workflow refinement using a tool we call the Workflow Instantiation Tool (WIT). WIT is a browser-based tool with which the user selects the input data instances to be used in the workflow computation. Data instances that will be calculated during the workflow execution do not need to be specified. However, data files that represent initial inputs to the workflow must be specified.

The WIT interface presents the user with each computational step in the workflow under refinement, and it shows the input data types, the computation type, and the output data types for each step. For each non-computed input data file, it presents the user with a list of existing data files of the appropriate type. WIT obtains information about the existing data files in the workflow system through the use of a Metadata Catalog Service (MCS) [7] that contains entries for all data files instances, and their associated data types, currently in the system. MCS metadata is represented by name-value pairs where each data type in the SCEC/CME system has a minimum set of required metadata attributes. As workflows execute, the workflow computational modules generate the required metadata for the data files that they produce and they register this metadata into the MCS.

In order to enable file replication, the MCS does not store file names directly but rather records a globally unique name in the form of a Uniform Resource Identifier (URI). When data files are imported into the system, or are created by computational modules, we form a URI by concatenating a SCEC namespace, a data type name, and a sequence number generated from the database. The URI can be mapped to one or more actual file names by another component of the SCEC/CME system called the Globus Replica Location Service (RLS).

If the user knows the URI of the desired input data, they may be specified directly. Otherwise, the SCEC/CME system provides two browser-based data discovery tools to help the user locate the appropriate files. One of the tools, designed for sophisticated users, provides string-based metadata attribute search capabilities. The other data discovery tool, designed to support non-expert users, provides concept-based metadata searches. With both data discovery tools, the user specifies some information describing the desired file(s) and the search tools query the system metadata to return the URIs of matching files.

The first tool, called MCS-Search, is a simple, metadata attribute-based, search tool. The user enters attribute names and values, and the system uses string-based matching to locate logical files with the desired attributes.

To improve upon this basic name-value, string-matching metadata search capability, we have also developed a semantic metadata search tool called DataFinder based on the PowerLoom knowledge

representation system [18]. DataFinder utilizes a concept-based domain and metadata attribute ontology that links geophysical and seismic hazard domain concepts with the metadata attributes that describe the computational products.

DataFinder provides semantic overlays for the existing metadata attributes, enriching the information content. The DataFinder domain and metadata attribute ontology is represented in the PowerLoom representation language, based on KIF [13]. DataFinder is implemented using a hybrid reasoning approach based on combining the strengths of the PowerLoom logical reasoning engine with the database technology underlying the MCS.

The direct connection between DataFinder and the MCS database is achieved via PowerLoom's database access layer, which provides DataFinder with the necessary scalability to handle the task of locating relevant data products in a large repository. It also allows us to add semantic enhancements by overlaying the raw metadata with a hierarchy of concepts, providing for more abstract views. The DataFinder system translates the domain concepts specified by the user into SCEC/CME metadata attributes. The DataFinder search system allows users to find data instances without requiring detailed knowledge of the (multiple) specific metadata attributes used to describe the data.

When all initial, non-computed, input data files have been discovered, the user can interactively refine the workflow specification by specifying a URI for each input file. Once URIs have been specified for all non-calculated data instances, interactive workflow refinement is complete.

The output of the interactive workflow refinement phase is a representation of the workflow we call an *abstract workflow*. An abstract workflow specifies the workflow using logical references to both the files, and to the computational modules to be used. This has several advantages. By specifying the data file instances logically, the automated workflow refinement parts of the system can select the most appropriate file replicas. By specifying the transformations logically, later elements of the workflow system can select both the transformation instance and computing resources to be used during execution.

## 5. Automated Workflow Refinement

In the automated workflow refinement phase, we use a sophisticated execution planning system to convert an abstract workflow into a concrete workflow. In this process, URIs must be mapped to actual input files, alternative implementations for computational steps chosen and execution sites selected. In addition, we must orchestrate the movement of data between computational sites as required by the execution sites and data sources.

This conversion is performed during the automated workflow refinement phase of our system using Pegasus (Planning for Execution in Grids) [4,5,6]. The Pegasus system performs automated workflow refinement utilizing a variety of tools including the Virtual Data System (VDS) that

includes Chimera, Condor DAGMan, MCS, RLS, and Globus. Program scheduling and execution tools are based on the high throughput Condor job submission and scheduling tools.

The automated workflow refinement capabilities of the Pegasus system are one more way that the SCEC/CME system shields users from workflow complexities. Non-expert users are frequently not familiar with computing details such as how to match executables to hosts. Our system supports these users by establishing transformation to compute resource mappings in data stores such as the VDS transformation catalog and then using Pegasus to automatically select appropriate execution environments for each transformation in a workflow.

Pegasus interacts with other VDS and Globus components to drive the process of refinement of workflows from an abstract format to an executable concrete workflow specification. These include a transformation catalog that contains descriptions of the SCEC/CME transformations and RLS which is used to map URIs specified during interactive refinement into specific file instances.

During the conversion from DAX to DAG, the system queries RLS and maps each URI to an appropriate physical file name based on the location of the replicas and the location where the data file needs to be read, and replaces each URI with a URL. The system also converts each logical transformation name to physical transformation name. Pegasus also selects an appropriate computer resource on which to run the transformation by mapping each transformation's required computing capabilities to available computing resources as specified in the Pegasus resource pool configuration file. If Pegasus determines there is more than one computing resource available for use for a computational module, it selects one of the appropriate grid resources using a user selected algorithms. Supported resource selection algorithms include both round robin and random approaches.

During the conversion from abstract workflow to concrete workflow, Pegasus may augment a workflow by adding data or module staging into the workflow if the data and executable movements are required to execute on the selected computing resources. If, as Pegasus converts an abstract workflow to a concrete workflow, it recognizes that data movement is required by the workflow to run on the selected grid-resources, it will automatically add the data movement module into the workflow.

The automated workflow refinement phase in the SCEC/CME workflow system shields users from the complexities of the grid-based execution environment. Development of the resource descriptions such as the transformation catalog and the resource pool configuration can be performed by specialists knowledgeable about these environments and through Pegasus, deliver the flexible computing and data management capabilities of the Grid to non-expert users.

## 6. Grid-based Execution

Once the SCEC/CME system has produced a DAG, the workflow is ready for execution on a Globus-based grid. Grid-based execution is the final phase our workflow system.

As our workflow is now represented as an executable DAG, we submit the DAG to the Condor DAGMan workflow execution system. Since the Condor DAGMan interacts with Globus tools, a workflow constructed by our non-expert user now executes securely distributed across a wide variety of computing and storage devices accessible through the SCEC grid. Our grid-based workflows benefits from the job submission and scheduling capabilities of the Condor DAGMan tools. DAGMan analyzes a workflow for parallel elements and will run the computational steps in parallel where possible.

During grid-based execution, the SCEC/CME system provides workflow monitoring tools that allow users to follow the progress of their workflow as it executes on the grid. As a workflow executes and produces new data instances, the system assigns logical file names to the new files, and inserts logical file name to physical file name mappings into RLS. Modules output metadata which is written into MCS to maintain metadata for each new data instance.

At the completion of the workflow, the solution to the user's geophysical problem is stored in the system as a data file and an associated metadata description. This data instance is now available for use an input into the next geophysical problem that the scientist wishes to solve address using a workflow-based solution.

## 7. Case Study: Probabilistic Hazard Maps

As a discipline, explorations in geophysics are particularly well suited to be represented as scientific workflows. *Probabilistic seismic hazard analysis* (PSHA) hazard curve calculations are a good case in point. A PSHA hazard curve shows the probability that a specific site will exceed a specified amount of ground motion due to likely earthquakes over some period of time, for example, 50 years. A collection of PSHA hazard curves can be combined into probabilistic hazard maps.

The issues of component selection, configuration and computation management are complex even for members of the geophysics community. For a practicing engineer or city planner, the complexity can be overwhelming.

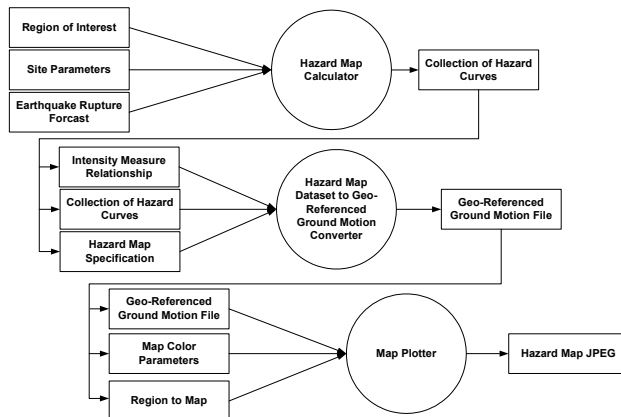
Representing PSHA calculation as a workflow mitigates many of these issues. Workflows help manage the complexity of the analysis in terms of the number of different types of calculations that must be performed, the number of alternatives for each processing step, and the number of data files that must be managed. Also, by enabling configuration and reuse of a pre-defined set of automatically configurable *workflow templates* we can make it possible to deliver this complex analysis to unsophisticated users, enabling them to generate PSHA maps on demand, customized to their use case requirements.

SCEC scientists utilized the SCEC/CME workflow system to perform a series of hazard map

calculations [9]. In these studies, a probabilistic seismic hazard map workflow was modeled in our system as a three transformation workflow. We will discuss this example to help illustrate the capabilities of our system.

In this example, the user wants to calculate a hazard map for the southern California region. The user begins in the workflow specification phase. To construct this workflow, the user begins the workflow specification using the CAT interface.

One way a non-sophisticated user may interact with CAT is to specify the target data type that they want to create. In this case the user wants to create a Hazard Map which is identified by the system as a *Hazard Map JPEG*. Once the user has identified this target data type, CAT is now able to suggest to the user which transformation are required to generate a files of this type. By interacting with CAT, and adding transformations into the workflow, the user develops a workflow-based solution to their problem. The workflow developed by the user with the assistance of the CAT is shown in Fig 2.



**Figure 2: Probabilistic Seismic Hazard Map calculation defined as a three step workflow.**

The workflow template produced by CAT indicates that each of the transformations requires two or more input data files. The figure shows that as the workflow executes output data sets will be used as inputs to subsequent transformations. However, some initial data sets must be supplied by the user in order to run this workflow. There are three user supplied initial inputs to the *Hazard Map Calculator*. The transformation called *Hazard Map Dataset to Geo-Referenced Ground Motion Converter* requires three inputs. One of these inputs will be calculated during the workflow. The other two inputs must be specified before the workflow can be run. The transformation called *Map Plotter* also requires three inputs, one of which is calculated during the workflow, and two of which must be specified prior to workflow execution.

Next, the user performs data discovery to find appropriate existing data files. For example, one input to the *Hazard Map Calculator* is a file of type *Region of Interest*. This file contains a description of

the geographical boundaries of the seismic hazard study. The user can search for a file with the desired boundaries using either the MCS-Search or DataFinder tools.

The user now begins the interactive workflow refinement. The WIT tool shows the user each transformation indicating the user supplied inputs and the calculated inputs. For each user-specified input file, WIT provides the user with a list of available files of the correct type. Once the user has selected a specific logical file name for each of the seven user-supplied, initial inputs to this workflow, the user submits the abstract workflow to the Pegasus system.

The Pegasus system now begins automated workflow refinement. The abstract workflow is converted to a concrete workflow. By making calls to the RLS system, each logical file name in the DAX is converted to a physical file name. The logical transformation names are converted to actual executable names. A hazard map calculation for the southern California region may require 10,000 hazard curve calculations and our workflow system must manage this number of files for each map.

Once Pegasus has generated a DAG, the DAG is submitted to the Condor DAGMan for job submission tools for execution on the grid. The data files created during the workflow execution are entered into the MCS and RLS.

When the workflow is submitted for execution, the system returns a logical file name identifying the desired final result of our workflow, which in this case is a *Hazard Map JPEG*. This file represents the solution to our geophysical problem, and it can be accessed as soon as the workflow completes.

## 8. Related Work

There has been increasing effort to enable scientists to create and manage complex scientific workflows. The myGrid project [21] exploits semantic web technology to support data intensive bioinformatics experiments in a grid environment. The semantic description of services in RDF and OWL is used for service discovery and match-making. They also provide interactive tools called Taverna for authoring and executing workflows. Kepler [1] is also a data-driven workflow system where the user can author data processing steps as a network of pre-defined workflow components called 'actors' and use 'directors' for describing execution models. The system allows semantic annotations of data and actors, and can support semantic transformation of data. Triana [3] allows the user to specify execution behavior easily by supporting an abstract layer for Grid computing called GAP, a subset of the GridLab GAT. Using a graphical interface, the user can drag and drop workflow component to form a workflow and also specify distribution policy for a group of nodes in the workflow. GAP services can be advertised, discovered and communicated with using abstract high level calls on Grids and P2P networks. Our work presents complementary capabilities in that the workflow composition tools in these systems provide limited assistance in validating user authored

workflows and providing suggestions to generate complete and consistent workflows. Also these systems do not employ sophisticated resource management approaches and cannot fully make use of grid resources.

There are many Grid tools that are developed to help end-users manage complex workflows. ICENI (Imperial College e-Science Network Infrastructure) [17] provides a component based Grid middleware. Users can construct an abstract workflow from a set of workflow components and the system generates a concrete workflow using its scheduler. ICENI also support different views of composed workflows: it uses spatial view to allow flexibility during composition and provides temporal view to support scheduling optimization. GridAnt [20] developed by Argonne National Laboratory is a client-side workflow system that assists users to express and control execution sequences and test Grid services. It uses Java Cog Kit that provides access to Grid services through a Java framework. GridFlow [2] provides user portal and services of both global grid workflow management and local grid sub-workflow scheduling. Unicore [19] provides a programming environment where users can design and execute job flows with advanced flow controls. Our workflow system presents a more comprehensive grid-based environment for scientific workflows and we believe that integrating our approach with these capabilities may result in improved environments to support complex scientific workflows.

## 9. Discussion

The SCEC/CME system helps geoscientist execute a wide variety of geophysical simulation codes including serial Probabilistic Seismic Hazard calculations, MPI-based high performance earthquake wave propagation simulations, and I/O intensive volumetric data post-processing software.

The SCEC/CME Workflow system represents a comprehensive grid-based workflow environment that addresses a full range of workflow capabilities from interactive composition of workflows, to data and metadata management, semantic metadata search capabilities, and job scheduling that makes full use of grid resources. The system's use of well-established workflow tools including Globus and VDS enhances its stability, robustness, and maintainability. Through its support for grid environment, the system supports workflows that run from simple Condor Pool-based applications to high performance computing system on the TeraGrid.

The SCEC/CME system utilizes standard semantic web technologies including OWL, SOAP, and WSDL to enhance platform independence and interoperability. The system supports a wide variety of computational modules, including, but not limited to web service invocations. This support for a variety of computational module types eases the integration of existing programs into the workflow system.

During this work, we simplified user interactions with the system by introducing intelligent assistants and knowledge driven resource selections. These tools utilize a variety of data stores including

computing and domain ontologies, as well as metadata, computational module, and computer resource descriptions. It is through these descriptive, rather than computational, data stores, that we are able to provide user assistance, and to simplify the user experience.

In our view, the development and use of these knowledge repositories shifts the burden of detailed knowledge of workflows, in an appropriate way, towards our domain and computing experts and away from end users of our workflow system. By capturing the knowledge of domain and computing experts in knowledge bases, and by constructing our tools to use this captured knowledge, we reduced the expertise required to use our scientific workflow system making the system more accessible to non-expert users.

## Acknowledgements

This work was support by the SCEC Community Modeling Environment Project which is funded by the National Science Foundation (NSF) under contract EAR-0122464 (The SCEC Community Modeling Environment (SCEC/CME): An Information Infrastructure for System-Level Earthquake Research). This work was also supported by the NSF GriPhyN Project, grant ITR-800864. SCEC is funded by NSF Cooperative Agreement EAR-0106924 and USGS Cooperative Agreement 02HQAG0008. The SCEC contribution number for this paper is 915.

## 10. References

- [1] Altintas, I., C. Berkley, E. Jaeger, M. Jones, B. Ludäscher, S. Mock, Kepler: Towards a Grid-Enabled System for Scientific Workflows, In the Workflow in Grid Systems Workshop in GGF10 - The Tenth Global Grid Forum, Berlin, Germany, March 2004.
- [2] Cao, J., S. A. Jarvis, S. Saini, and G. R. Nudd. GridFlow: Workflow Management for Grid Computing. In 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2003), Tokyo, Japan, May 2003.
- [3] Churches, D., G. Gombas, A. Harrison, J. Maassen, C. Robinson, M. Shields, I. Taylor and I. Wang, Programming Scientific and Distributed Workflow with Triana Services. In Grid Workflow 2004 Special Issue of Concurrency and Computation: Practice and Experience, to be published, 2005.
- [4] Deelman, E., James Blythe, Yolanda Gil, Carl Kesselman, Scott Koranda, Albert Lazzarini, Gaurang Mehta, Maria Alessandra Papa, Karan Vahi (2003a). Pegasus and the Pulsar Search: From Metadata to Execution on the Grid, Applications Grid Workshop, PPAM 2003, Czestochowa, Poland 2003.
- [5] Deelman E., James Blythe, Yolanda Gil, Carl Kesselman (2003b). Workflow Management in GriPhyN, Grid Resource Management, Kluwer, 2003.
- [6] Deelman, E., James Blythe, Yolanda Gil, Carl Kesselman, Gaurang Mehta, Karan Vahi, Kent Blackburn, Albert Lazzarini, Adam Arbre, Richard Cavanaugh, and Scott Koranda (2003c). Mapping

Abstract Complex Workflows onto Grid Environments, *Journal of Grid Computing*, Vol.1, no. 1, 2003, pp. 25-39.

[7] Deelman, E., James Blythe, Yolanda Gil, Carl Kesselman, Gaurang Mehta, Sonal Patil, Mei-Hui Su, Karan Vahi, Miron Livny (2004), Pegasus : Mapping Scientific Workflows onto the Grid, Across Grids Conference 2004, Nicosia, Cyprus, 2004

[8] Domenico B., John Caron, Ethan Davis, Robb Kambic and Stefano Nativi (2002). Thematic Real-time Environmental Distributed Data Services (THREDDS): Incorporating Interactive Analysis Tools into NSDL, *Journal of Digital Information*, Volume 2 Issue 4 Article No. 114, 2002-05-29

[9] Field, E.H., V. Gupta, N. Gupta, P. Maechling, and T.H Jordan (2005). Hazard Map Calculations Using GRID Computing, to be published in *Seismological Research Letters*, 2005

[10] Foster, I., C. Kesselman, S. Tuecke, The Anatomy of the Grid: Enabling Scalable Virtual Organizations, *International J. Supercomputer Applications*, 15(3), 2001.

[11] Foster, I., J. Voeckler, M. Wilde, and Y. Zhao. Chimera: A Virtual Data System for Representing, Querying and Automating Data Derivation Proceedings of the 14th Conference on Scientific and Statistical Database Management, Edinburgh, Scotland, July 2002.

[12] Frey, J. T. Tannenbaum, I. Foster, M. Livny, S. Tuecke, Condor-G: A Computation Management Agent for Multi-Institutional Grids, *Cluster Computing*, 5(3):237-246, 2002.

[13] Genesereth, M.R. (1991). Knowledge interchange format. In J. Allen, R. Fikes, and E. Sandewall, editors, Proceedings of the 2<sup>nd</sup> International Conference on Principles of Knowledge Representation and Reasoning, pages 599--600, San Mateo, CA, USA, April 1991.

[14] Jordan, T. H., P. J Maechling, and the SCEC/CME Collaboration (2003). The SCEC community modeling environment: an information infrastructure for system-level science, *Seism. Res. Lett.* 74, 324-328

[15] Kim, J., M. Spraragen, and Y. Gil (2004a), An Intelligent Assistant for Interactive Workflow Composition, In Proceedings of the International Conference on Intelligent User Interfaces (IUI-2004); Madeira, Portugal, 2004.

[16] Kim, J., M. Spraragen, and Y. Gil (2004b), A Knowledge-Based Approach to Interactive Workflow Composition, In Proceedings of the 2004 Workshop on Planning and Scheduling for Web and Grid Services, at the 14th International Conference on Automatic Planning and Scheduling (ICAPS 04); Whistler, Canada, 2004.

[17] McGough, S., L. Young, A. Afzal, S. Newhouse, and J. Darlington, WorkflowEnactment in ICENI In UK e-Science, All Hands Meeting, p. 894--900, Nottingham, UK, Sep. 2004

[18] PowerLoom:  
<http://www.isi.edu/isd/LOOM/PowerLoom/>

[19] Romberg, R., The UNICORE Architecture-Seamless Access to Distributed Resources, Proceedings of 8th IEEE International Symposium on

High Performance Computing, Redondo Beach, CA, USA, August 1999, pp. 287-293

[20] von Laszewski, G. K. Amin, M. Hategan, N. J. Zaluzec, S. Hampton, and A. Rossi, GridAnt: A Client-Controllable Grid Workflow System. In 37th Annual Hawaii International Conference on System Sciences (HICSS'04) IEEE Computer Society Press, Los Alamitos, CA, USA, January 5-8, 2004.

[21] Wroe C., C.A. Goble, M. Greenwood, P. Lord, S. Miles, J. Papay, T. Payne, L. Moreau (2004), Automating Experiments Using Semantic Data on a Bioinformatics Grid, In IEEE Intelligent Systems special issue on e-Science Jan/Feb 2004.