Institute of Physics Publishing Journal of Physics: Conference Series **46** (2006) 468–478
doi:10.1088/1742-6596/46/1/065 SciDAC 2006

Accelerating the scientific exploration process with scientific workflows

Ilkay Altintas[1], Oscar Barney[2], Zhengang Cheng[3], Terence Critchlow[4], Bertram Ludaescher[5], Steve Parker[2], Arie Shoshani[6], Mladen Vouk[3]

[1] San Diego Supercomputer Center, University of California San Diego
[2] Scientific Computing and Imaging Institute, University of Utah
[3] Department of Computer Science, North Carolina State University
[4] Center for Applied Scientific Computing, Lawrence Livermore National Laboratory
[5] Department of Computer Science, University of California Davis
[6] Computing Research Division, Lawrence Berkeley National Laboratory

Email: altintas@sdsc.edu, oscar@sci.utah.edu, zcheng@ncsu.edu, critchlow@llnl.gov, ludaesch@ucdavis.edu, sparker@cs.utah.edu, shoshani@lbl.gov, vouk@ncsu.edu

Abstract Although an increasing amount of middleware has emerged in the last few years to achieve remote data access, distributed job execution, and data management, orchestrating these technologies with minimal overhead still remains a difficult task for scientists. Scientific workflow systems improve this situation by creating interfaces to a variety of technologies and automating the execution and monitoring of the workflows. Workflow systems provide domain-independent customizable interfaces and tools that combine different tools and technologies along with efficient methods for using them. As simulations and experiments move into the petascale regime, the orchestration of long running data and compute intensive tasks is becoming a major requirement for the successful steering and completion of scientific investigations.

A scientific workflow is the process of combining data and processes into a configurable, structured set of steps that implement semi-automated computational solutions of a scientific problem. Kepler is a cross-project collaboration, co-founded by the SciDAC Scientific Data Management (SDM) Center, whose purpose is to develop a domain-independent scientific workflow system. It provides a workflow environment in which scientists design and execute scientific workflows by specifying the desired sequence of computational actions and the appropriate data flow, including required data transformations, between these steps. Currently deployed workflows range from local analytical pipelines to distributed, high–performance and high-throughput applications, which can be both data- and compute-intensive. The scientific workflow approach offers a number of advantages over traditional scripting-based approaches, including ease of configuration, improved reusability and maintenance of workflows and components (called actors), automated provenance management, "smart" re-running of different versions of workflow instances, on-the-fly updateable parameters, monitoring of long running tasks, and support for fault-tolerance and recovery from failures.

We present an overview of common scientific workflow requirements and their associated features which are lacking in current state-of-the-art workflow management systems. We then illustrate features of the Kepler workflow system, both from a user's and a "workflow engineer's" point-of-view. In particular, we highlight the use of some of the current features of Kepler in several scientific applications, as well as upcoming extensions and improvements that are geared specifically for SciDAC user communities.

1. Expanding Technological Horizons in Science

"Why does this magnificent applied science, which saves work and makes life easier, bring us so little happiness? The simple answer runs: Because we have not yet learned to make sensible use of it."
– Albert Einstein, in an address at Cal Tech, 1931. (Harper)

Scientific problem solving is an evolving process. Scientists start with a set of questions then observe phenomenon, gather data, develop hypotheses, perform tests, negate or modify hypotheses, reiterate the process with various data, and finally come up with a new set of questions, theories, or laws. This flow of science has been continuously transformed with the advances in computer science in the last few decades. The simplest examples of this transformation are the use of personal computers to record scientific activity and the way scientists publish and search papers on online databases. This push from computer science combined with evolving scientific applications and algorithms, resulted in technologies for making the automation of the scientific process more efficient and easier to use.

Current technology significantly accelerates the scientific problem solving process by allowing scientists to access data remotely, distribute job execution across remote parallel resources, efficiently manage data, and even perform observations in remote environments via technologies like sensor networks and cameras. As simulations and experiments move into the petascale regime, the orchestration of long running data and compute intensive tasks is becoming a major requirement for the successful steering and completion of scientific investigations. Although an increasing amount of middleware has emerged in the last few years to achieve remote data access, distributed job execution, and data flow, the orchestration of these technologies with minimal overhead still remains a difficult task for scientists. Scientific workflow systems improve this situation by creating interfaces to a variety of technologies and automating the execution and monitoring of the workflows [1,2,3]. Workflow systems provide domain-independent customizable interfaces and tools that combine different technologies along with efficient methods for using them.

The Scientific Data Management (SDM) Center [4], funded under the first phase of SciDAC (referred to as SciDAC-1), has made a large impact on the SciDAC-1 domain scientists and the community by providing advanced data management technologies. The center's focus is on deploying known and emerging data management technologies to scientific applications with a goal of integrating software-based solutions to efficiently and effectively manage large volumes of data generated by scientific applications. Previous achievements include the development and application of parallel I/O technology applied to astrophysics and climate, efficient indexing of large datasets applied to high energy physics and combustion, features analysis applied to fusion simulation, parallelization of the popular package R, and the automation of production-level biology and astrophysics workflows using a scientific workflow system. The Scientific Process Automation (SPA) [5] layer in the SDM Center architecture builds upon the SDM Center tools and technologies to utilize them in scientific workflows. SDM SPA activities help a broad range of scientific communities in collaboration with the open-source Kepler scientific workflow system [6].

In the rest of this paper, we first discuss a couple of workflow examples and the requirements for workflows to help with the scientific process. We then give a brief overview of the Kepler scientific workflow system and introduce some components in the Kepler architecture influenced by the DOE application requirements. Finally, we describe our vision of the additional requirements for next generation scientific workflow systems to help with accelerating the scientific exploration process.

2. Scientific Workflows Help with the Scientific Process

A scientific workflow is the process of combining data and processes into a configurable, structured set of steps that implement semi-automated computational solutions of a scientific problem. Workflows greatly improve data analysis, especially when data is obtained from multiple sources and generated by computations on distributed resources and/or various analysis tools. These advances in

systematic analysis of scientific information made possible by workflows have unleashed a growing need for automated data-driven applications that also provide scientists with interfaces to design, create, execute, share, reuse scientific workflows, and collect and manage the provenance of the data and processes with little overhead. The scientific workflow approach offers a number of advantages over traditional scripting-based approaches, including ease of configuration, improved reusability and maintenance of workflows and components (actors), automated provenance management, "smart" re-running of different versions of workflow instances, on-the-fly updateable parameters, monitoring of long running tasks, and support for fault-tolerance and recovery from failures [1].

2.1. Scientific Workflow Examples

In this section, we provide two different typical high-level scientific knowledge discovery workflows that we have worked on in the SDM Center. The workflows have requirements that relate to some of the generic requirements mentioned above.

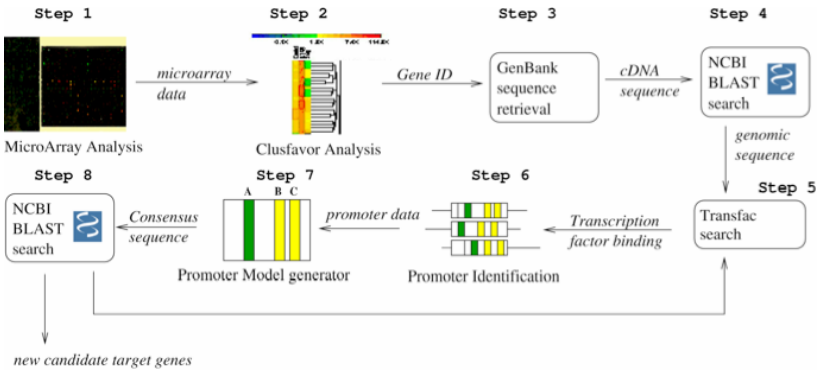

Figure 1. Conceptual design of the "Promoter Identification Workflow" [7]

2.1.1. *Promoter Identification Workflow.* Figure 1 shows the "Promoter Identification Workflow" (PIW) that links genomic biology techniques such as microarrays with bioinformatics tools such as BLAST to identify and characterize eukaryotic promoters [1, 7]. Starting from microarray data, cluster analysis algorithms are used to identify genes that share similar patterns of gene expression profiles which are then predicted to be co-regulated as part of an interactive biochemical pathway. Given the gene-ids, gene sequences are retrieved from a remote database (e.g., GenBank) and fed to a tool (e.g., BLAST) that finds similar sequences. In subsequent steps, transcription factor binding sites and promoters are identified to create a promoter model that can be iteratively refined.

While Figure 1 leaves many details out, some features of scientific workflows can already be identified. Specifically, there are a number of existing databases (such as GenBank) and computational tools (such as Clusfavor and BLAST) that need to be combined in certain ways to create the desired workflow. In the past, accessing remote resources often meant implementing a wrapper that mimics a human entering the input of interest, submitting an HTML form, and "screenscraping" the result from the returned page [8]. Today, more and more tools and databases become accessible via web services, greatly simplifying this task. Another trend are web portals such as NCBI [9] that integrate many tools and databases and sometimes provide the scientist with a "workbench" environment.

This PIW workflow was implemented in the Kepler scientific workflow system (see Section 3). The early prototype PIW implementation in Kepler [1, 6, 7] illustrated a number of features including the hierarchical workflow design, Web Service composition, and seamless integration of resources via customization of features. Currently, this workflow runs as a production workflow. The production workflow extends the above-mentioned proven features with execution monitoring, fault-tolerance and domain-specific visualization components, e.g., a transcription factor alignment display.

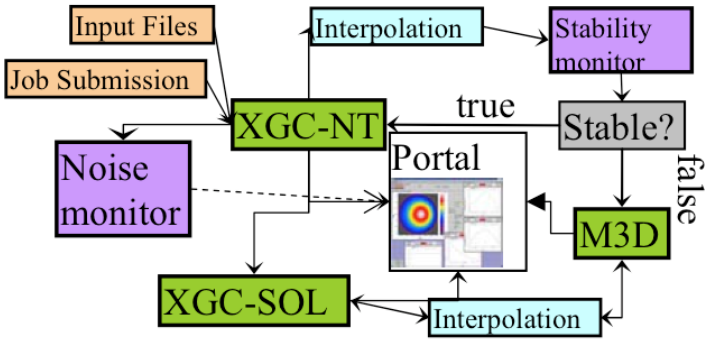

Figure 2. Conceptual design of "Center for Plasma Edge Simulation XGC-M3D Workflow" [10]

2.1.2. *The Center for Plasma Edge Simulation XGC-M3D Workflow.* The Center for Plasma Edge Simulation (CPES) is a Fusion Simulation Project under the DOE SciDAC program to develop a novel integrated plasma edge simulation framework. The CPES project aims to integrate and utilize the existing fusion codes via a scientific workflow system. Within CPES a set of specialized high-performance computing (HPC) extensions coupled with fusion codes and a monitoring system is used to explore, analyze and visualize data, and extract information and features. [10]

Figure 2 illustrates one of the workflows, i.e. "XGC-M3D Coupling Workflow", which is under development in the CPES project using Kepler. It shows the coupling of the kinetic code, *XGC—NT*, to the MHD code, *M3D*, which is then deployed on a Portal interface for user interaction. An important point is that some fusion codes run on thousands of processors, and send each other 2D-3D variable with a requirement to exchange information in less then a second to keep the codes synchronized.

Other CPES workflows have requirements like dynamic coupling of models and codes, adaptive workflow execution, transparent and efficient access to data, data analysis and visualization, and run time monitoring with interactive and autonomic control. For the success of projects with requirements similar to CPES's, the workflow system is expected to handle memory-to-memory data movement between the individual processors and efficient, local, and on-the-fly inter-processor interaction.

2.2. Requirements for Scientific Workflows

Scientific workflows vary in their requirements and the technologies they use depending on the disciplines involved. We found that even though the development in scientific workflows is motivated by applications from different scientific disciplines and projects, and the disciplines have their own wish lists and technical requirements, it is possible to unify these requirements and come up with a set of common requirements useful for multiple disciplines. Consider, for example, the Cyberinfrastructure desiderata of a geoscientist and a computational chemist. A geoscientist's wish list might involve online data acquisition and access, managing large databases, indexing data on spatial and temporal attributes, quick subsetting operations on the databases, large scale resource sharing and management, collaborative and distributed applications, parallel gridding algorithms on large data sets using high performance computing, integrating data with other related data sets, e.g., geologic maps and hydrology models, and easy-to-use user interfaces from portals and scientific workflow environments [11]. A computational chemist's wish list is comprised of performing synchronously many embarrassingly parallel calculations with different molecules, configurations, methods and/or other parameters, using existing data, codes, procedures, and resources with the usual sophistication and accuracy, letting different programs seamlessly communicate with each other, integrating computational experiment preparation and analysis steps, distributing jobs automatically onto a computational grid, having interfaces to database infrastructures to obtain starting structures and to allow data mining of results, and tools and scientific workflow environments that are as easy-to-use, general, flexible, manageable and reusable as possible [12, 13]. While these requirements have different focuses, their fundamental requirements are similar, i.e., querying and integrating data sets,

execution of domain specific codes and algorithms, and using different tools through easy-to-use unified interfaces. When we extend this to other sciences, we come up with a list of requirements that are generically applicable to different disciplines via customization and domain-knowledge integration.

The following requirements are the results of our interaction with a variety of scientists from scientific disciplines involving molecular biology, ecology, astrophysics, fusion, geosciences, oceanography, computational chemistry and phylogeny:

- Design tools that are scalable and especially helpful to non-technical users
- Easy to use, fairly simple user interfaces having more complex features hidden
- Hierarchical component and service composition [14] to reduce complexity and enhance design reusability
- Reusable generic features that are generic enough to serve different communities but specific enough to serve one scientific domain, i.e., customizable
- Extensibility for the expert user
- Registration, publication & provenance of data and process (a.k.a. workflow) products [15]
- Seamless access to and dynamic plug-in of data and processes from registries/repositories [16]
- Distributed [17] and detached [18] workflow execution (e.g. Web and Grid awareness)
- Semantics awareness [19] using ontologies and domain knowledge
- Execution monitoring, failure recovery, smart re-runs [1, 15] and interactive user steering
- Workflow deployment tools (as a web site, as a web service, as a "Power app" similar to SCIRun [20]) that could be utilized via the user interface

3. Kepler: An Emerging Open Source Scientific Workflow System

A scientific workflow is the process of combining data and processes into a configurable, structured set of steps that implement semi-automated computational solutions of a scientific problem. Kepler is a cross-project collaboration, co-founded by the SciDAC SDM Center, whose purpose is to develop a domain-independent scientific workflow system. It provides a workflow environment in which scientists design and execute scientific workflows by specifying the desired sequence of computational actions and the appropriate data flow, including the required data transformations, between these steps. Currently deployed workflows range from local analytical pipelines to distributed, high–performance and high-throughput applications, which can be both data- and compute-intensive.

Kepler builds on top of the mature Ptolemy II software [21], which is a Java-based system and a set of APIs for heterogeneous hierarchical modeling. The focus of Ptolemy II is to build models based on the composition of existing components, which are called 'Actors', and observe the behavior of these simulation models when executed using different computational semantics, which are implemented as components called 'Directors.'

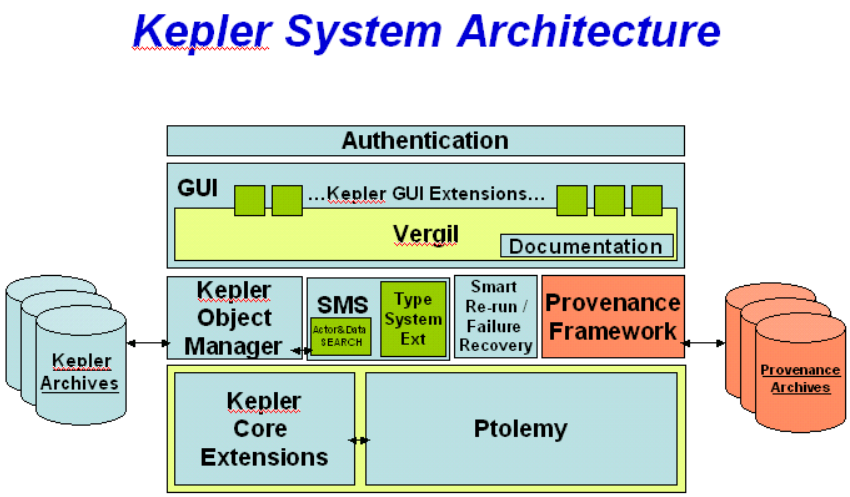

Figure 3. The Kepler System Architecture

Actors are the encapsulations of parameterized actions performed on input data to produce output data. Actors communicate between themselves by sending Tokens, which encapsulate data or messages, to other actors through ports. An actor can have multiple ports and can only send Tokens to an actor that it is connected to one of its output ports. The director specifies the model of computation under which the workflow will run. For example, in a workflow with a Process Network (PN) director, actors can be thought of as separate threads that asynchronously consume inputs and produce outputs. Under the Synchronous Dataflow (SDF) director, actors share a common thread, and the order of execution is statically determined because the number of tokens each actor will produce and consume is known ahead of time. Also, different domains control how the ports relay Tokens. In PN each port behaves like a FIFO queue of unlimited size where as a port controlled by the SDF director acts like a FIFO queue with a size limited to the number of tokens an actor can produce or consume.

Kepler actors perform operations including data access, process execution, visualization, and domain specific functions. Kepler uses Ptolemy II's hierarchical actor oriented modeling paradigm to create workflows, where each step is performing some action on a piece of data. Workflows can be organized visually into sub-workflows. Each sub-workflow encapsulates a set of executable steps that conceptually represents a separate unit of work. The Kepler system can support different types of workflows ranging from local analytical pipelines to distributed, high–performance and high-throughput applications, which can be data- and compute-intensive [17]. Along with the workflow design and execution features, Kepler has ongoing research on a number of built-in system functionalities (a.k.a. architectural components), as illustrated in Figure 3, including support for single sign-in GSI-based authentication and authorization; automated data and process provenance tracking; "smart" re-runs and failure recovery; semantic annotation of actors, types, and workflows; creating, publishing, and loading plug-ins as archives using the Vergil user interface; and documenting entities of different granularities on-the-fly.

Ptolemy II separates the description of important aspects of a design such as behavior and architecture, or computation and communication [22]. Kepler inherits this concept of separation of concepts in design from the Ptolemy II. This provides significant advantages such as lower design time and better reusability of the design because system designers can build new components for the system and plug them in for testing without changing any of the underlying architecture. Also, workflow designers do not have to use ad hoc techniques to implement the workflow's design and execution of the workflow graph. The Ptolemy II system provides a general strategy for separating the workflow composition from the overall orchestration of the model by introducing the separate concerns for actors, their composition, and the implementation of computational domains that run the

workflows. These 'separate concerns' are combined visually into a model on the screen, which provides an easy way for system users to see what the exact behavior of the workflow will be without clicking on menu to find out things like what the model of computation will be, for example.

3.1. Kepler Provenance Framework

Most of the time before a scientific process can end in results, scientists will fine-tune the experiments, iterating many times with different parameters [23]. This repeating process can reveal a lot about the nature of a specific scientific problem and, thus, provide information on the steps leading to the solution and end results. Making this information available requires efficient usage and collection of data and process provenance information. Another very important requirement for any scientific process is the ability to reproduce results and to validate the process that was followed to generate these results. Provenance tracking provides this functionality and also helps the user and publication reviewer/reader understand how the run progressed and what parameters and inputs were associated with the workflow run [15].

A provenance collection system must have the ability to create and maintain associations between workflow inputs, workflow outputs, workflow definitions, and intermediate data products. Collecting intermediate data products in addition to other provenance data serves multiple purposes as the results of some processes in the workflow can vary from run to run and some workflow tests require manually stepping through some of the steps to verify and debug results. The intermediate results along with the other provenance data can be also used to perform "smart" reruns. These requirements illustrate the strong need to retain origin and derivation information for data and processes in a workflow, to associate results with customized and executable workflow version, and to track workflow evolution as described in [24]. This paper discusses an implementation that aims to keep track of all these aspects of provenance in scientific workflows.

Given the collaborative and domain-independent nature of the Kepler project, our provenance framework needs to include plug-in interfaces for new data models, metadata formats and cache destinations. To accomplish this goal we have created a highly configurable provenance component that can be easily used with different models of computation using the separation of concerns design principle. Just like a director, provenance collection is modeled as a separate concern that is bound visually to the associated workflow. This way a user can easily see if provenance is being collected for a certain run of the workflow. Another advantage to this design is its compliance and consistency with Kepler's actor-oriented programming paradigm and user interface.

In Kepler, we have added functionality to enable efficient reruns of a workflow by mining the stored provenance data. The idea behind a "smart" rerun [1] is as follows. When a user changes a parameter of an actor and runs the workflow again, re-executing all the preceding, unchanged steps in the workflow may be redundant and time consuming. A "smart" rerun of the workflow will take data dependencies into account and only execute those parts of the workflow affected by the parameter change [25]. The ability to store and mine provenance data is required to enable "smart" reruns since the data products generated in previous runs are used as the inputs to the actors that are to be rerun.

3.2. Kepler Authentication Framework

The recent advances in Grid and distributed computation enables automated data analyses to be performed in heterogeneous and distributed environments, helping scientists to perform scientific exploration at scales both finer and greater than ever before. It is no longer sufficient for current information systems to focus only on individual tasks. The availability of efficient data collection and analysis tools presents researchers with vast opportunities to process heterogeneous data within a distributed environment. More and more applications require computer systems to control, monitor, and support the logistical aspects of scientific processes. To support the opportunities enabled by available massive computation, the Kepler project provides scientists an open-source scientific workflow system to manage both data and programs, and to design reusable procedures of scientific experimental tasks. The distributed environments may spread over multiple computers, clusters, and networks or even multiple Grid systems. Most Grid systems employ Grid Security Infrastructure (GSI)

[26] to manage the authentication and authorization of users to access resources. Thus, an authentication framework is needed for workflows to manage the GSI-related accessibility to multiple computing resources. This framework should be able to coordinate between the different security architectures. With more workflows executed on multiple secured computing resources, Kepler users need a mechanism to coordinate between different security infrastructures, and to generate and manage the certificates/proxies for the user. Kepler authentication framework is such a management component that not only facilitates users to login into various computing resources and store their corresponding authentications, but also provides workflows a unified interface to access remote resources with the support of GSI.

Workflows may not only access multiple Grid resources that belong to different security systems, but also access the same Grid resource multiple times during their execution. A workflow needs the authentication information at each time it accesses a Grid resource. It is not practical to prompt users for login whenever the authentication is required, as most of the time, these workflows run in batch mode while the user is away. The authentication framework should be able to store user's authentication in order to avoid user interventions for repeatedly used proxies and manage the lifetime and updates of the stored authentication.

Kepler's authentication framework is designed and implemented for storing, managing, and updating user authentications for workflows running in Grid environments. This framework is integrated into the Kepler project to coordinate the accesses to multiple Grid resources for the processes (a.k.a actors in Kepler) in the workflow. It not only provides workflows a uniform interface to access user authentications, but also permits users to create and manage their proxy certificates and authentications. The authentication framework is a novel contribution to Kepler and, to the best of our knowledge, no other scientific workflow system currently has such a mechanism.

3.3. Distribute Execution and Flexible I/O Services in Kepler

In Kepler, individual workflow components (e.g., for data movement, database querying, job scheduling, remote execution etc.) are abstracted into a set of generic, reusable tasks [17]. Through actor-oriented and hierarchical modeling features built into Ptolemy, Kepler scientific workflows can operate at very different levels of granularity, from low-level "plumbing workflows" that explicitly move data around, start and monitor remote jobs, etc., to high-level "conceptual workflows" that interlink complex, domain specific data analysis steps. Grid workflows (and distributed applications in general) can be seen as special scientific workflows. They involve high performance and/or high throughput computational tasks. One of our targets to run distributed applications in Kepler is a framework that supports the design and reuse of grid workflows.

Our goals for a distributed Kepler include also the ability to easily form ad-hoc networks of cooperating Kepler instances. Each Kepler cooperating network (collaboration) can have access constraints and allows Kepler models or sub-models to be run on participating instances. Once a Kepler cooperating network has been created, it can configure one or more subcomponents of a workflow to be distributed across nodes of the newly constructed network (Grid).

One of the more significant challenges for Grid computing remains the difficulty in developing software that is both concurrent and distributed. Whilst there have been many proposals over the years on how to build such systems, including very recent work in Web services, these are not sufficient. Existing Grid workflow tools assume that individual components either communicate by passing files from one application to another, or are explicitly linked using interprocess communication pipes. When files are used, it is usually difficult to overlap read and write execution because the file is only copied from one computation to the next once it has been completed and closed. This makes it difficult to develop deeply pipelined computations in which input is consumed at the same time as it is produced. So, pipelined computations usually require the use of interprocess communication primitives such as TCP/IP pipes. However, these are usually invasive and require substantial modification to existing code, and, once coded, it is difficult to revert to use a more conventional file-based IO mechanism.

GriddLeS [27] is a library that provides a rich set of interprocess communication facilities. Such facilities can be used to implement data sharing in a Grid workflow. The core concept in GriddLeS is GridFiles, a device which enables file based interprocess communication between software components. GriddLeS overloads conventional file IO operations, and support either local, remote or replicated file access, or direct communication over pipes. Thus, the program thinks it is reading or writing a file, but GriddLeS may choose an alternative transport at run time. Together, Kepler and GriddLeS, provide a powerful implementation of Grid workflows [16]. On its own, Kepler provides functionality to construct, edit and execute workflows using SDF or PN execution models. However, Kepler requires substantial modifications to the underlying applications in moving from one mode to another, making it difficult to tune the performance of the overall system. On the other hand, GriddLeS offers the ability to delay the choice of inter-process data transfer mechanism until run time. Since no application changes are required, a user can optimize and tune the performance of the system at run time using the same Kepler interface that they used to build the original workflow.

3.4. Using Kepler as an Execution Engine

Kepler [1, 6] workflow system can be used as a comprehensive environment for coordinating distributed resources using emerging Grid technologies, and the Grid infrastructure, to provide efficient and reliable data analysis. In order to accomplish this, Kepler was utilized in a three tier architecture: the portal layer, the workflow layer or control layer, which is the Kepler workflow system, and the Grid layer as the computation layer.

The portal layer, a portlet, serves as a front end user interface. It enables the user to provide the data and choose from the algorithms, processing attributes and desired derivative products via a web interface. Within Kepler, one can design a predefined modularized and configurable workflow template using place holder stubs to be set prior to the workflow execution. A workflow pattern serves as a conceptual workflow template, defined in the Kepler workflow description language, MoML. A workflow instance is created on the fly from the conceptual workflow based on the user selections. The instantiated workflow is then scheduled to be executed by the workflow layer [18].

The workflow layer, also referred to as the main control layer, communicates both with the portal and the Grid layers. This layer, controlled by the Kepler workflow manager, coordinates the multiple distributed Grid components in a single environment as a data analysis pipeline. It submits and monitors jobs onto the Grid, and handles third party transfer of derived intermediate products among consecutive compute clusters, as defined by the workflow description. In addition, it sends control information (a.k.a. tokens) to the portal client about the overall execution of the process. The workflow is executed by the Kepler engine in a batch mode. Once a job is submitted, the user can detach from the system and receive an email notification after the process has completed.

The Grid layer, or the execution layer is where the actual processing implementations are deployed on the distributed computational Grids. Currently a simple submission and queuing algorithm is used for mapping jobs between various resources based on the number of allocated tasks and the size of the data to be processed. In the near future we plan to utilize the Pegasus [29] Grid scheduler to benefit from mapping jobs based on resource efficiency and load, thus making the process more robust.

4. Outlook and Conclusions

The DOE SDM Center and the Kepler scientific workflow system have made a big impact working with both DOE scientists and other national and international scientist. In this paper, we have provided an overview of scientific workflow management issues, motivated by real-world examples that we encountered in a number of application projects. We identified a number of common requirements and desiderata of scientific workflows. We observed an extension in these requirements as the technology is being used and evolves. The combined list of all the requirements that a multi-disciplinary scientific workflow system today needs to provide includes: secure and seamless access to heterogeneous data and computational resources and multiple virtual organizations using multiple roles; efficiently connect to the existing data and integrate heterogeneous data from multiple resources; offer extensible and customizable graphical user interfaces for scientists from different scientific domains; link to

different domain knowledge and interface to multiple applications and analysis tools; support computational experiment creation, execution, sharing and reuse; manage complexity, user and process interactivity; provide extensions for adaptive and dynamic workflows; track provenance of workflow design evolution, execution, and intermediate and final results; perform efficient failure recovery and smart re-runs; support various file and process transport mechanisms (e.g., main memory and Java shared file system); support the full scientific process by means to use and control instruments, networks and observatories in observing steps; scientifically and statistically analyze and control the data collected by the observing steps; set up simulations as testbeds for possible observatories; come up with efficient and intuitive workflow deployment methods; and most importantly do all these in a secure and usable way.

To support the above-mentioned extended requirements, Kepler and the rest of the scientific workflow research community faces challenges in areas including usability, workflow design methodologies, provenance tracking, new computational models to support dynamic and adaptive changes in the running workflows, failure recovery, re-running workflows with minor changes, authentication and authorization, and incorporation of semantic Grid and semantic annotation technologies. Among these challenges, usability and provenance tracking requires special attention, as these features have been worked on less in the previous years, and are vital for the acceptance of the technology by wider scientific communities and for the success of all the other features in a scientific workflow.

5. Acknowledgements

The authors would like to thank the DOE scientists working with the SDM Center, especially to Matthew Coleman, Doug Swesty, Eric Myra, John Blondin and Scott Klasky, and the SDSC SWAT Lab team and the rest of the Kepler team for their excellent collaboration. This work is supported by DOE SciDAC DE-FC02-01ER25486 (SDM), NSF/ITR 0225676 (SEEK), NSF/DBI-0078296 (Resurgence) and NSF/ SCI-0438741 (Delivering Cyberinfrastructure: From Vision to Reality).

References

[1] Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger-Frank, E., Jones, M., Lee, E., Tao, J., Zhao, Y.: Scientific Workflow Management and the Kepler System. Concurrency and Computation: Practice & Experience, Special Issue on Scientific Workflows, to appear, 2005. http://kepler-project.org/

[2] Oinn, T., Greenwood, M., Addis, M., Alpdemir, M.N., Ferris, J., Glover, K., Goble, C., Goderis, A., Hull, D., Marvin, D., Li, P., Lord, P., Pocock, M.R., Senger, M., Stevens, R., Wipat, A. Wroe, C.: Taverna: Lessons in creating a workflow environment for the life sci-ences". Accepted for publication in Concurrency and Computation: Practice and Experience Grid Workflow Special Issue

[3] Churches, D., Gombas, G., Harrison, A., Maassen, J., Robinson, C., Shields, M., Taylor, I., Wang, I.: 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] Scientific Data Management Center. http://sdmcenter.lbl.gov

[5] Scientific Process Automation. http://www-casc.llnl.gov/sdm/

[6] Kepler Scientific Workflow System and Project. http://kepler-project.org, 2006.

[7] Altintas, I., Bhagwanani, S., Buttler, D., Chandra, S., Cheng, Z., Coleman, M., Critchlow, T., Gupta, A., Han, W., Liu, L., Ludaescher, B, Pu, C., Moore, R., Shoshani, A., Vouk., M.: A Modeling and Execution Environment for Distributed Scientific Workflows. In 15th Intl. Conf. on Scientific and Statistical Database Management (SSDBM), Boston, Massachussets, 2003.

[8] Liu, L., Pu, C., Han, W.: An XML-Enabled Data Extraction Tool for Web Sources. Intl. Journal of Information Systems. Special Issue on Data Extraction, Cleaning, and Reconciliation, 2001.

[9] National Center for Biotechnology Information (NCBI). http://www.ncbi.nlm.nih.gov/, 2006.

[10] Klasky, S.A., Ludaescher, B., Parashar, M.: The Center for Plasma Edge Simulation Workflow Requirements. Post-ICDE IEEE Workshop on Workflow and Data Flow for Scientific Applications (SciFlow'06), Atlanta, GA, April 8th 2006.

[11] GEON: Geosciences Network. http://geongrid.org

[12] RESURGENCE: RESearch sURGe ENabled by CyberinfrastructurE. http://www.baldridge.unizh.ch/resurgence/

[13] Sudholt, W., Altintas, I., Baldridge, K.K.: A Scientific Workflow Infrastructure for Computational Chemistry on the Grid. 1st International Workshop on Computational Chemistry and Its Application in e-Science in conjunction with ICCS 2006.
[14] Ludaescher, B., Altintas, I., Gupta, A.: Compiling Abstract Scientific Workflows into Web Service Workflows. In 15th Intl. Conf. on Scientific and Statistical Database Management (SSDBM), Boston, Massachussets, 2003.
[15] Altintas, I., Barney, O., Jaeger-Fank, E.: Provenance Collection Support in the Kepler Scientific Workflow System, In the Proceedings of International Provenance and Annotation Workshop (IPAW'06), Chicago, Illinois, USA, May 3-5, 2006.
[16] Abramson, D., Kommineni, J., Altintas, I.: Flexible IO services in the Kepler Grid Workflow System. In the Proceedings of the 1st E-Science Conference, Melbourne, Australia, December 2005.
[17] Altintas, I., Birnbaum, A., Baldridge, K.K., Sudholdt, W., Miller, M., Amoreira, C., Potier, Y., Ludaescher, B.: A Framework for the Design and Reuse of Grid Workflows. In P. Herrero, M.S. Perez, V. Robles, editor, Scientific Applications of Grid Computing: First International Workshop, SAG 2004, Beijing, China, September 20-24, 2004, Revised Selected and Invited, in series Lecture Notes in Computer Science, nr 3 pp. 119-132. Springer-Verlag GmbH, 2005. ISBN 3-540-25810-8.
[18] Jaeger-Frank, E., Crosby, C.J., Memon, A., Nandigam, V., Arrowsmith, J.R., Conner, J., Altintas, I., and Baru, C.: Three Tier Architecture for LiDAR Interpolation and Analysis, 1st International Workshop on Workflow systems in e-Science in conjunction with ICCS 2006.
[19] Bowers, S., Ludaescher, B.: An Ontology Driven Framework for Data Transformation in Scientific Workflows. In International Workshop on Data Integration in the Life Sciences (DILS), LNCS 2994, Leipzig, Germany, March 2004.
[20] SCIRUN II. http://software.sci.utah.edu/scirun.html
[21] Ptolemy Project, See Website: http://ptolemy.eecs.berkeley.edu/ptolemyII/
[22] Yang, G., Watanabe, Y., Balarin, F., Sangiovanni-Vincentelli, A.: Separation of Concerns: Overhead in Modeling and Efficient Simulation Techniques. Fourth ACM International Conference on Embedded Software (EMSOFT'04), September, 2004.
[23] Lipps, J. H.: The Decline of Reason?. http://www.ucmp.berkeley.edu/fosrec/Lipps.html.
[24] Callahan, S., Freire, J., Santos, E., Scheidegger, C., Silva, C., and Vo, H.: Managing the Evolution of Dataflows with VisTrails. In Proceedings of the IEEE Workshop on Workflow and Data Flow for Scientific Applications (SciFlow'06), Atlanta, GA, April 8th 2006.
[25] Bavoil, L., Callahan, S., Crossno, P., Freire, J., Scheidegger, C., Silva, C., and Vo, H.: Vistrails: Enabling interactive multipleview visualizations. In IEEE Visualization 2005, pages 135–142, 2005.
[26] Globus Security Infrastructure. http://www.globus.org/toolkit/docs/4.0/security/.
[27] Abramson, D., Kommineni, J.: A Flexible IO Scheme for Grid Workflows. IPDPS-04, Santa Fe, New Mexico, April 2004.performance Distributed Computing Conference, August, 2000, pp. 147-154.
[28] Carter, W.E., Shrestha, R.L., Tuell, G., Bloomquist, D., Sartori, M.: Airborne Laser Swath Mapping Shines New Light on Earth's Topography. Eos (Transactions, American Geophysical Union), v. 82. p. 549, 2001.
[29] lythe J., Jain S., Deelman E., Gil Y., Vahi K., Mandal A., Kennedy K.: Task Scheduling Strategies for Workflow-based Applications in Grids. CCGrid 2005.