A Workflow Job Manager for the Nevada Climate Change Portal

Ivan Gibbs, Eric Fritzinger, Sergiu M. Dascalu, Frederick C. Harris, Jr., and Yantao Shen
Department of Computer Science and Engineering
University of Nevada MS-171
Reno, Nevada 89557 USA
igibbs@cse.unr.edu, ericf@cse.unr.edu, dascalus@cse.unr.edu, fredh@cse.unr.edu, yshen@unr.edu

***Abstract*—Model coupling has been of interest due to its promise of assisting massive amounts of reuse and thereby speeding up development cycles. A project at the University of Nevada, Reno dubbed the Nevada Climate Change Portal is currently being developed to assist in providing climate data and data services to researchers. One part of this portal is being developed to enable users to define scientific workflows and then run them. It is in this particular functionality of the Nevada Climate Change Portal in which the topic of this paper resides. This paper is concerned with the creation of a web-service based workflow job manager to present information more graphically than the typical text-based job manager. Modern software technologies such as .Net, Silverlight, WSDL, SOAP, and Graphviz are used to develop the features of this software. Aside from the gross technical details, the proposed job manager contains a way to represent the current progression status of a simulation in a graphical form and it is hoped that this will allow the adaption of the GUI to a wider variety of platforms such as smart phones or tablet computers. Increasing the availability of the job manager to different platforms is expected to allow users to collaborate in a more convenient manner, because they no longer need to use a PC with a large display in order to view the status of their running jobs.**

***Keywords*—Scientific computing; World Wide Web; Web services; Client server systems; Mobile computing**

I. Introduction

Currently, scientists use mathematical models to study phenomenon and to make predictions of future events. In the environmental sciences, these mathematical models typically model the environment. A situation that currently exists is that many models have been created and they are of different types and follow different methodologies. Since many models already exist, there is increased interest in attempting to chain models together to simulate the environment on a larger scale. A current limitation of this idea is the diverse models and their ability to work together in a chain. Demeter, the model and data interoperability component of the Nevada Climate Change Portal (NCCP) [1]–[4], has been proposed to address this issue. This component would allow scientists to more easily chain different models together and to run simulations [5].

This paper, which is based on [6], is focused on creating a job management web service for the Demeter framework. The intention is to allow scientists to use a web application to create, manage, and run their simulations. The job manager is focused on giving them the power to run their simulations. It will allow concurrent users and be power-loss tolerant. With scientists as the main user of this system, we want to to eliminate the need for the users to know anything about esoteric computing topics, in order to free up their concentration for their work. Another benefit of this system is to allow users to easily share their simulations or parts of their simulations with others, in order to enable others to benefit from and build on their work.

The rest of this paper is organized as follows. Supporting technology for the project is described in Section II. Section III details the need for the application. The software model is presented in Section IV with requirements, use cases, and design documentation. A prototype is described in Section V. Subsequently, Section VI briefly explains similar works and how they compare among each other. Finally, the conclusion and future work are addressed in Section VII.

II. Web Services

In using the World Wide Web (WWW) for a communication medium, we are exposed to specific constraints. First, the WWW works on the HyperText Transfer Protocol (HTTP) protocol. This provides a lot of capabilities and some limitations. Thus far, the WWW has become very popular and has a lot of development tools and documentation. The area is quickly changing to adapt to new realizations on what is needed or wanted from the WWW. Nothing appears to be very certain for any length of time, which leaves an environment that is ripe with opportunity and pitfalls. Despite the quickly changing times of the WWW, the wide adoption of this medium gives it great value.

A. Service-Oriented Architecture

A Service-Oriented Architecture (SOA) is a way to promote reuse in a software system. It is generally set up as a client-server architecture where the server offers services to various clients. A SOA is a way to promote the reuse of specific software or a way of capitalizing on software that has been written.

In the days of the WWW, SOA has taken on a more precise definition. According to some they are architectures designed to support business enterprises on the internet [7]. They comprise a service, a communication medium and protocol, and clients. Unfortunately, there is no formal definition of what

constitutes a SOA and what is out there is vague and abstract rather than concrete and specific.

B. Web Applications

With the rise in the number of web applications, one begins to wonder what the real costs and benefits are. Some of the obstacles and opportunities to the growth of cloud computing are analyzed by the University of California, Berkeley [8]. For many, the concept of Software as a Service (SaaS) will lower the cost of computing and, at the same time, open up new opportunities for interoperability of different software packages. The current state of the different development methodologies of web services at this time can be viewed as a democratic process in which each developer of a web service is casting their vote. In the light of this viewpoint, the requirements of the desired web service become all the more critical.

C. Workflows

Workflows are commonly used in SOAs. Tools exist such as Microsoft's Windows Workflow Foundation (WWF), BonitaSoft's Bonita Open Solution, Yet Another Workflow Language (YAWL), the Workflow Toolkit, Kepler [9] and others. In addition, non-business workflows have been proposed [10] and are currently being researched. Some of these scientific workflow packages are available for use, such as the C++ Workflow Management System [11] offered on Source Forge or the python-workflow-engine [12].

III. The Need

The need to couple scientific models together to create more diverse and complex simulations has been recognized by studies [13], [14]. In addition to that, the current state of the software written for scientific models is of concern, in that those writing the software generally have had little education regarding how to write software [15]. Many concerns exist regarding the verification and validation of climate models [16], [17]. The field of software engineering has been plagued by problems and various methodologies have come along proposing to fix the problems, but each has failed to live up to its claims [18]. Given the state of the software field today, it can only be expected that tackling large systems of systems involving multiple scientific models working together is quite a complex and involved task with no easy solutions.

Is there a need to make this software via software engineering methods? Though scientists and can generally program software, it is not their core focus and they are not aware of sophisticated software methods. The nature of software is complex and this compounds when you are attempting to face novel problems. It is in this realm that software engineering is focused and it is proposed here to help with constructing such a scientific model combining system. Since being defined as a field in 1968, software engineering has identified strong methods that can aid in designing and building software. In the software marketplace and in the software research field there are many methods that are proposed or sold to the general population which do not have decent empirical evidence for their claims [19]. It is this situation that makes it difficult for someone without software expertise to easily create high quality software without much background knowledge [20]. Furthermore, the area of scientific climate models typically include resource hungry applications which can possibly benefit from distributed computing paradigms, which may take quite a long time for a novice to learn.

A. Motivation

In essence, the motivation to create a web service for the climatology model coupling project is driven by digital technology, the complexity of climatology research, augmenting collaboration between researchers, and the ubiquity of the WWW. Together all of these factors give a basis for attempting this endeavor to potentially increase the utility of computers for this part of the research community.

B. Technical Approach

When discussing Web Services in particular we can observe a highly dynamic area of software. Web Services can be complex with SOAP, Universal Description, Discovery, and Integration (UDDI) and Web Services Description Language (WSDL) or simpler with REpresentational State Transfer (REST) and EXtensible Markup Language (XML) [21].

Another topic of concern is how to represent data so that it can be displayed on different devices. Though the desktop PC has been ubiquitous for a number of years now, the popularity of cell phones is revealing the possibility that the traditional PC screen will not necessarily dominate forever. In order to deal with this complexity, we start out with the intention of representing the view of a workflow through any of the currently available devices.

IV. The Software Model

A. Requirements

The functional and non-functional requirements are enumerated here. Functional requirements list what the job manager must do. The non-functional requirements list additional constraints that the software also needs to satisfy.

1) Functional Requirements: These requirements specify the needs of a job management system for a workflow as listed in Table I. They allow us to determine when the application meets its objectives.

TABLE I. Functional Requirements: JM = Job Manager

R1	The JM must allow a user to view the current status of a workflow.
R2	The JM must allow a user to start a workflow.
R3	The JM must allow a user to stop a workflow.
R4	The JM must allow a user to pause a workflow.
R5	The JM must allow a user to unpause a workflow.
R6	The JM must provide an ID for each job.
R7	The JM must provide the current state of a job.
R8	The JM must allow a user to visualize a job's execution.
R9	The JM must allow a user to see all of their jobs.
R10	The JM must allow multiple users to view a job concurrently.

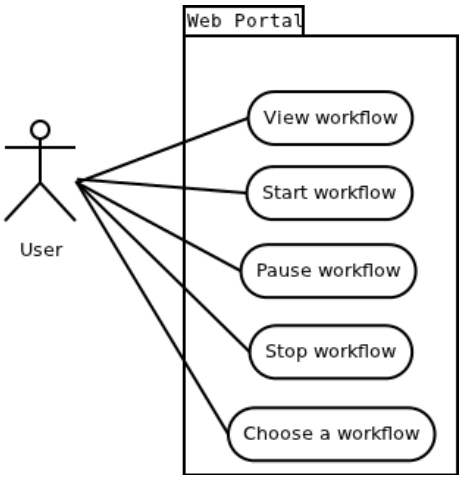

Figure 1. Use case diagram.

2) Non-Functional Requirements: These requirements document the constraints imposed regarding the achievement of the functional requirements [22] (Table I). The non-functional requirements in Table II were all inherited from the Demeter [2] project that the job manager was written for. We understand that these non-functional requirements are a bit atypical because they don't refer to quality requirements of the design such as reliability or speed, but our stakeholder's felt that these aspects of the design were important requirements. From our perspective, non-functional requirements list the 'how to implement' refinements of the functional requirements and therefore these technical constraints are listed here.

TABLE II. NON-FUNCTIONAL REQUIREMENTS: JM = JOB MANAGER

NFR1	The JM shall comply with: a) XHTML 1.1 b) CSS 2.1 c) SVG 1.1 d) MathML 2.0
NFR2	The JM shall be accessible via the following browsers: a) Mozilla Firefox 2.0 or above b) Internet Explorer 6.0 or above
NFR3	The JM shall be programmed using the .NET Framework 4.0 or above.
NFR4	The JM shall use the Silverlight 4.0 Framework.

B. Use Cases

Use cases give us a chance to analyze an application from the user's perspective and thereby drive user oriented applications [23]. The user's view of the system is related to interaction with some job that they want to monitor or control. The software allows users to remotely control a job and to view progress. A use case diagram [24] is shown in Figure 1. R6-R10 are condensed into 'View workflow' while R4 and R5 are both supported by 'Pause workflow.' These use cases are supported for the user through web services.

Detailed use cases are presented in Table III.

TABLE III. DETAILED USE CASES

UC1	The user will be able to view the current status of a workflow.
UC2	The user will be able to start the execution of a workflow.
UC3	The user will be able to pause the execution of a workflow.
UC4	The user will be able to stop the execution of a workflow.
UC5	The user will be able to choose from the available workflows.

C. Web Application

The web application runs on the client computer and periodically receives information from the server regarding the client workflow. In addition, the application can ask for information from the web server or request actions to be performed on the workflow. Decisions tables [25] are used to document the design of these web services. To follow the table, identify the column using your current workflow conditions (conditions are listed above the double line), then progress cell by cell down that column and execute all actions marked with an 'X' (actions are listed below the double line).

TABLE IV. VIEW WORKFLOW DECISION TABLE: R=RUNNING, S=STOPPED, P=PAUSED, F=FINISHED

Workflow Exists?	N	Y	Y	Y	Y	Y
Workflow State?	*	R	S	P	F	*
Workflow deleted while viewing?	N	N	N	N	N	Y
Display 'No WF Exists'	X					
Show current WF status		X	X	X	X	
Show results					X	
Show deleted by other user msg						X

Table IV represents the conditions for the specified actions by the client. The lag between the time that conditions change and the time that the client is notified depends on whether we use a listener or polled model of communication between server and client.

A user is also allowed some control over the workflow from the web service. Currently, there is no plan to allow for modifications to the workflow from the web service, but the user will be able to start, stop, or pause the workflow. Obviously concurrency regarding more than one user accessing and attempting to control the workflow might be a problem. It is here that we use some state machine techniques to overcome concurrency problems. The state transition matrix in Table V illustrates the user's desired action and the simulation's action in the first column and the current state of the workflow in the first row. An example would be that the user views the workflow and observes that it is currently running a simulation. If the user presses the pause button, the service will record the desired user action as a PAUSE and since the state machine is currently in the RUNNING state, it is valid to move to a PAUSED state. However, an illustration of an erroneous user action is that if for some reason the user was able to press the PAUSE button when the workflow had previously been in the STOPPED state then the state machine will report this as an erroneous action and refuse to modify the state of the workflow.

TABLE V. WORKFLOW STATE TRANSITION TABLE: E=EXISTS, R=RUNNING, P=PAUSED, S=STOPPED, F=FINISHED

		Current State			
		E	R	P	S or F
User	START	R	error	error	error
	PAUSE	error	P	R	error
	PLAY	error	error	R	error
	STOP	error	S	S	error
Simulation	FINISH	error	F	error	error

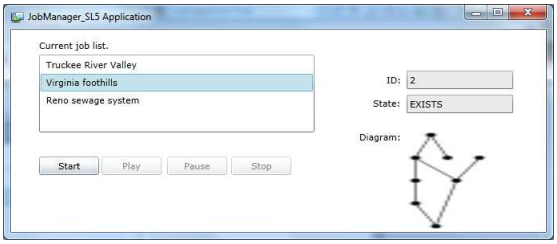

Figure 2. Stand-Alone prototype–Current information for a job.

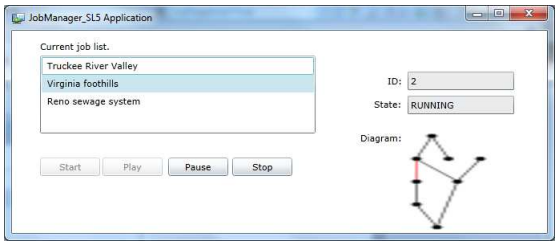

Figure 3. Stand-Alone prototype–A running job.

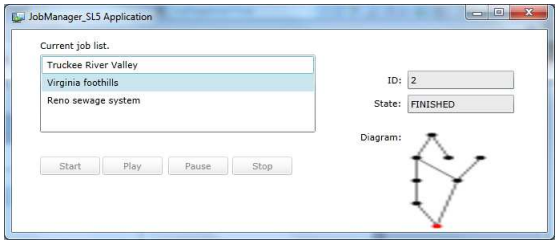

Figure 4. Stand-Alone prototype–A finished job.

The communication between the client and web server consists of the following messages: get job status, start job, play, pause, stop. The sequence diagrams for these actions are quite simple as they only require one request / response pair of messages.

D. Concurrency of Users

Two users may access a simulation and view it at the same time. The finite state machine design of the state of the simulation allows us to easily accomplish synchronization of multiple accesses. The state transitions will be protected by a locking mechanism and thereby prevent confusing conflicts to arise if multiple user desires correspond in time.

V. PROTOTYPE

A. Stand Alone Prototype

The real-world prototype of the job manager was created with 5 functions; get info, start, play, pause, and stop. These functions and their purpose are equivalent to the detailed use cases shown in Table III. After the user clicks on a job name in the list box the status of that job is shown (Figure 2). One can also see that the available buttons are enabled according to the current state of a particular job; see Table V for details on the state machine representation for the button enablement scheme. When the user starts their simulation, they will see the status change as shown in Figure 3. When the simulation has run to completion, the status will be reported as in Figure 4. At this stage, all of the job buttons are disabled. Each job is only meant for one simulation run and no more. After that simulation is finished or stopped, the user must create a new job to run that simulation again.

In addition to the typical simulation scenario where it runs to completion, the job manager can handle requests to pause the execution. When the user decides that they can proceed with the simulation they just press the ”Play” button to continue.

Though the only operations expected with the job manager are according to the defined workflow, accommodations were made for erroneous state transitions. These types of errors could occur if a client and server were out of sync due to polling updates, or multiple users concurrently accessing the same job workflow. If an illegal transition is requested, an error is returned and no state transition is effected for the illegal request (Figure 5).

B. NCCP Prototype

Two sub-projects of the NCCP were modified to add the Job Manager: Demeter and its Graphical User Interface (GUI) client Persephone. GUI items were added to the client Persephone and the Job Manager service was added to Demeter.

A typical usage of this Persephone client is to load or create a simulation and then run that simulation. A sample simulation is shown in Figure 6. Job control buttons are context sensitive. The user will press the Start button and the simulation will run. The status for the job is shown as ”RUNNING” to give the user some feedback regarding what is currently happening. Eventually, the simulation finishes and the final status is shown. The Job Status window is updated.

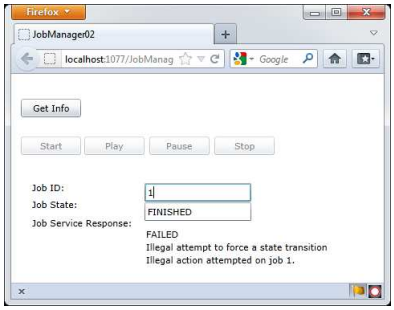

Figure 5. Stand-Alone prototype–An error.

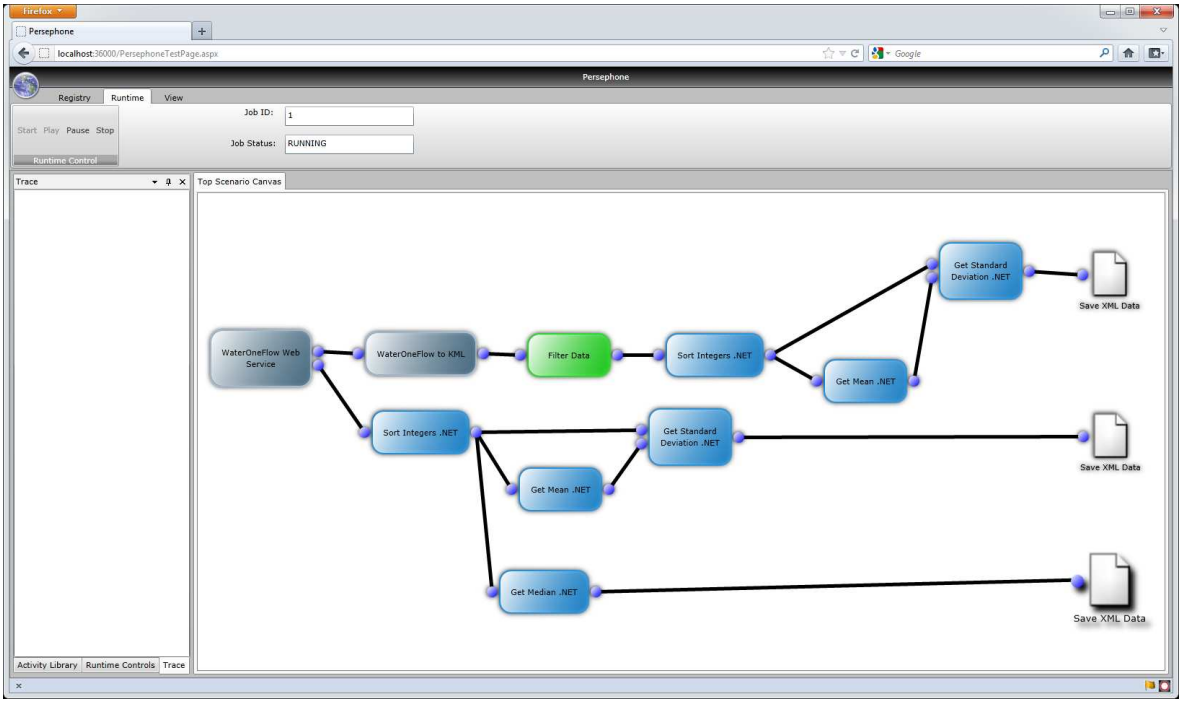

Figure 6. Demeter framework [2]–A running simulation.

VI. Similar Work

As there is a great need for scientific simulators, researchers and developers have created different ones. They all have a workflow management scheme. The following workflow management systems were surveyed in connection with our workflow job manager utility for the NCCP: the Trident Workbench, the RealFlow workflow system, the Natural Organic Matter Research Portal, the HPC Job Manager, Kepler, the Taverna workbench, and the Pegasus project. All of these scientific workflow tools have been described further here to give us a feel for how the NCCP job manager fits into the overall context of scientific workflow simulation.

A. Trident Workbench

Microsoft Research, the University of Washington, and the Monterey Bay Aquarium collaborated to provide this scientific workflow workbench [26] for the field of oceanography. It provides a workflow manager, visualization package, UI, data management, and data storage abilities to researchers.

The actual use of the job manager through the GUI is demonstrated in a video [27].

B. RealFlow Job Management

RealFlow Job Management [28] is an application for running graphical simulations for fluids and dynamics. It runs on a personal computer and job management is central to the application. RealFlow is an industrial strength application which has been used for movies, television, and commercials.

RealFlow consists of two applications, a job manager and an IDE for the user to create graphical simulations. The user creates their graphical simulation and then submits it as a job to the job manager. The IDE and job manager do not have to be running on the same computer, as the job manager can work over a network. Using networking and separate programs for the job manager and IDE open up a lot of opportunities for the users such as: splitting a simulation up to run on different job managers, assigning jobs to workgroups of computers, and an RSS feed to notify users of job progress. The job manager also gives the user specific preferences such as: specifying the port number for the manager or deleting temporary files after completion.

An actual demo of RealFlow can be seen online [29] which demonstrates the web interface to the job manager. The interface is designed as tables of text to display the different facets of the job manager such as job status, nodes, etc.

C. Natural Organic Matter Research Portal

Paper [30] refers to a portal that did some interesting things with their job manager. The job manager for this project was fairly simple. It assigns tasks on several simulation servers to achieve load balancing using a round robin algorithm. Intelligent agents, not the job manager, provide functions to send an email to the user when a job is done, predict the running time of a job, find a similar simulation to attempt to learn from those results, and to restart a stopped job from where it was stopped.

D. HPC Job Manager

Recently, a paper [31] was published describing the design of a job manager to ease the use of job management for scientists. The actual product was a collection of Unix and web services called the Simulation Application Manager (SAM).

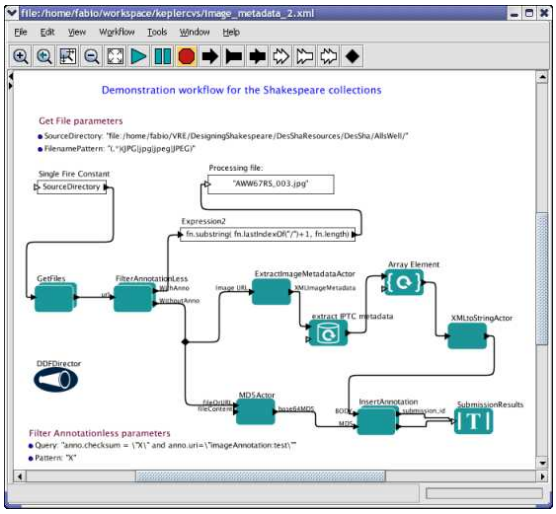

Figure 7. Kepler Workflow Manager [34].

It was made to ease the creation of portal interfaces for robust workflows. The tool uses other tools such as Globus Toolkit, Condor-G, and DAGMan (Directed Acyclic Graph Manager). In order to use this tool, a user will create a web portal representing the various configurations for their chosen job management. Then SAM can be used to code the job queuing, dispatching, processing and management. SAM will create files and workflow descriptions to be sent to other tools such as Globus. The API for SAM allows for: submitJob, getRunStatus, and getInfo. This API reveals the passive nature of the job manager in that the user cannot optionally start, stop, or restart a job after it has been submitted.

E. Kepler

Though Kepler [32] currently works from a local computer (Figure 7), it has a suggested requirements list for a web based UI interface for viewing workflows. They are listed as [33]: asynchronously view outputs and reports from workflows, configure workflows, run workflows, monitor status of a running workflow, and have a report actor to collect results from one or more workflows and report it via a Web UI.

F. Taverna Workbench

The Taverna Workbench Management System [35] provides tools for users to create and run scientific simulations. It is an extensible open-source project supported by the UK e-Research Community. This system is separated into four parts: the Taverna Engine, Taverna Workbench [36], Taverna Server, and a command line tool. The workflow manager has the following features: allow number of retries to be set for an activity that sometimes fails, allow user to play / stop / pause, and to operate from a stand-alone desktop computer.

G. Pegasus

Pegasus [37] is a workflow management system that offers excellent configuration choices [38]. It is used to map and execute application workflows using a variety of execution platforms such as Amazon Elastic Compute Cloud (Amazon EC2), TeraGrid, campus clusters, etc.

H. Discussion

Here, we discuss the different software packages named and compare them side by side to improve the ease of reading. In regards to job management, many properties may be compared. Due to various reasons, including the difficulty of performing an in-depth analysis of the feature-rich systems, only a brief comparison is done here. First, technical aspects of the job managers were compared regarding the platforms they run on, the type of GUI interface, and if they used some common 3rd party software libraries. In addition to the technical aspects, we wanted to know which functional aspects that the different managers meet. These comparisons were summarized in a table form with associated discussion.

The main technical aspects of job managers, are listed in Table VI. Though the GUI and Platform categories are quite easy to understand, some of the other categories are specific to high performance computing. The Condor Project develops policies and software tools to enhance the use of high performance computing for the non-expert user. Condor-G enables resource access in multi-domain environments and managing the computation within a single administrative domain. DAGMan manages dependencies between jobs that are described through a directed acyclic graph. Globus Toolkit allows users to build grids in order to benefit from distributed resource sharing. And, finally Graphviz supports visualizations of graphs. The different software products are similar with respect to the GUI and Platforms. However, they do differ quite a bit in regards to the different third party libraries employed.

TABLE VI. COMPARISON OF TECHNICAL ASPECTS: W=WINDOWS, M=MAC, L=LINUX

Software	GUI	Platforms	Condor-G	DAGMan	Globus	GraphViz
Trident Workbench	stand-alone web portal	W,M,L	N	N	N	N
RealFlow Job Mgmt		W,M,L	N	N	N	N
Natural Organic	web portal	Web	N	N	N	N
HPC Job Mgr.		W,M,L	Y	Y	Y	N
Kepler	stand-alone	W,M,L	N	N	Y	N
Taverna Workbench	JRE client cmd line	JRE	N	N	Y	Y
Pegasus	stand-alone Java webstart	W,M,L Web	Y	Y	Y	N
NCCP	web portal	W,M	N	N	N	Y

As shown in Table VII, the functional aspects of job management are more familiar than the technical aspects. The workflow systems listed here all satisfy most of these aspects. The Natural Organic Research Portal did not have pause/resume/stop capacity listed in the documents we read, but the system is somewhat dated and the lack of an actual artifact to investigate makes the rating here somewhat in doubt. Pegasus had no obvious pause/resume/stop capacity in the Java

Web Start interface they offer as a free download from their website. In other aspects the different packages agreed exactly.

TABLE VII. COMPARISON OF FUNCTIONAL ASPECTS

Software	pause/ resume/ stop	monitor execution via web	distribute execution
Trident Workbench	Y	Y	Y
RealFlow Job Mgmt	Y	Y	Y
Natural Organic	N	Y	Y
HPC Job Mgr.	Y	Y	Y
Kepler	Y	Y	Y
Taverna Workbench	Y	Y	Y
Pegasus	N	Y	Y
NCCP	Y	Y	Y

Though this comparison is somewhat limited, it is meaningful from a higher level point of view. A more comprehensive review would only delve into smaller and more specific details regarding the bigger picture. What emerges from the comparison laid out here is that these software packages compare quite well with each other. We have to note though that we do not claim that the job manager we created for NCCP has the same level of sophistication and technical capabilities as the software environments surveyed, some of them developed by many people over several years. Rather, we would like to show that our job manager, essentially developed by a couple of people in a relatively short period of time, has the core functionality needed for its purpose. What distinguishes our solution is in fact that it is simple (streamlined) and customized for our larger project, the Demeter component for model interoperability. This gives us control over the future developments of Demeter and allows for straightforward extension of our project.

VII. CONCLUSIONS AND FUTURE WORK

A. Conclusions

The job manager, offered as a service, was the main goal for this paper. This manager allows scientists to run simulations via the WWW rather than being confined to their personal computers. The event oriented job manager supports concurrency among multiple jobs that have been submitted. The user interface for interacting with the job manager has been incorporated into the Demeter component of the NCCP. Though this is a run of the mill job manager, some of its contributions include the display of a graph representing the current progress of the simulation and the use of a state transition matrix to enhance separation of concerns and thereby improving readability. The obstacles encountered in the creation of this software included doing the necessary research of WWW technologies and SOA approaches, integration difficulties with the .Net platform, lack of documentation for the NCCP, and the issue of the NCCP constantly changing during this project.

By implementing the job manager with a state transition matrix, thread locking code have been reduced, assertions have been reduced, and the code become easier to read and modify. Through the enhanced code clarity, confidence in the proper operation of the code is also achieved. Instead of using a separation of concerns via Aspect Oriented Programming (AOP) which separates each concern, many concerns are co-located in one spot thereby eliminating the need for the programmer to mentally keep track of parallel threads at the same time.

In many job managers that work with workflows, there will be a UI containing a diagram of the workflow. The representation is typically the same representation used when allowing the user to drag and drop components to create the diagram. Though this is a valid solution, this paper has gone outside of this typical solution by providing a more abstract representation of the workflow. A workflow was decomposed into a generic graph representation and that representation was used to create a simplified image of the workflow. To indicate the current status of execution of a workflow, appropriate nodes were highlighted in red. The advantages to this approach is that the representation of the workflow is reduced to a generic picture, which enables it to be re-sized to fit on smaller or larger screens.

With the new momentum behind smaller displays associated with tablet computing and smart phones, the sizing abilities of the workflow representation are believed to be an advantage in that researchers can access their jobs from any device. Communication forms of texting and using wireless cell phones have not been around before, but they have definitely increased our communication abilities. As such, we believe the newfound freedom of researchers due to mobile access of job management services enhances their ability to work with others. In particular, collaboration is facilitated through sharing of simulations and parts of simulations for the benefit of the scientific community.

B. Future Work

Some of the possible future modifications to our workflow job manager include more sophisticated scheduling of jobs, allowing for priority levels, email notification of job completion, usability improvements via feedback from user studies, and unit tests. Currently, the job manager does not take control of scheduling for jobs, but only tracks and monitors them. Jobs could eventually be distributed to appropriate computing devices depending upon job requirements. Priority levels would allow more important users to get their jobs to complete more quickly. For long running jobs, email notification could be used. Usability tests could give important feedback for improvements. Unit tests would increase the maintainability and reliability of the job manager. .Net enhancements and job specificity could be added. Other enhancements are also possible, but the ones listed here seem to be the most pressing ones at this time. If a usability study is conducted, the importance of proposed enhancements can be evaluated in a more precise manner.

ACKNOWLEDGMENT

We want to thank Michael McMahon, who provided details regarding web technologies used in this study.

This work was made possible through the support provided by the National Science Foundation under Cooperative Agreement No. EPS-0814372.

REFERENCES

[1] M. McMahon, S. Dascalu, F. Harris, S. Strachan, and F. Biondi, "Architecting Climate Change Data Infrastructure for Nevada," in *Advanced Information Systems Engineering Workshops CAISE-2011*, ser. Lecture Notes in Business Information Processing, LNBIP-83. New York, NY, USA: Springer, Jun 2011, pp. 354–365.

[2] E. Fritzinger, S. Dascalu, D. Ames, K. Benedict, I. Gibbs, M. McMahon, and F. C. Harris, "The Demeter framework for model and data interoperability," in *Proceedings of the International Congress on Environmental Modeling and Software (IEMSS-2012)*, ser. IEMSS-2012. Leipzig, Germany: iEMSs, 2012.

[3] S. Dascalu, E. Fritzinger, S. Okamoto, and F. Harris, "Towards a Software Framework for Model Interoperability," in *Industrial Informatics (INDIN), 2011 9th IEEE International Conference on*, Lisbon, Portugal, july 2011, pp. 705 –710.

[4] M. McMahon. (2012, October) Nevada Climate Change Portal (NCCP). http://sensor.nevada.edu.

[5] S. Rybacki, J. Himmelspach, and A. M. Uhrmacher, "Using Workflows to Control the Experiment Execution in Modeling and Simulation Software," in *Proceedings of the 5th International ICST Conference on Simulation Tools and Techniques*, ser. SIMUTOOLS '12. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2012, pp. 93–102. [Online]. Available: http://0-dl.acm.org.innopac.library.unr.edu/citation.cfm?id=2263019.2263031

[6] I. Gibbs, "Workflow Job Manager for NCCP," Master's thesis, University of Nevada, Reno, Department of Computer Science and Engineering, 2012.

[7] R. N. Taylor, N. Medvidovic, and E. M. Dashofy, *Software Architecture Foundations, Theory, and Practice*. John Wiley & Sons, Inc., 2010.

[8] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, and M. Zaharia, "Above the Clouds: A Berkeley View of Cloud Computing," University of California at Berkely, Berkeley, California, USA, Tech. Rep., 2009, technical Report UCB/EECS-2009-28.

[9] I. Altintas, B. Ludaescher, S. Klasky, and M. A. Vouk, "Introduction to Scientific Workflow Management and the Kepler System," in *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, ser. SC '06. New York, NY, USA: ACM, 2006. [Online]. Available: http://doi.acm.org/10.1145/1188455.1188669

[10] D. Zinn, S. Bowers, T. McPhillips, and B. Ludäscher, "Scientific Workflow Design with Data Assembly Lines," in *Proceedings of the 4th Workshop on Workflows in Support of Large-Scale Science*, ser. WORKS '09. New York, NY, USA: ACM, 2009, pp. 14:1–14:10. [Online]. Available: http://doi.acm.org/10.1145/1645164.1645178

[11] CNAF of the National Institute of Nuclear Physics. (2011, April) C++ Workflow Management System. http://sourceforge.net/projects/cppwfms/.

[12] University of Colorado. (2011, April) python-workflow-engine. http://code.google.com/p/python-workflow-engine/.

[13] S. M. Easterbrook, "Climate Change: A Grand Software Challenge," in *Proceedings of the FSE/SDP Workshop on the Future of Software Engineering Research*, ser. FoSER '10. New York, NY, USA: ACM, 2010, pp. 99–104. [Online]. Available: http://doi.acm.org/10.1145/1882362.1882383

[14] B. Liu, H. Liu, L. Xie, C. Guan, and D. Zhao, "A Coupled Atmosphere-Wave-Ocean Modeling System: Simulation of the Intensity of an Idealized Tropical Cyclone," *Monthly Weather Review*, vol. 139, no. 1, pp. 132–152, 2011. [Online]. Available: http://journals.ametsoc.org/doi/abs/10.1175/2010MWR3396.1

[15] S. M. Easterbrook and T. C. Johns, "Engineering the Software for Understanding Climate Change," *Computing in Science Engineering*, vol. 11, no. 6, pp. 65 –74, Nov.-Dec. 2009.

[16] A. Masys, "Understanding Climate Change through Modeling and Simulation: A Case for Verification, Validation and Accreditation," in *EIC Climate Change Technology, 2006 IEEE*, May 2006, pp. 1 –10.

[17] C. L. Conwell, R. Enright, and M. A. Stutzman, "Capability Maturity Models Support of Modeling and Simulation Verification, Validation, and Accreditation," in *Proceedings of the 32nd Conference on Winter Simulation*, ser. WSC '00. San Diego, CA, USA: Society for Computer Simulation International, 2000, pp. 819–828. [Online]. Available: http://portal.acm.org/citation.cfm?id=510378.510496

[18] W. W. Gibbs, "Software's Chronic Crisis," *Scientific American*, p. 86, September 1994.

[19] F. P. Brooks, "No Silver Bullet: Essence and Accidents of Software Engineering," *IEEE Computer*, vol. 20, pp. 10–19, 1987.

[20] S. D. Fraser, F. P. Brooks, Jr., M. Fowler, R. Lopez, A. Namioka, L. Northrop, D. L. Parnas, and D. Thomas, ""No Silver Bullet" Reloaded: Retrospective on "Essence and Accidents of Software Engineering"," in *Companion to the 22nd ACM SIGPLAN Conference on Object-oriented Programming Systems and Applications Companion*, ser. OOPSLA '07. New York, NY, USA: ACM, 2007, pp. 1026–1030. [Online]. Available: http://doi.acm.org/10.1145/1297846.1297973

[21] Q. Wang and Y. Liu, "Representation Design for RESTful Web Services," in *Next Generation Web Services Practices, 2008. NWESP '08. 4th International Conference on*, Oct. 2008, pp. 5 –9.

[22] F. Tsui and O. Karam, *Essentials of Software Engineering, Second Edition [Online Edition]*. Jones and Bartlett Publishers, 2011.

[23] I. Jacobson, *Object Oriented Software Engineering: A Use Case Driven Approach*. Addison-Wesley Professional, July 1992.

[24] M. Fowler and K. Scott, *UML Distilled, A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley, 2000.

[25] R. B. Hurley, *Decision Tables in Software Engineering*. New York, NY, USA: John Wiley & Sons, Inc., 1983.

[26] Microsoft. (2012, October) Project Trident: A Scientific Workflow Workbench. http://research.microsoft.com/en-us/collaboration/tools/trident.aspx.

[27] E. C. Withana. (2012, December) Running WRF workflows on Microsoft Azure through Trident Scientific Workflow Workbench. http://www.youtube.com/watch?v=BZMjqVgJiIo.

[28] Next Limit Technologies. (2012, July) Realflow. http://www.realflow.com/.

[29] RealFlowLabs. (2012, December) Real Flow: Job Manager 1. http://www.youtube.com/watch?v=-1pnD5w7kz8.

[30] X. Xiang, G. Madey, Y. Huang, and S. Cabaniss, "A Web Portal for Environmental Research," in *Environmental Online Communication*. Springer, 2004.

[31] A. Kulshrestha and G. Allen, "Service oriented architecture for job submission and management on grid computing resources," in *High Performance Computing (HiPC), 2009 International Conference on*, Dec. 2009, pp. 13 –19.

[32] UC Davis, UC Santa Barbara, and UC San Diego. (2012, July) Kepler. https://kepler-project.org/.

[33] Kepler. (2012, October) Overview of web UI needs. https://kepler-project.org/developers/interest-groups/webui/diversity-of-web-ui-needs.

[34] The Kepler Project. (2012, October) The Multivalent - Kepler - Chesire VRE Project. http://bodoni.lib.liv.ac.uk/VRE/.

[35] C. Goble. (2012, July) Taverna. http://www.taverna.org.uk/.

[36] Taverna. (2012, October) Taverna Users Guide. http://www.cagrid.org/display/workflow13/Taverna+Users+Guide.

[37] E. Deelman, G. Mehta, K. Vahi, F. Silva, and et.al. (2012, July) Pegasus. http://pegasus.isi.edu/.

[38] Pegasus Collaborative Computing Group. (2012, October) Pegasus GUI Documentation. http://pegasus.isi.edu/documentation/gui.