e-Science Project and Experiment Management with Microsoft Project

Gregor von Laszewski and Leor E. Dilmanian
Rochester Institute of Technology, Rochester, NY 14623
Email: laszewski@gmail.com

***Abstract*—The design, execution, and monitoring of challenging scientific applications is often a complex affair. To cope with the issue, several tools and frameworks have been designed and put into use. However, the entry barrier to using these tools productively is high, and may hinder the progress of many scientists or non-experts that develop workflows infrequently. As part of the Cyberaide framework, a workflow tool called Grid Manager had been developed and integrated using the Microsoft Project software package as an elementary component. The advantages of using such a tool are discussed in this paper. Microsoft Project is a user friendly project management tool that is being used in a new context. It is being used to design and monitor the execution of Grid based projects. The motivation for this choice is that many scientists are already familiar with Microsoft Project. Grid Manager enables seamless access to and execution over computational Grids, such as the NSF sponsored TeraGrid. Our framework also allows integration with other resources, including Microsoft Windows HPC Server 2008 (HPC) clusters. We test our hypothesis of usability while evaluating the tool as part of several graduate level courses taught in the field of Grid and Cloud computing.**

Keywords: Grid Computing, Workflow, Microsoft Project, CoG Kit

I. Introduction

The design, execution, and monitoring of distributed scientific applications is often a complex affair. As part of the scientific communities, the most demanding computational problems require scheduling of large-scale resources, orchestration of activity, and assertion of quality of service requirements. Quality of service is typically described in terms of performance, cost, reliability, trust, or fidelity of resources or scientific instruments.

The workflow paradigm has proven itself to be an effective paradigm to impose over challenges produced by many scientific problems. To address these challenges, several different frameworks have been developed [1] to provide usable tools which streamline the development and execution of distributed applications on computational Grids. In this paper, they are referred to as *Grid Workflow Systems*.

Although these systems do exist, there is a need to provide a tool which is more user friendly, and does not create a large entry barrier into the Grid. To cope with the complexities presented, we have created a framework called Cyberaide. Cyberaide contains a prototype tool called Grid Manager. It is based on Microsoft Project, and it is used to create and execute Grid based workflows. There are several advantages towards using such a tool.

It is one of the objectives of this paper to discuss the use of Microsoft Project in several ways. First and foremost, Microsoft Project can be used as a workflow modeling, or design tool. The end user uses the software package to define a project in terms of tasks, which are pieces of work that need to be done, and dependencies among tasks, which are used to describe the order in which they need to be done. The end user may or may not decide to manually assign resources, such as computers, data storage devices or instruments, to the various tasks for execution.

We will integrate Microsoft Project with various middleware components typically found in Grid workflow systems (GWS). These components include a workflow engine, information services, and the Commodity Grid Kit (CoG Kit). Once the workflow design is complete, it is executed seamlessly by a workflow engine. In turn, information services may be used either at design time by the end user, or at run time by the workflow engine, to create a suitable schedule(task/resource assignment). The CoG Kit is used by the workflow engine for convenient access to Grid services, such as file transfer and job submission. We discuss these components and their integration individually.

During execution of a workflow, it is desired that the end user is able to monitor and track execution related information from the workflow engine in close to real time. Once execution of the distributed application begins, Microsoft Project can be used to track or build visual reports on the current execution progress, accrued cost, or errors, for example, of the running project. Various graphical interfaces, called *views* are provided to manage workflows and resources. It is another objective of this paper to discuss how Microsoft Project is used as a workflow and resource monitoring tool during the execution stage.

We discuss the strengths and weaknesses of Microsoft Project for integration within Cyberaide. After an evaluation of several project management software packages, it is expected that Microsoft Project contains the appropriate feature sets, GUI's and API's to accomplish all of the activities stated above. End users should find the well known software package to be easy to learn and use, while organizations benefit from increased productivity and collaboration in the development of Grid based applications. The prototype of Grid Manager demonstrates its ability to seamlessly access and execute jobs on the TeraGrid. It is believed that Microsoft Project is an appropriate tool for the development and execution monitoring

of scientific workflows on the Grid.

This paper is structured as follows. First we review some related research. A vision for Cyberaide is presented. The role of Grid Manager in e-Science project management is described. The implementation is presented prior to conclusion.

II. Related Research

Many activities have been conducted in areas related to our research. Cyberaide reuses ideas and concepts ranging from the early meta-computers [2], to Grid [3], [4] and Cloud [5], [6] computing. In contrast to developing middleware and middleware services, however, focus is spent on the development of the next level of services and tools that simplify the use and the integration of such middleware in application domains [7]. Thus the framework is targeted towards making use of this middleware more feasible for those that do not have the time to learn the complex middleware or build one-of-a-kind applications.

A large compilation of workflow tools exist [1] or are under development. We have ourselves contributed to that area with a robust Grid workflow engine that allows the integration of interactive steering [8]. Much of the work is focused on developing abstractions to the rapidly changing Grid environment so that new and different Grid and Cloud tools can be integrated while exposing familiar use patterns.

Other areas of interest include the creation of sophisticated scripting environments and languages which simplify the development of workflows and scripts through interpretive scripting frameworks. Traditionally, workflow tools such as Condor [9] and CoG Kit [10] have been used as part of the job management process in Grids.

From the commodity market a number of project management tools [11] exist. From the scientific field we are influenced by scientific portals [12], [13], electronic notebooks [14], and collaboration technologies [15] including the newest social networking tools [16].

The paper is structured as follows. First we review some related research and technologies. A vision for Cyberaide is described, along with a detailed outline of the role of Grid Manager within Cyberaide, a discussion on implementation followed by the conclusion.

III. Cyberaide Framework

The objective of Cyberaide is to provide a framework for simplifying the development of advanced Cyberinfrastructure or e-Science applications. Instead of just focusing on the middleware we also intend to provide access and guidance to include higher-level toolkits and services. Much of the framework will be available through an Interface Development Environment (IDE) allowing one to custom design applications while integrating various components through form base templates or drag and drop component directories. The components include access to Grid and Cloud middleware and tools that allow access to social networking services. Through this integration, a rich set of functionality addressing the development, deployment, and execution of e-Science applications is available to the end user (see Figure 1).

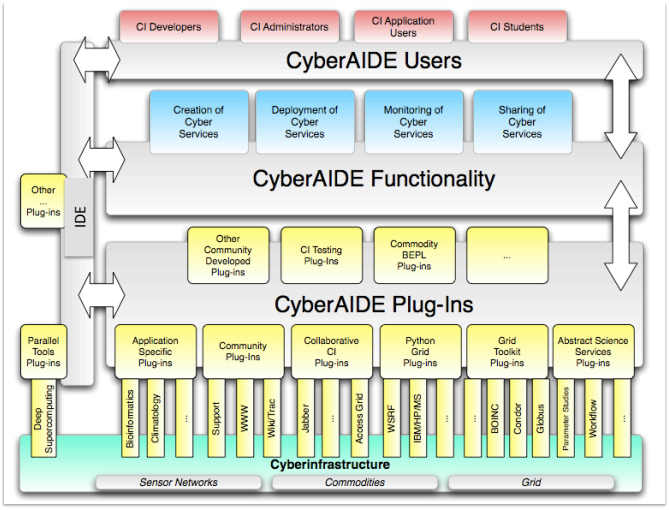

Fig. 1. Cyberaide Framework

IV. e-Science Project Management

Likewise, the objectives for Grid Manager are to provide users familiar with Microsoft's tools (e.g. Microsoft Project) an integrated user experience so they can focus on conducting science rather than learning yet another tool. This was one of the main complaints that were communicated by some users of workflow systems. Furthermore, one is able not only to manipulate workflow through a graphical user interface, but also through a simple script or command line interface. The entire execution of the workflow must be conducted dynamically.

Thus the Cyberaide Grid Manager can be used

- as a task definition tool, to easily compose workflow (modifying tasks, their dependencies, resources, task/resource assignments).
- as a workflow verification tool, to maintain a valid workflow.
- as an execution monitoring tool, to track or monitor the *experienced* quality of service and execution. Such parameters may include the actual baseline or task execution progress, execution failure, cost, or status of tasks.
- to view a listing of resources and their corresponding quality of service parameters. Such parameters may include reliability, usage, cost, fidelity, architecture, and availability.
- as a front end to all of the above through a GUI, shell/console and scripts.

The Grid Manager aims to use certain components of the CoG Kit.

A. Microsoft Project

The Microsoft Project and its graphical user interface provides a feature rich environment with powerful visualization capabilities. For example, end users have several task and resource management views defined for them. Each view is an interface which portrays different information. Such views

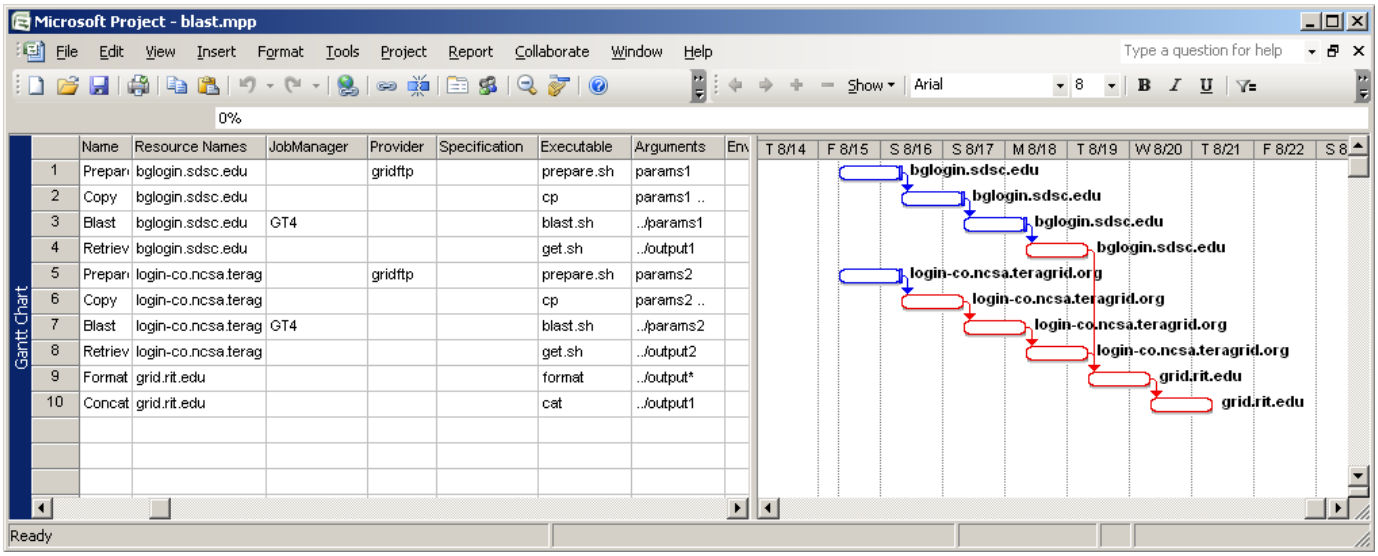

Fig. 2. Using Microsoft Project to build and execute a Grid based workflow.

include resource graphs, calendars, charts, and usage diagrams. Within Grid Manager, the Microsoft Project environment had been customized for Grid based workflows.

Figure 2 shows a workflow and highlights the well known features of Microsoft Project in the *Gantt Chart* view. This view consists of the task table in the left window pane, and Gantt chart in the right window pane. Within this view, one can visualize the workflow using the Gantt chart, and easily modify it using the task table. The Gantt chart is currently configured to convey some progress tracking and resource assignment information along with the workflow.

The task table has many custom fields, or columns, defined for execution parameters. When changes are made to the workflow, adjustments are automatically calculated and displayed. Invalid changes to the workflow are rejected. Grid Manager provides the ability to model deadlines (compare [17]), advanced reservation [18], reoccuring tasks, and task splits. Graphical indicators can be used to display custom notes about task execution on the Grid or warnings related to resource leveling and deadlines. Additionally, Visual Studio Tools for Office can used to create custom forms for the end user. Using both standard and custom fields associated with tasks and resources, the software package can be used to store and monitor various quality of service and execution parameters.

A rich set of features assist in multiple project management and collaboration. These features include resource sharing, interproject dependencies, and sub-workflows. Resource sharing and discovery can be accomplished using built in support from Microsoft Project, if not from other information services. Namely, it relies on active directory, the address book, or a Microsoft Project Server to discover and share resources. Additional tools of interest include built in data analysis for quality of service prediction and visual reports for the sharing of historical information.

The features of Microsoft Project had been compared to those of other leading project management solutions in a review [19]. One of the major complaints about Microsoft Project is that it is not free nor open source. However, there are many advantages towards using Microsoft Project. It received the highest overall rating. Criteria used to judge the product include collaboration, resource management, project management, ease of use and support. The review rates Microsoft Project favorably in each category.

Our main concern in choosing the right software package, a concern voiced by many professionals, is ease of use. It is our goal to minimize the entry barrier into Grid computing. The availability of additional Microsoft tools that can support planning, workflow composition, execution, document publication, notification, information sharing may be conductive to the scientific discovery and review process, which can be of immense value. Microsoft Project provides the necessary abstraction and is a tool of choice for many users. Simplicity and ubiquity are the essential factors for the user adoption. According to the review, “many professionals are already familliar with Microsoft Project”. The software package requires a small learning curve because of its layout. Its user interface appropriately hides a large volume of information and functionality and is divided into sections (views) to simplify navigation. The speadsheets makes it convenient to enter information without a mouse. An additional review of Microsoft Project over existing Grid workflow composition tools found in [1], [20] shows that the same advantages hold.

B. Console

In addition to the manipulation of the object through a graphical user interface, our implementation also contains a command line console allowing integration of scripts and ad-hoc computational steering. The console application immediately opens an instance of Microsoft Project in a new win-

dow, and starts running a command shell within the console window. From within the console widow, the end user can issue commands (workflow composition commands) which manipulate the active project.

For example, when a command is issued to add a task to the workflow, the corresponding graphical changes are reflected within the graphical user interface. Workflow composition commands available in the command line environment allow one to add, remove, specify, search, list, save, create and remove dependencies among, or assign resources to tasks. Microsoft Project still provides the extra added benefit of rejecting some invalid workflow operations, even from the console. Once the workflow is created, the end user may start execution using the run-all command, which submits the active project to a workflow engine. The run-all command may additionally start or destroy a single sign on proxy, or ask for other information such as credentials, the host name for and location of a tracking database, and a polling interval. Each command is implemented as a command object registered with the command shell within Grid Manager before starting.

C. Workflow Engines

There are two modalities with which Cyberaide Grid Manager can be used. It is first and foremost a workflow definition tool. Once a workflow is defined, it may be saved and exported for execution in other frameworks such as Condor, or the CoG Kit Karajan workflow engine [21], [22]. It is the job of the workflow engine to orchestrate the execution of jobs. In turn, this implies the interpretation of events, such as the completion of tasks, and the issuing of new tasks to be executed. There are many types of workflow engines with different features. Our main interest lies in Grid based workflow engines, or engines which can otherwise be customized for the Grid. A Grid based workflow engine may have features to support fault tolerance, data movement, and scheduling on the resources. For example, the Karajan workflow engine supports failure handling, checkpointing, dynamic and distributed workflows.

Most workflow applications are utilizing tools that clearly separate between the definition and execution. However, a significant number of problems will need a more interactive mode to allow for ad-hoc computational steering [23]. Hence changes to the workflow definition can occur during execution. This includes redefinition of jobs and tasks to be executed, dependency manipulation, and adaptation to variations in resource availability that requires human intervention. Scenarios where such modalities are common are emergency management, large-scale instruments utilization such as synchrotrons and photon sources, and the interactive process often found in many e-Science applications and research activities. There is need to support real time, dynamic ad-hoc job management for e-Science applications.

Additionally, there is a need for workflow engines to support persistence for long running workflows. Not all workflows are small enough to fit in main memory. It is desired that the client machine can be powered down and restarted at any point in time while the workflow is running (for days or weeks) on another host. The end user should be able to view the current state of the workflow at any point in time.

There are many features found in Windows Workflow Foundation which are suitable for the interactive and long running workflows. Some of these features include SQL persistence and tracking services, suspension and cancellation of activities, and dynamic workflow updates. Other features of interest include support for transaction and compensation. Overall, Workflow Foundation provides a highly customizable environment for creating and hosting interactive workflows. Many aspects of the runtime engine or local or core services can be manually configured or rewritten.

Cyberaide Grid Manager currently contains its own workflow engine, used to demonstrate tracking capabilities of Microsoft Project. Grid Manager was successfully used for workflow execution on the TeraGrid, using the CoG Kit for remote job submission. In this mode, each task specified for execution has two or more execution parameters defined for it, including executable, arguments, environment, and resource. These parameters will be passed along to the CoG kit accordingly, and are referred to as *control parameters*. Another set of fields relating to a running instance of the workflow are displayed. We refer to these parameters as *execution parameters* because they are related to a running instance of the workflow. Execution parameters include job identification numbers, error messages, and status updates.

In our project, a workflow execution over the TeraGrid [24] begins with authentication. The end user authenticates with a *single sign on* proxy. Prior to or during execution, the end user schedules the workflow by assigning each task a resource. Workflow enactment, the orchestration of remote job submission begins. The tasks are submitted for execution, passing control parameters to CoG Kit job submission commands. Upon the completion of task execution, the standard input, output, and error streams appear appended to the *task notes* associated with the corresponding task in Microsoft Project, and its status is updated.

Grid Manager will be modified to submit workflows to Karajan or Windows Workflow Foundation. It is expected that the various workflow engines can be integrated with Grid Manager, and provide some tracking service (information regarding progress that has been made). It is also expected that the various workflow engines can be customized to use the CoG Kit for job submission. Internally, tasks are represented through the Cyberaide Execution Object (CEO) and include attributes such as name, label, keyword, serviceContact, jobManager, provider, specification, executable, arguments, environment, directory, stdin, stdout, stderr, attributes, checkpoint, batch, redirected, and predecessors.

D. The CoG Kit

The workflow engine takes advantage of a number of runtime-oriented middleware operations. The CoG Kit provides a convenient API for these activities, allowing global single sign on, job submission, execution, cancellation, and file transfer. It utilizes the abstraction and provider model to

simplify access to job submission queues, job executions, and file transfer modes. Task level workflow abstractions include checkpointing for fault tolerance, various file transfer modes for data management, and access to information services.

E. Use Cases

As obvious from the functionality of Microsoft Project, the use cases can include the execution of partially ordered tasks with resource assignment constraints. Such constraints are typical for unique and large-scale instruments that need to be shared amongst a group of scientists. Microsoft Project supports sharing and collaboration, potentially providing much benefit to those using shared and scarce resources. Priorities corresponding to tasks assist in scheduling; priorities can be used to settle resource contention problems within a single workflow or among multiple workflows from various parties when an advanced reservation system is unavailable. Such functionality can be made available through both the Microsoft Project client and server products.

Projects using synchrotrons, flow-cytometers, and electron microscopes as resources can benefit greatly from such a tool. Even without specifying resource constraints as part of the workflow, it allows scientists to organize the experiments and enables a rigorous planning of the often complex calculations while being able to integrate them in the overall planning of an experiment. Furthermore, it can be used to guide resource allocations needed to invoke scientific applications required as part of emergency management situations, such as intrusion detection of contaminants in the water management system of a city [25].

Other use cases are based on the actual graphical user interfaces supporting workflow designs. Our framework will allow the import and export of workflows to be executed with other tools and workflow frameworks common in the Grid community.

F. Example

In our scenario, Larry, who is the end user, decides he wants to run two blast simulations concurrently on the TeraGrid. He has already received the proper resource allocations from the TeraGrid, and he has his access credentials which are set up and stored through the CoG Kit installation process. For each blast simulation, Larry will begin by uploading a script called "prepare.sh". The script is used for setting up his environment. He will copy the output of that script into another directory, using a "cp" command. The output file is used as input for the blast simulation. Larry begins by describing the the tasks that need to be executed.

```
add-task -path prepare.sh -arguments params1
add-task -path cp -arguments "params1 .."
add-task -path blast.sh
         -arguments "../params1 output1"
add-task -path get.sh -arguments "../output1"
```

These tasks must be accomplished in a particular sequential order. Larry uses the task IDs found before the first column of the task table to specify the ordering of tasks:

```
add-dependency 1 2
add-dependency 2 3
add-dependency 3 4
```

He has now specified how to run a single blast simulation, and can see what his workflow looks like at this point on the Gantt chart or task table. Two of these simulations will run concurrently. He will add another simulation by using cut and paste features on the task table spreadsheet, or by typing in the following commands:

```
add-task -path prepare.sh -arguments params2
add-task -path cp -arguments "params2 .."
add-task -path blast.sh
         -arguments "../params2 output2"
add-task -path get.sh -arguments ../output2
add-dependency 5 6
add-dependency 6 7
add-dependency 7 8
```

After these blast simulations run, he uses another script, "get.sh" to download output from both simulations to a nearby location on his school campus. Once the output had been retrieved, it is ready to be formatted and agregated:

```
add-task -path format -arguments ../output*
add-task -path cat
         -arguments "../output1 ../output2"
add-dependency 4 9
add-dependency 8 9
add-dependency 9 10
```

Since his workflow is small, he can add resources manually:

```
add-resource bglogin.sdsc.edu
add-resource login-co.ncsa.teragrid.org
add-resource grid.rit.edu
```

And assign resources to tasks using IDs:

```
add-assign 1 1
add-assign 2 1
add-assign 3 1
add-assign 4 1
add-assign 5 2
add-assign 6 2
add-assign 7 2
add-assign 8 2
add-assign 9 3
add-assign 10 3
```

Larry will run his workflow using the *runall* command. When he monitors execution, he may see something simmilar to figure 2.

V. Implementation

It is obvious that Microsoft Project [26] and its API were successfully used for implementation. Programmatic manipulation of Microsoft Project in C# was made possible through the Microsoft Project Primary Interoperability Assembly. Microsoft Visual Studio 2005 and the C# programming language were used to program the solution while reusing the Java implementation CoG Kit functionality to abstract the Grid.

Grid Manager is simply a console application. It opens a new active project in Microsoft project and provides a command line shell to accept and run workflow commands. Each workflow command manipulates the active project via the API, and is implemented as command objects which use the Apache Command Line Interface (CLI) to parse the command line parameters.

Once a workflow is defined, it can be exported to a workflow engine. Grid Manager can export a saved project using the save command. When this command is invoked, the project is serialized in JSON format. It can then me exported to the *mediator*, discussed in the next section.

Microsoft Project 2007 clients and servers, along with the workflow engines discussed all take advantage of some type of event driven architecture for the processing of workflow related events. In Karajan, elements react to events received from other elements and generate their own events. These events provide notification of state change, to control the execution of tasks and notify a front-end or tracking database of project status. In Microsoft Project Server, external application clients can respond to server side events. In the Microsoft Project client, event handlers for events related to workflow manipulation can be overwritten. The task of having the Microsoft Project client reflect the execution status of a workflow engine becomes trivial.

A. Status

A temporary workflow enactment engine, to demonstrate the project, was written in C# and follows the logic introduced by many other workflow engines operating on *directed acyclic graphs*. Thus, initial prototype of Cyberaide's Grid Manager issues Java CoG Kit execution commands by spawning them on a new process. Grid Manager will no longer use the CoG Kit to submit jobs directly to resources on the Grid. Future versions which utilize a web services based approach to create a *mediator* between the Grid and client machine are presently being developed.

Our current research efforts include the creation of abstractions for Cloud services such as Amazons Cloud Services and the integration. In addition, we are integrating our Grid and Cloud shell [27] that is under active development as part of the Cyberaide project. Additional research efforts include evaluation and integration of Grid Manager with alternative CoG Kit components, including the Karajan workflow engine.

For interactive steering and long running workflows, we will consider Windows Workflow Foundation. The Microsoft HPC SDK can be used for job execution on HPC clusters. The team will also experiment the integration of various alternative Microsoft products, including the Microsoft Office System, active directory, Address Book, SharePoint, Visual Studio Tools for Office, and PowerShell. Finally, the team will consider taking advantage of various information services, some of which include the Network Weather Service [28] or Globus MDS.

B. Evaluation

Cyberaide successfully demonstrates a workflow framework that uses Microsoft Project as a composition and execution monitoring tool for various Grids, including the TeraGrid. As part of our framework to simplify the use of advanced Cyberinfrastructure, Cyberaide Grid Manager had been developed and is able to steer computations on the Grid. It is extensible and provides an abstraction layer for the execution of tasks on computational resources. Figure 2 shows a screenshot of our tool in action while composing and executing jobs on a variety of resources on the TeraGrid.

We test the hypothesis of usability of the software solution. One of the major concerns regarding the automated manupulation of Microsoft Project is performance. Many end users complain about the amount of time it takes to insert information into or retrieve information from Microsoft Project. This issue presents a challenge particularly for techniques requiring automatic generation of workflows, such as interpretation of scripted workflows, or status updates.

Several techniques have been identified and implemented to improve performance. Grid Manager does not use any recorded macros to manipulate the active project, instead it directly accesses the Microsoft Project object model. Functions written in C# are expected to yield better performance results than subroutines in Visual Basic. Visual instructions, such as selection, are avoided.

Another major complaint affecting both performance and appearance is the redrawing of graphical changes onto the screen after each operation. By default, Microsoft Project automatically recalculates and updates the active screen to reflect the most current status changes within the project. Automatically generated or bulk operations manipulate the active project in the background. For example, screen updating and recalculation are shut off prior to the execution of a workflow script. These features can once again be enabled upon the completion of the script.

One major complaint regarding the solution involving screen updates is that the end user may be unaware that changes are taking place, and may grow impatient. For our purposes, the user can be made aware that operations are being performed through the console. Additionally, one can easily set the mouse pointer to an hour glass or messages on a status bar to inform the end user that background operations are taking place. As a direct result of this new approach, project changes appear to be updated periodically, in close to real time.

A Visual Studio Tools for Office addin was created. The objective is to measure performance of adding a large number of background tasks to an active project versus the performance of adding the same amount of foreground tasks to the active project. We define foreground jobs to be jobs that are performed without disabling screen updates or recalculation. The results show that the amount of time it takes to add N tasks to the active project scales linearly with respect to the number of tasks. The amount of time it takes to add N background tasks takes is proportionally faster than the addition of the

same number, N of foreground tasks. Unfortunately, neither technique can be used to create a workflow with more than one thousand tasks in a reasonable amount of time (five minutes). Previous performance tests [29] run on an earlier version of Microsoft Project and written in Visual Basic for Applications yields dramatically better performance results. The target size of ten thousand tasks was reached in twenty seconds on a laptop.

Another area for improvement is the command language found in Grid Manager. These commands can be simplified, and consolidated with those of the Grid shell. One complaint about the add-assignment and add-dependency command is that their parameters require that tasks and dependencies be described in terms of their ID's. As shown in our example with Larry, the Grid Manager commands appear to use ID numbers to identify tasks, rather than names which may be more intuitive. This design pholosophy is consistent with the Microsoft Project API functions depended on by Grid Manager commands. The Microsoft Project API uses some form of numerical ID to key or index objects such as tasks and resources. When a task is entered into the spreadsheet, for example, it is immediately given an ID number. Efficient lookup of these objects occur with the ID, or alternatively what is reffered to as a GUID. It is more desirable that the end user be able to describe tasks on the command line in terms of their name. In turn, a solution to this problem may require an external hash, which maps task names to task ID's for lookup. There are two major problems with this approach. The first problem is that the name field within Microsoft Project is not distinct. Two different tasks may have the same name. It is too difficult to identify and efficiently access a task for manipulation. Another problem is that maintenance of consistency between the hash and spreadsheet is a troublesome process, since the end user may decide to modify the spreadsheet directly. If a hash is not used, table lookups would be otherwise require inefficient linear scans of the spreadsheet.

VI. Conclusion

A prototype implementation of Cyberaide, which uses Microsoft Project as a definition and execution monitoring tool for the computational Grid has been developed. Microsoft Project provides a very feature rich environment for the workflow needs described. Current development efforts so far demonstrate and showcase the many features that Microsoft Project and the Java CoG Kit have to offer. They have been used successfully to execute Grid based workflows on the TeraGrid. The new user interface provides much of the functionality found in the Karajan components of the CoG Kit, while providing some new views such as resource usage and calendar views. The intention of the project is not just to create an interface to Grid back-ends. The full potential of Cyberaide becomes apparent with the release of a shell environment that consists of sophisticated experiment management abstractions and APIs as well as a convenient mediator service to hide complexities in the Grid or Cloud backends.

Acknowledgment

Work conducted by Gregor von Laszewski is supported (in part) by NSF CMMI 0540076 and NSF SDCI NMI 0721656.

References

[1] I. Taylor, E. Deelman, D. Gannon, and M. Shields, Eds., *Workflows for e-Science: Scientific Workflows for Grids*. Springer, 2007, iSBN: 978-1-84628-519-6.

[2] G. von Laszewski, "A Loosely Coupled Metacomputer: Cooperating Job Submissions Across Multiple Supercomputing Sites," *Concurrency, Experience, and Practice*, vol. 11, no. 5, pp. 933–948, Dec. 1999, the initial version of this paper was available in 1996. [Online]. Available: http://www.mcs.anl.gov/~gregor/papers/vonLaszewski--CooperatingJobs.pdf

[3] "The Globus Toolkit." [Online]. Available: http://www.globus.org

[4] "Condor: High Throughput Computing." [Online]. Available: http://www.cs.wisc.edu/condor/

[5] Amazon, "Simple Storage Services." [Online]. Available: http://www.amazon.com/gp/browse.html?node=16427261

[6] ——, "Elastic Compute Cloud." [Online]. Available: http://www.amazon.com/gp/browse.html?node=201590011

[7] K. Amin, M. Hategan, G. von Laszewski, and N. J. Zaluzec, "Abstracting the Grid," in *Proceedings of the 12th Euromicro Conference on Parallel, Distributed and Network-Based Processing (PDP 2004)*, La Coruña, Spain, 11-13 Feb. 2004, pp. 250–257. [Online]. Available: http://www.mcs.anl.gov/~gregor/papers/vonLaszewski--abstracting.pdf

[8] G. von Laszewski and M. Hategan, "Grid Workflow - An Integrated Approach," in *Technical Report.*, Argonne National Laboratory, Argonne National Laboratory, 9700 S. Cass Ave., Argonne, IL 60440, 2005. [Online]. Available: http://www.mcs.anl.gov/~gregor/papers/vonLaszewski-workflow-draft.pdf

[9] J. Frey, T. Tannenbaum, I. Foster *et al.*, "Condor-G: A Computation Management Agent for Multi-Institutional Grids," *Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing (HPDC10)*, 2001.

[10] G. von Laszewski, J. Gawor, P. Lane, N. Rehn, M. Russell, and K. Jackson, "Features of the Java Commodity Grid Kit," *Concurrency and Computation: Practice and Experience*, vol. 14, pp. 1045–1055, 2002. [Online]. Available: http://www.mcs.anl.gov/~gregor/papers/vonLaszewski--cog-features.pdf

[11] "Omniplan," Web Page. [Online]. Available: http://www.omnigroup.com/applications/omniplan/

[12] "Open Grid Computing Environments." [Online]. Available: http://www.ogce.org

[13] G. von Laszewski, B. Ruscic, K. Amin, P. Wagstrom, S. Krishnan, and S. Nijsure, "A Framework for Building Scientific Knowledge Grids Applied to Thermochemical Tables," *The International Journal of High Performance Computing Applications*, vol. 17, no. 4, pp. 431–447, Winter 2003. [Online]. Available: http://www.mcs.anl.gov/~gregor/papers/vonLaszewski--knowledge-grid.pdf

[14] A. Geist, "Electronic Notebook." [Online]. Available: http://www.csm.ornl.gov/~geist/java/applets/enote/

[15] "The Access Grid." [Online]. Available: http://www-fp.mcs.anl.gov/fl/accessgrid/

[16] "Facebook." [Online]. Available: http://www.facebook.com

[17] D. Abramson, "Nimrod Home Page," http://www.csse.monash.edu.au/ davida/nimrod.html/, March 2002. [Online]. Available: http://www.csse.monash.edu.au/~davida/nimrod.html/

[18] B. Sotomayor, K. Keahey, and I. Foster, "Combining batch execution and leasing using virtual machines," in *HPDC '08: Proceedings of the 17th international symposium on High performance distributed computing*. New York, NY, USA: ACM, 2008, pp. 87–96.

[19] "Project Management Software Review 2008," Web Page. [Online]. Available: http://project-management-software-review.toptenreviews.com/

[20] "Grid Workflow Forum, Workflow Composition Tools," Web Page. [Online]. Available: www.gridworkflow.org/snips/gridworkflow/space/Workflow+Composition+Tools

[21] G. von Laszewski, "Java CoG Kit Workflow Concepts," *Journal of Grid Computing*, Jan. 2006, http://dx.doi.org/10.1007/s10723-005-9013-5. [Online]. Available: http://www.mcs.anl.gov/~gregor/papers/vonLaszewski-workflow-taylor-anl.pdf

[22] G. von Laszewski, I. Foster, J. Gawor, and P. Lane, "A Java Commodity Grid Kit," *Concurrency and Computation: Practice and Experience*, vol. 13, no. 8-9, pp. 643–662, 2001. [Online]. Available: http://www.mcs.anl.gov/~gregor/papers/vonLaszewski--cog-cpe-final.pdf

[23] G. von Laszewski, M.-H. Su, J. A. Insley, I. Foster, J. Bresnahan, C. Kesselman, M. Thiebaux, M. L. Rivers, S. Wang, B. Tieman, and I. McNulty, "Real-Time Analysis, Visualization, and Steering of Microtomography Experiments at Photon Sources," in *Ninth SIAM Conference on Parallel Processing for Scientific Computing*, San Antonio, TX, 22-24 Mar. 1999. [Online]. Available: http://www.mcs.anl.gov/~gregor/papers/vonLaszewski--siamCmt99.pdf

[24] "TeraGrid Portal." [Online]. Available: http://www.teragrid.org/userinfo/portal.php

[25] G. von Laszewski, K. Mahinthakumar, R. Ranjithan, D. Brill, J. Uber, K. Harrison, S. Sreepathi, and E. Zechman, "An Adaptive Cyberinfrastructure for Threat Management in Urban Water Distribution Systems," Argonne National Laboratory, Tech. Rep., Jan. 2006, to be submitted. [Online]. Available: http://www.mcs.anl.gov/~gregor/papers/vonLaszewski-water-iccs.pdf

[26] "Microsoft Project Server 2007: Getting Started with a New Platform for Developers." [Online]. Available: http://msdn.microsoft.com/enus/library/bb456485.aspx#officepj2007platform__EventsDataSets

[27] G. von Laszewski, A. J. Younge, X. He, and F. Wang, "GridShell: Interactive Task Management for Grid and Cluster Computing," (submitted for review), Sep. 2008.

[28] R. Wolski, N. Spring, and J. Hayes, "The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing," *Journal of Future Generation Computing Systems*, vol. 15, no. 5-6, pp. 757–768, 1999.

[29] "Microsoft Project Performance Limitations," Web Page. [Online]. Available: http://zo-d.com/blog/archives/microsoft-project/microsoft-project-performance-limitations.html