National Computational Infrastructure for Lattice Gauge Theory SciDAC-2 Closeout Report Workflow and Reliability Subprojects

A proposal in response to Office of Science Notice DE-FG02-06ER06-04 and Announcement Lab 06-04: Scientific Discovery through Advanced Computing.

Lead Institution: Fermilab
Batavia, IL 60510

Lead Principal Investigator and DOE Contact: Paul Mackenzie
Address: Theoretical Physics Department
Fermilab
Batavia, IL 60510
Email: mackenzie@fnal.gov
Phone: 630-840-3347

Office of Science Programs Addressed: High Energy Physics and Nuclear Physics

Office of Science Program Office Technical Contacts: Lali Chatterjee, Ted Barnes, and Randall Laviolette

Participating Institutions and Principal Investigators:

Physics:

Boston University*, Richard Brower † ‡
Brookhaven National Laboratory*, Frithjof Karsch † ‡
Columbia University, Norman Christ †
Fermi National Accelerator Laboratory*, Paul Mackenzie † ‡
Indiana University*, Steven Gottlieb ‡
Massachusetts Institute of Technology*, John Negele † ‡
Thomas Jefferson National Accelerator Facility*, David Richards † ‡
University of Arizona*, Doug Toussaint ‡
University of California, Santa Barbara*, Robert Sugar † ‡
University of Utah*, Carleton DeTar ‡
University of Washington, Stephen Sharpe †

Computer Science:

DePaul University*, Massimo DiPierro ‡
Illinois Institute of Technology*, Xian-He Sun ‡
University of North Carolina*, Rob Fowler ‡
Vanderbilt University*, Abhishek Dubey ‡

* Institution submitting an application

† Project Principal Investigator, Member of the USQCD Executive Committee

‡ Institution Principal Investigator

1 Reliability Project

As part of this project work, researchers from Vanderbilt University, Fermi National Laboratory and Illinois Institute of technology developed a real-time cluster fault-tolerant cluster monitoring framework. This framework is open source and is available for download upon request. This work has also been used at Fermi Laboratory, Vanderbilt University and Mississippi State University across projects other than LQCD.

In future, we will release the framework on a public website. Attached below, is an summary of the work performed. Finally, a list of publications generated during this work is attached.

1.1 2006-2008

During the first year, Vanderbilt and Fermilab evaluated the suitability of a number of solutions OpenNMS and AWARE frameworks as a complete solution for cluster control and monitoring. Both proved to not match our requirements well. However, OpenClovis, an implementation of open specifications for service availability produced by the Service Availability Forum (SAF), did provide suitable functionality to support our needs for both controls and monitoring. However, it was no longer under development nor was it in a form which could be deployed on the existing systems.

Over the summer, the work was started by doing a preliminary investigation to classify the messages sent on the LQCD cluster support email list. This was the first step in automatically locating error-related messages that appear on the email lists.

Then the team from Vanderbilt University started working on an automated fault monitoring and mitigation system for large Lattice QCD clusters. In collaboration with colleagues at FNAL they developed and deployed sensors written in python language to monitor the critical health parameters for the nodes in the cluster. These sensors intelligently capture system health parameters and periodically report values outside of specified normal ranges to a regional node, which is responsible for storing the health data.

One of the initial problems faced during this phase was the affect of sensor executions on the physics computations being performed on the cluster. In experiments it was observed that the asynchronous execution of sensor on cluster nodes caused significant performance impact on a large MILC configuration generation job stream. This indicated the need for coordination in monitoring. The Vanderbilt team then started investigating various optimization and synchronization techniques to minimize these impacts.

FNAL and Vanderbilt also devised a classification system for monitoring data and a schema for storage in a relational database. We completed a prototype system for transfer and automatic storage of all health and cluster-related monitoring information in a database maintained at FNAL.

The personnel who contributed during this period to the cluster reliability subproject are:

- Ted Bapty, Vanderbilt
- Abhishek Dubey, Vanderbilt
- Sandeep Neema, Vanderbilt
- Don Holmgren, Fermilab
- Jim Kowalkowski, Fermilab
- Nirmal Seenu, Fermilab
- Amitoj Singh, Fermilab

1.2 2008-2009

The main part of the work in this phase was the implementation of inter-node synchronization of monitoring agents to minimize jitter that can affect performance of parallel codes. This work resulted in a publication [1]. Also, the existing code that implements heartbeat sensors was refined to increase robustness.

The prototype system (implemented in Python) for transfer and storage of monitoring information developed in the previous year was expanded, and installed in the production system at Fermilab. In addition to node health information, this system was expanded to store process accounting data and data related to batch system and MPI jobs. The latter required modifications to MPI job launch software; from this stored data, the status of the executions of all binaries associated with an MPI job can be correlated with the cluster hardware and batch system information.

This information was used by the SC LQCD II hardware project at Fermilab to determine, report, and optimize time lost on failed MPI jobs. The data accumulated over the last two years has been used in simulations to explore whether reliable predictions of node failure can be made based on various sensor readings.

During this period the team also began the design of actuators, codes which take mitigating action when failures are detected or anticipated based on the automated analysis of monitoring data. The crucial use cases and behavioral requirements for various actuators were investigated and documented. One of the key problems observed during this stage was the communication overhead. Based on the observed traffic patterns, it was decided that a hierarchical configuration (under investigation) will be necessary for scaling to thousands of nodes.

Additionally, we started to work closely with the workflow subproject during these years. An important requirement of the workflow subproject (see Fermilab and Illinois Institute of Technology, above) is that automated workflow execution must be able to recover from participant (atomic tasks, such as jobs in the batch system) failures. During this year, the reliability subproject started to define what it means for participants to succeed or fail in terms of preconditions, post-conditions and invariants.

Several papers including [2, 3, 4] were published during this phase.

The personnel who have contributed during the past period to the reliability subproject are Ted Bapty, Abhishek Dubey, and Sandeep Neema at Vanderbilt, and Jim Kowalkowski, Nirmal Seenu, Amitoj Singh, and Don Holmgren at Fermilab.

1.3 2009-2010

For the reliability sub-project, the period was spent upgrading the monitoring system developed in the previous years to a modern message passing system based on OMG Data Distribution Services DDS, prototyping plotting and interaction interfaces, and working through the details of providing fault tolerance to a workflow system.

Early in 2009 we met with the CiFITS team (Coordinated infrastructure for Fault Tolerant Systems) from Argonne National Lab to evaluate their reporting API and the sample communications implementation they provide. Their API is of interest to us because it is targeted at HPC applications, and there appears to be buy-in by some low-level software library providers (such as the Ohio State University MPI over Infiniband project, mvapich). CiFITS was a young project at the time of the meeting, with little to no testing on clusters (most testing had been on supercomputers); our team decided to perform an evaluation on clusters. Vanderbilt personnel did preliminary robustness and performance runs on their cluster, as well as an API evaluation. The CiFITS communications software as implemented lacked the robustness needed to serve as the underlying messaging system for LQCD. The API, however, can be used, and an implementation for LQCD message passing using the API has been defined. Consequently, the resulting LQCD system will also be able to receive information from other software, such as mvapich, which reports according to the CiFITS API.

Our previous prototype monitoring/control system, which uses syslog-ng as a communication mid-

Middleware layer, has been extended; this prototype system is used on the LQCD production clusters at Fermilab.

During this period, we started the work on the next version of the monitoring framework that used publish-subscribe middleware built upon the OMG Data Distribution Service standard (DDS).

This approach was taken because it allowed us to design actuators (reactive software components) as separate processes that listen to specific command data or message topics. This approach results in robust and efficient bidirectional communication. Moreover, it eliminates single points of failure because of the Reliability QoS provided by DDS. We are currently working on completing the first version of this subsystem using OpenSplice's DDS implementation. This system will serve as the underlying distributed message passing system for both the cluster reliability software system, and for communications within the LQCD workflow system.

During this period, some integration between the workflow team and the reliability team was done. This resulted in the extension of the LQCD workflow management system to become a collaboration of several components: a workflow manager, a workflow instance manager, and participant managers that track the state of job execution using timed state machines as separate threads running on the local machines. Vanderbilt applied its expertise in developing these state models to further the design of both the reliability and workflow projects. The APIs for all abstract components have been developed to include aspects necessary to ensure reliable operation: preconditions, postconditions, and invariant checks.

Exploratory work in user interface also took place during these years, including the use of Matlab, ISIS GME, and other web-based tool to show active displays of performance and alarm conditions, and also to send commands into the system for control purposes.

The personnel who have contributed during the past period to the reliability sub-project are Ted Bapty, Abhishek Dubey, and Sandeep Neema at Vanderbilt, and Jim Kowalkowski, Nirmal Seenu, and Amitoj Singh at Fermilab. Community interactions include the projects: CiFITS, OpenSplice DDS (Data Distribution Service), OpenDDS, the Fermilab component of the JDEM SOC (Joint Dark Energy Mission Science Operation Center) project, LSST Controls Group (Large Synoptic Survey Telescope), SC2009.

1.4 2010-2012

On the cluster reliability sub-project, a joint effort of Fermilab and Vanderbilt, from February 2010 through August 2010 we continued our work extending a monitoring system based upon the OMG Data Distribution Service (DDS) standard. This approach allows us to design reactive software components as separate processes that listen to specific command data or message topics.

This resulted in a fault-tolerant distributed monitoring approach (RFDMon). RFDMon can be used for measuring system variables (CPU utilization, memory utilization, disk utilization, network utilization, etc.), system health (temperature and voltage of Motherboard and CPU) application performance variables (application response time, queue size, and throughput), and scientific application data structures (PBS information and MPI variables) accurately with minimum latency at a specified rate and with controllable resource utilization.

This framework is designed to be tolerant to faults in monitoring framework, self-configuring (can start and stop monitoring the nodes and configure monitors for threshold values/changes for publishing the measurements), aware of execution of the framework on multiple nodes through HEART-BEAT messages, extensive (monitors multiple parameters through periodic and aperiodic sensors), resource constrainable (computational resources can be limited for monitors), and expandable for adding extra monitors on the fly. Since RFDMon uses a Data Distribution Services (DDS) middleware, it can be used for deploying in systems with heterogeneous nodes. Additionally, it provides a functionality to limit the maximum cap on resources consumed by monitoring processes.

We also started work on a model-based, front-end tool using the ISIS Generic Modeling Environment (GME). GME is good for designing system assemblies made up of distributed software components with well-defined interaction behaviors. The modeling and design environment allows a developer

to define process monitoring and control components, along with their resource constraints and relationships, without specifying implementation and deployment details. The component models define the customization, deployment properties and communication methods for a given platform. They also define interaction ports for a component, which include the types of data that the component produces and consumes, along with any other commands that the component may accept or emit. The tool chain allows for a model parser to generate source code for connecting the modeled components using available systems such as DDS and deployment of the modeled configuration on a physical cluster.

During this time, we also developed a formal state machine model for scientific workflow and reliability systems. This includes the use of Vanderbilt's Generic Modeling Environment (GME) tool for code generation for the production of user APIs, code stubs, testing harnesses, and model correctness verification. It is used for creating wrappers around LQCD applications so that they can be integrated into existing workflow systems such as Kepler.

Finally, during this period Fermilab personnel developed a system that tracks in close to real time the status of the various Torque batch systems used on the LQCD productions clusters at the site. This system consists of programs that actively monitor the accounting logs of Torque, noting all job transitions (queued, modified, started, ended, and so forth) and the characteristics of all jobs (nodes, times, resources) in a database. We conceived of this database as a resource required by any workflow system for determining the current and past states of jobs and nodes. However, the database and a large set of queries have proven very useful for many aspects of monitoring and administrating the LQCD clusters, including tracking progress against allocations, troubleshooting jobs and nodes, and managing allocated projects.

The personnel who have contributed during the past period to the reliability sub-project are Ted Bapty, Abhishek Dubey, and Sandeep Neema at Vanderbilt, and Jim Kowalkowski, Randy Herber, Don Holmgren, Nirmal Seenu, and Amitoj Singh at Fermilab.

Community interactions include the projects: CiFITS, OpenSplice DDS (Data Distribution Service), OpenDDS, the Fermilab component of the JDEM SOC (Joint Dark Energy Mission Science Operation Center) project, LSST Controls Group (Large Synoptic Survey Telescope).

2 Workflow

The goal for the scientific workflow project is to investigate and develop domain-specific workflow tools for LQCD to help effectively orchestrate, in parallel, computational campaigns consisting of many loosely-coupled batch processing jobs.

Major requirements for an LQCD workflow system include: a system to manage input metadata, e.g. physics parameters such as masses, a system to manage and permit the reuse of templates describing workflows, a system to capture data provenance information, a systems to manage produced data, a means of monitoring workflow progress and status, a means of resuming or extending a stopped workflow, fault tolerance features to enhance the reliability of running workflows. Requirements for an LQCD workflow system are available in documentation.

2.1 2006-2007

Fermilab and the Illinois Institute of Technology have worked very closely together since the start of the SciDAC-II LQCD project. Since September 2006, the subproject surveyed existing active workflow projects. A workshop with the VDS (now called Swift) workflow project was held at Fermilab on December 18. Video conferences were held in early 2007 with the Askalon and Kepler workflow project teams. A detailed requirements document for an LQCD workflow system was prepared jointly by IIT and Fermilab [5, 6]. This document had been used in discussions with other workflow projects and served well as a learning tool so that the various parties can understand one another. The Karajan execution engine was installed on one of the Fermilab clusters, and one of the weak decay analysis workflows had been coded in the Karajan language as a test case.

Work after the initial meetings with workflow projects was concentrated on understanding Swift and Askalon in depth. Both of these workflow systems have been installed on the Kaon cluster at Fermilab. As both workflow systems rely on GRID (Globus) tools for dispatching work, some effort was required to layer Globus tools or interfaces on top of the Torque (PBS) resource scheduler used on Kaon. In the case of Swift, we worked with the Swift developers to debug a PBS "provider" for their workflow execution engine (Karajan). This work included debugging and modifying the launcher (mpirun.rsh) from the MVAPICH MPI used on the Kaon cluster.

In both Swift and Askalon, we prototyped a few simplified LQCD workflows, specifically, configuration generation and heavy-light two-point analysis. A number of feature mismatches between these workflow systems and the LQCD workflow requirements were identified. We established working relationships with both the Askalon and Swift development teams, and both groups began working with us to evolve their products to be better suited to LQCD workflows.

The personnel who contributed during this period to the workflow subproject are:

- Jin Hui, IIT
- Pr. Xian-He Sun, IIT
- Don Holmgren, Fermilab
- Jim Kowalkowski, Fermilab
- Luciano Piccoli, Fermilab
- Nirmal Seenu, Fermilab
- Jim Simone, Fermilab
- Amitoj Singh, Fermilab

2.2 2008

Fermilab and IIT participated in the developments from November 2007 until October 2008. The efforts this year on the workflow subproject began with the emphasis on implementing candidate LQCD workflows in two existing systems, Swift (U. Chicago) and Askalon (Innsbruck), which met a number of the requirements that we had established and documented in the first year of the subproject. The candidate workflows were configuration generation [7] and heavy-light two-point analysis. Also, at the February LQCD software committee workshop in Boston, the subproject demonstrated the Chroma regression test suite implemented as Askalon and Swift workflows [8].

During the first six months, the subproject team interacted frequently with both the Swift and Askalon development groups. For Swift, the interactions focused on refining features of the Swift language so that the candidate LQCD workflows could be described completely and in a less awkward fashion. For Askalon, the work involved installation and configuration on Fermilab machines, as well as the implementation of the candidate workflows in the Askalon language. The workflow subproject produced a written report that discusses the suitability of the Kepler, Triana, JBPM, Swift, and Askalon workflow systems to our documented LQCD workflow requirements [9].

Neither Swift nor Askalon, nor the other workflow systems evaluated fully met the requirements for adequately describing and controlling LQCD analysis campaigns. Specifically, we found that they were inadequate in terms of their abilities to parameterize LQCD campaigns, to record and provide detailed run time histories, to record and provide provenance information, or to interact with secondary data storage systems. The focus of the subproject had now shifted to developing a workflow system that relies on front and back-end systems that wrap, and are independent of, existing workflow engines. The front-end software provides the parameterization features. The back-end software provides history, provenance, information necessary for job recovery, and structured storage (and access to) scientific data products. Prototype implementations of both front-end and back-end software were written using Ruby on Rails. The wrapped software was permitted

to be any of the various existing workflow engines (e.g. Swift/Karajan, Askalon, Kepler, Ruote BPM). The wrapped engine layer supervised the dispatch of individual jobs to our batch systems using a set of instructions which were assembled from our front-end system. The engines potentially permitted dispatching of jobs to systems on external clusters or grids. We used Ruote BPM as the workflow engine in our investigations.

Community interactions of the workflow subproject included the various meetings and phone conferences with the Swift and Askalon teams, papers and/or posters for SuperComputing 2007, Lattice 2008, eScience 2008 Conference (SWBES Workshop), participation in tutorials and workshops at SuperComputing 2008, and a talk submission to CHEP09.

The personnel who have contributed during the past period to the workflow sub-project are Xian-He Sun at the Illinois Institute of Technology (IIT), Luciano Piccoli of IIT and Fermilab, and Don Holmgren, Jim Simone, Jim Kowalkowski, Nirmal Seenu, and Amitoj Singh of Fermilab.

2.3 2009

For the workflow sub-project, the past period of November 2008 through January 2010 was spent iterating on earlier developments, including the design and initial implementation of new subsystems, and continuing the search for better tools. Early in the period, effort included construction of a virtual machine-based test facility, and the review of the prototype LQCD workflow system described and implemented during the previous year (Nov 1, 2007 - Nov 1, 2008). This prototype "wraps" an existing non-LQCD workflow engine (Ruote) with front- and back-end software that parametrize an LQCD user's workflow, provide persistence (job execution history, provenance, and information needed for recovery), and provide structured storage and access to scientific data products. The prototype LQCD workflow system was tested with an additional simple LQCD workflow. The effort then shifted into continuing the evaluation of other existing workflow systems, with the goal of finding a better workflow engine to wrap. The last half of the year was spent in the design of workflow system components and their interactions, and in producing a complete specification and initial implementation of a parameter set language capable of configuring all aspects of LQCD jobs.

The virtual machine (VM) test facility was created to evaluate workflow systems and their interactions with batch submission software. Many of the workflow systems previously evaluated (e.g., Kepler, Askalon) had the disadvantage of being tied to GRID software, making it difficult to install, maintain, and operate in our non-GRID environment. The VM test facility provides the environment needed to do testing without perturbing the existing LQCD production systems. This helped in our understanding of how computing cloud systems might be used to check failure handling software, and it also helped to do configuration and software testing involving multiple nodes without use of the LQCD production clusters.

The design and code review of our prototype workflow system took place in March 2009 [10, 11]; the review covered the basic workflow information storage system that was designed and started during the previous year, and also the use of the Ruote workflow engine. The major conclusions were that (1) the system provided a good start, (2) the design needed to be better documented, (3) several aspects of the design needed to be clarified, and (4) some restructuring of the code was necessary. The review report also called out the need for better testing. In response to the review, documents were added describing the systems design and operation.

Shortcomings in the Ruote workflow engine became evident during the review and during the implementation of additional test workflows. Pegasus was evaluated as a possible replacement. Lack of dynamic and flexible scheduling features, and lack of expressiveness in the Pegasus workflow language, resulted in that version of Pegasus not being able to satisfy enough of the LQCD requirements for adoption as the workflow engine.

Summer 2009 was spent developing designs for necessary workflow components and making a more complete and concise definition of concepts, APIs, and component behaviors. State models were developed to express the correct behavior of running workflow "participants" (individual tasks or functions within the workflow). Parts of the reliability sub-project became coupled to the workflow

project, including the message passing subsystem (see the Vanderbilt/Fermilab subsection). At present the reliability and the workflow designs are moving forward together, and joint meetings are held weekly.

We realized in early fall of 2009 that trying to simultaneously solve problems involving the wrapped workflow engine, and problems involving the front and back-end software, was too difficult and was hindering overall progress. We elected to narrow temporarily the scope of work and concentrate on the front and back-end software, wrapping a more limited but functional - and quite old - existing LQCD workflow engine (Runjob). As the first task, we concentrated on the complete specification of a configuration language suitable for LQCD and the tools to manipulate it. The design work was finished by the end of the year, and the first version of a parser and manipulation library is now complete. Simultaneously work was started to improve the design of the parameter set storage system in order to accommodate the new parameter sets from the configuration language, and also to clean up the code and the interface of the parameter set storage software (as suggested by the earlier review).

The personnel contributing to this workflow sub-project effort during the past period are Xian-He Sun at the Illinois Institute of Technology (IIT), Luciano Piccoli of IIT and Fermilab, and Don Holmgren, Jim Simone, Jim Kowalkowski, Nirmal Seenu, Amitoj Singh, and Randolph Herber of Fermilab. Community interaction include the following groups: CIFTS (Argonne), Pegasus (ISI at University of Southern California), Kepler (UCSD), Fermilab component of the JDEM SOC (Joint Dark Energy Mission Science Operation Center) project, ESO workflow (European Southern Observatory).

2.4 2010-2011

For the workflow sub-project during February 2010 and August 2010, a large portion of our effort was spent implementing the designs completed during the previous year. These 2009 designs include workflow system components and their interactions, and a complete specification and initial implementation of a parameter set language capable of configuring all aspects of LQCD jobs. The reliability and the workflow implementations moved forward together, and joint meetings continued to be held weekly. In the fall of 2009, we elected to narrow the scope of work and concentrate on the front and back-end software, wrapping a more limited but functional - and quite old - existing LQCD workflow engine (Runfile).

We designed and implemented a system to manage parameters (e.g. masses, algorithmic parameters) that are inputs to LQCD production campaigns. The system is capable of tracking and versioning parameter values through a relational database. A language superset of the JSON data-interchange format was developed to represent parameter sets as easily readable text. A fully functional parser, available in Ruby (as a scripting language), javascript, and in C++, was implemented. A user interface tool was developed by a Fermilab summer intern, who built a GUI editor to manipulate parameter sets based upon the Adobe Air stand-alone web application development platform. This tool was designed to be used to manage collections of parameter set files that collectively form a single configuration of any complex calculation campaign. It eases the maintenance of parameter set files that are stored in version control systems such as SVN and compliments the long-term storage database tools that are better suited for production running and provenance tracking.

Parameter values must typically be substituted into the text input prompts of existing LQCD applications such as MILC. We developed a system to create and manage text templates that represent the inputs to LQCD applications. A given template can be stored as a text in a relational database or as text in a file. Before an LQCD application is started, its input lines would be created by binding parameter values from a parameter set to a specific template. In general, both the number of input lines to an application and their content may depend upon parameter values, for example, when an application must process each item from a list of parameter values. Hence, the system permits the embedding of language constructs such as function calls and flow-control (e.g. "if-then-else" and "foreach" loops) in templates.

Interactions with the JDEM-SOC (Joint Dark Energy Mission - Scientific Operations Center)

project increased during this time. JDEM-SOC team members joined the weekly LQCD meetings and contributed to the workflow engine investigation. They have demonstrated the use of Kepler as a workflow engine. Members from both teams started to define a working model of the interactions between three critical pieces: the program (e.g. MILC), the actor (the management piece of this job), and the workflow engine scheduling element ("director" in Kepler terminology).

The personnel contributing to this workflow sub-project effort during the past period are Luciano Piccoli of IIT and Fermilab, and Don Holmgren, Jim Simone, Jim Kowalkowski, Nirmal Seenu, Amitoj Singh, and Randolph Herber of Fermilab. Community interaction include the following groups: Kepler (UCSD), and Fermilab component of the JDEM SOC (Joint Dark Energy Mission Science Operation Center) project. Early in the year Piccoli left IIT and Fermilab. Herber has taken on some of his work.

As of July 2011, formal definitions of scientific workflow system components and functions were created to clarify and further the requirements list, and also to reconcile them with other Fermilab projects that involve workflow systems. In addition, an analysis of several common MILC input decks was done to discover configuration practices and rules for this tool. This resulted in finding a few anomalies in production run scripts and also the release of a tool to help diagnose configuration problems. This work also resulted in new documentation for configuration card ordering, dependencies, and desk building rules. A new real use case was also documented and added to the collection of workflows.

We formalized the parameter definition and management system (named fhicl). Fhicl started out as being largely JSON compatible, and has diverged only slightly to accommodate LQCD and other HEP requirements. The fhicl language and C++ binding were already being used within other Fermilab HEP Intensity Frontier projects as a common language for configuring scientific applications. A file system storage and retrieval system was added. These tools were packaged for use as a library. Many features were added, including parameter value substitution, file includes, and parameterized values. A Python binding was delivered for processing fhicl files. The Ruby binding was improved.

A template engine was developed to move LQCD analysis applications toward being usable in workflow systems. This engine was integrated into the "run_onium" scripts. It permits sophisticated parameterization, including generation of blocks of input desk cards using looping constructs. Such a facility permits removal of a portion of the hardcoded processing logic. Tools were also developed to help with integration with the fhicl parameter definitions. A workflow participant (scientific application participating in a workflow) protocol was developed, documented, and tested. Several LQCD application-specific graph execution engines were put together to test this protocol.

We evaluated Kepler as a candidate workflow execution engine and modeling environment and worked with the Kepler team to bring a demonstration system up. This work was done in conjunction with another Fermilab project.

A workflow modeling and execution system was established using mature Vanderbilt ISIS Generic Modeling Environment (GME) toolkit. The system was demonstrated using one of the sample use cases weve collected. The system incorporated and tested several of the concepts and protocols weve developed, including the participant protocol and state machine process behavior management. GME allows for multiple aspect modeling, so the correct program behavior (states) can be modeled independent of the workflow components (participants), and also independent of the deployment on a cluster (node assignments).

3 Achievement highlights

3.1 Extension period

The extension period lasted from August 2011 through December 2011. During that time, the Ruby gem system was set up for deployment of the Ruby code. The Ruby and Python parsers for FHICL were documented. Most of the available time was spent on documenting, expanding, and

debugging the monitoring database for tracking jobs and the associated command line interface tools.

3.2 Uses outside LQCD

3.2.1 FHICL

FHICL, introduced as part of these projects, was introduced to the Intensity Frontier (IF) experiments at Fermilab. After a series of refinements to the syntax and semantics of the language, it was accepted into the main offline software framework (called art) used by the IF experiments. It is currently used as the configuration language for mu2e, NOvA, g-2, and the experiments using the LArSoft (Liquid Argonne Software) package. A full-featured C++ binding was developed and is distributed as a standard package.

3.2.2 Wrapper protocol

The wrapper protocol that was started within the LQCD workflow project permits embedding (or adopting) existing external applications into a workflow execution engine and processing environment. The wrapper protocol was taken and refined for use in the JDEM (Joint Dark Energy Mission) Science Operation Center, a Cosmic Frontier project assigned to Fermilab. The protocol was expanded to use FHICL. More recently this protocol was also used as the basis for the engineering of the DES (Dark Energy Survey) workflow software framework. The design elements were reused because they demonstrate a clean separation between coordination functions and functions that carry out the science.

3.2.3 Problem analysis reuse

In addition to the wrapper protocol, the basic set of requirements, goals, and system architecture for the LQCD workflow and reliability projects have been reused in discussions with additional experiments. The requirements and the system architecture formed a starting point for work that was done on JDEM. They have also served as a starting point for the reengineering discussions with DES. They have been heavily relied on in discussions with LSST on a potential level-3 toolkit project.

3.3 Summary

- Implemented a software system to manage parameters. This includes a parameter set language based on a superset of the JSON data-interchange format, parsers in multiple languages (C++, Python, Ruby), and a web-based interface tool. It also includes a templating system that can produce input text for LQCD applications like MILC.
- Implemented a monitoring sensor framework in software that is in production on the Fermilab USQCD facility. This includes equipment health, process accounting, MPI/QMP process tracking, and batch system (Torque) job monitoring. All sensor data are available from databases, and various query tools can be used to extract common data patterns and perform ad hoc searches. Common batch system queries such as job status are available in command line tools and are used in actual workflow-based production by a subset of Fermilab users.
- Developed a formal state machine model for scientific workflow and reliability systems. This includes the use of Vanderbilt's Generic Modeling Environment (GME) tool for code generation for the production of user APIs, code stubs, testing harnesses, and model correctness verification. It is used for creating wrappers around LQCD applications so that they can be integrated into existing workflow systems such as Kepler.

- Implemented a database system for tracking the state of nodes and jobs managed by the Torque batch systems used at Fermilab. This robust system and various canned queries are used for many tasks, including monitoring the health of the clusters, managing allocated projects, producing accounting reports, and troubleshooting nodes and jobs.

4 List of Publications

- [1] A. Dubey, G. Karsai, and S. Abdelwahed, “Compensating for timing jitter in computing systems with general-purpose operating systems,” in *ISORC*, 2009, in press.
- [2] A. Dubey, S. Nordstrom, T. Keskinpala, S. Neema, T. Bapty, and G. Karsai, “Towards a model-based autonomic reliability framework for computing clusters,” in *EASE ’08*, 2008, pp. 75–85.
- [3] A. Dubey, S. Neema, J. Kowalkowski, and A. Singh, “Scientific computing autonomic reliability framework,” in *eScience*, 2008, in press.
- [4] L. Piccoli, A. Dubey, J. N. Simone, and J. B. Kowalkowski, “Lqcd workflow execution framework: Models, provenance and fault-tolerance,” *Journal of Physics: Conference Series*, vol. 219, no. 7, 2010. [Online]. Available: <http://stacks.iop.org/1742-6596/219/i=7/a=072047>
- [5] LQCD Workflow Team. (2006, December) Lqcd workflow functional requirements. [Online]. Available: <https://cdcvs.fnal.gov/redmine/attachments/6332/FunctionalRequirements.pdf>
- [6] —. (2009, February) Glossary of workflow terms. [Online]. Available: <https://cdcvs.fnal.gov/redmine/attachments/6363/LQCD-Glossary.pdf>
- [7] —. (2008, March) Confgen analysis: Managing parameters, provenance, and secondary products. [Online]. Available: <https://cdcvs.fnal.gov/redmine/attachments/6338/ConfGenAnalysis-v2.pdf>
- [8] —. (2008, February) Workflow systems for lqcd. [Online]. Available: <https://cdcvs.fnal.gov/redmine/attachments/6348/LQCD-Workflows-SciDAC-swc-2008.pdf>
- [9] —. (2009, October) Workflow evaluation: Swift and askalon. [Online]. Available: <https://cdcvs.fnal.gov/redmine/attachments/6334/WorkflowEvaluation.pdf>
- [10] E. N. W.E. Brown, D. Holmgren and M. Paterno. (2009, March) Lqcd workflow (code) mini-review. [Online]. Available: <https://cdcvs.fnal.gov/redmine/attachments/6336/MiniReviewMarch09.pdf>
- [11] L. Piccoli. (2009, February) Lqcd workflow code review intro. slides. [Online]. Available: <https://cdcvs.fnal.gov/redmine/attachments/6366/LQCD-code-review-slides.pdf>
- [12] R. Mehrotra, A. Dubey, S. Abdelwahed, and W. Monceaux, “Large scale monitoring and online analysis in a distributed virtualized environment,” in *Engineering of Autonomic and Autonomous Systems (EASe)*, 2011 8th IEEE International Conference and Workshops on, april 2011, pp. 1–9.
- [13] R. Mehrotra, A. Dubey, J. Kwalkowski, M. Paterno, A. Singh, R. Herber, and S. Abdelwahed, “Rfdmon: A real-time and fault-tolerant distributed system monitoring approach,” , 10 2011.
- [14] A. Dubey, S. Nordstrom, T. Keskinpala, S. Neema, T. Bapty, and G. Karsai, “Towards a verifiable real-time, autonomic, fault mitigation framework for large scale real-time systems,” *Innovations in Systems and Software Engineering*, vol. 3, pp. 33–52, March 2007.

- [15] A. Dubey, L. Piccoli, J. B. Kowalkowski, J. N. Simone, X.-H. Sun, G. Karsai, and S. Neema, “Using runtime verification to design a reliable execution framework for scientific workflows,” in *Proceedings of the 2009 Sixth IEEE Conference and Workshops on Engineering of Autonomic and Autonomous Systems*, ser. EASE ’09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 87–96. Online: <http://dx.doi.org/10.1109/EASE.2009.13>
- [16] S. Nordstrom, A. Dubey, T. Keskinpala, R. Datta, S. Neema, and T. Bapty, “Model predictive analysis for autonomic workflow management in large-scale scientific computing environments,” in *EASE ’07*, 2007, pp. 37–42.
- [17] L. Piccoli, X.-H. Sun, J. Simone, and et al., “The lqcd workflow experience: What we have learned,” in *SuperComputing 2007*, 2007.
- [18] L. Piccoli, J. Simone, J. Kowalkowski, and et al., “Tracking lqcd workflows,” in *Lattice 2008*, 2008.
- [19] L. Piccoli. (2006, October) Workflow project. Online: <https://cdcvns.fnal.gov/redmine/attachments/6341/wf-slides-boston.pdf>
- [20] L. Piccoli and J. Kowalkowski. (2008, September) Workflow project status. Online: <https://cdcvns.fnal.gov/redmine/attachments/6350/SciDAC-SEP-11-2008.pdf>