FireWorks and the Materials Project: Facilitating Scientific Workflows

Morgan Hargrove
Louisiana State University
Baton Rouge, LA
mhargrove@cct.lsu.edu

Anubhav Jain, Dan Gunter
Lawrence Berkeley National Laboratory
Berkeley, CA
{ajain,dkgunter}@lbl.gov

ABSTRACT
This paper describes the design, development, and real-world application of BaseSite, a front-end user interface for the scientific workflow system FireWorks.

Categories and Subject Descriptors
D.2.2 [**Software Engineering**]: Design Tools and Techniques – *evolutionary prototyping, user interfaces.*

General Terms
Design, Human Factors.

Keywords
User interfaces; scientific workflow system; queries; performance dashboard.

1. INTRODUCTION
As scientific research and experimentation move more towards a system of data-driven, distributed collaboration, the need for innovative new tools to facilitate these processes becomes readily apparent. One such tool is FireWorks, a general purpose scientific workflow management system that is currently being used by the Materials Project. BaseSite was developed in order to simplify the use of FireWorks by providing a visual way to easily monitor the system's status.

2. BACKGROUND

2.1 The Materials Project
Technological innovation – faster computers, more efficient solar cells, more compact energy storage – is often enabled by materials advances. Yet, it takes an average of eighteen years to move new materials discoveries from the research laboratory to the market. This is largely because materials designers operate with very little information and must painstakingly tweak new materials in the lab. Computational materials science is now powerful enough that it can predict many properties of materials before those materials are ever synthesized in the lab.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

The Materials Project [1,2,3] aims to accelerate materials innovation and design by providing free access to a searchable, interactive database of computed materials properties spanning most known inorganic compounds. By scaling materials computations over supercomputing clusters, the Materials Project has computed some properties of over 80,000 materials, helping to remove guesswork from materials design in a variety of applications. The current Materials Project implementation runs as a 24/7 production system with over 2,500 registered users.

Researchers are able to use innovative design tools provided by the Materials Project to data-mine scientific trends in materials properties. Experimental research can then be targeted to the most promising compounds from computational data sets.

2.2 e-Science
Computation is currently accepted as the third paradigm of science, alongside experimentation and theory. In sciences from biology to particle physics, hundredfold to thousandfold increases in the data from simulations and post-analysis of experiments has caused a fourth paradigm to emerge, as proposed by Jim Gray [4]. This fourth paradigm, referred to as "e-Science," lays the foundation for exploratory data-driven science to complement traditional hypothesis-driven research through emphasis on a collaborative, networked, and data-driven framework.

By utilizing a distributed system of computing resources, the Materials Project is pioneering e-Science in the materials science community. Its data-centered framework allows scientists to collaborate on large-scale experiments and derive scientific insight towards materials science in a way that is not possible with more traditional scientific tools.

2.3 Scientific Workflow Systems
In order to enable the creation of reliable, reproducible results within e-Science, scientists have begun developing scientific workflows to track the evolution of their data, results, and discoveries. A scientific workflow system is designed to compose or execute a series of computational or data manipulation steps, also known as a workflow. Scientific workflows are a crucial element that provide a way for scientists to model, design, execute, and re-run their experiments.

Computations for the Materials project are currently driven by a workflow management system based on the high-throughput framework FireWorks [5].

2.4 FireWorks

FireWorks is a Python [6] based, open-source code for defining, managing, and executing scientific workflows. It can be used to automate most types of calculations over arbitrary computing resources, including those that have a queuing system. FireWorks is intended to be a friendly workflow software that is easy to get started with but flexible enough to handle complicated use cases. Basic operation of FireWorks is simple, and it can be run on resources ranging from a single laptop to a supercomputing center.

2.4.1 Features and Limitations

Some, but not all, of FireWorks' features include storage management of workflows through MongoDB (a noSQL datastore that is flexible and easy to use) [7], a clean and powerful Python API for defining, submitting, running, and maintaining workflows, the ability to distribute calculations over multiple worker nodes, each of which might use a different queuing system and process a different type of calculation. FireWorks provides support for dynamic workflows that react to results programmatically, allowing for users to pre-specify code that performs actions, such as terminating a workflow, adding a new step, or completely altering the workflow based on the output of a job. FireWorks also features automatic detection of duplicate sub-workflows, skipping duplicated portions between two or more workflows while still running unique sections.

FireWorks has not yet been stress-tested on an extremely large scale (hundreds of jobs within a single workflow or millions of separate workflows). FireWorks also does not automatically optimize the distribution of computing tasks over worker nodes in order to minimize data movement or to match jobs to appropriate hardware; however, users can define these such optimizations themselves. To date, FireWorks has only been tested on Linux and Macintosh machines (not Windows platforms).

2.4.2 Workflow Model

Workflows in FireWorks are composed of three main components: FireTasks, FireWorks, and Workflows. FWActions may also exist between FireWorks.

2.4.2.1 FireTasks

A FireTask is an atomic computing job. It can call a single shell script or execute a single Python function that is defined by the user, either within FireWorks or in an external package. Non-Python code, such as C++ or Java, can be run by calling the code as a shell script or writing a Python function that executes your code. Each FireTask receives input data in the form of a JavaScript Object Notation (JSON) [8] specification, or "spec."

2.4.2.2 FireWorks

A FireWork contains the JSON spec that includes all the information needed to create a job. The spec typically contains an array of FireTasks to execute in sequence as well as any input parameters to pass to your FireTasks. The same function can easily be performed over different input data by creating FireWorks with identical FireTasks and different input parameters in the spec. Specs can be designed however the user would like, as long as it's valid JSON. The JSON format used for FireWork specs is extremely flexible, very easy to learn, and immediately makes rich searches over the input data available to end users through MongoDB's JSON document search capabilities.

2.4.2.3 Workflows

A workflow is essentially a set of FireWorks with dependencies between them. Workflows can be designed to be as simple as requiring a "parent" FireWork to finish and generate output files before running "child" FireWorks or as complex as automatically re-running a crashed job at a different FireWorker with somewhat different parameters.

2.4.2.4 FWActions

An FWAction that can store data or modify the Workflow depending on the output may be returned between FireWorks. An FWAction can pass data to the next step, cancel the remaining parts of the Workflow, or even add new FireWorks that are defined within the object.

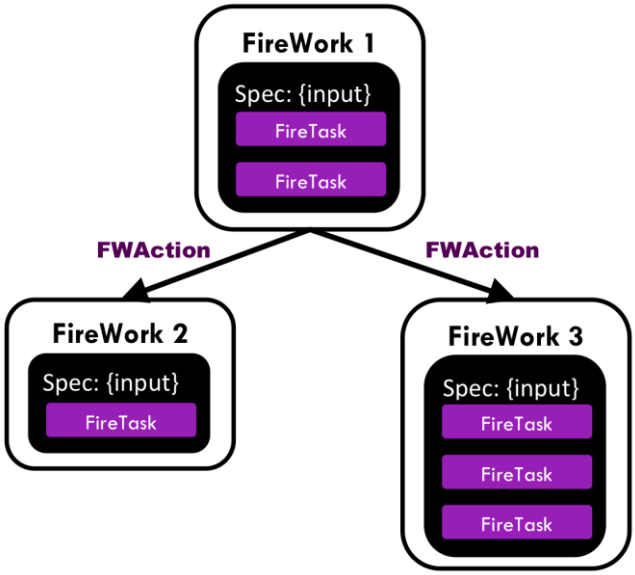

Figure 1. The FireWorks Workflow model

More sophisticated types of workflow operations can be used to send different categories of FireWorks to different FireWorkers, get the status of all existing jobs, where they're running, and how long they took to run or waited in the queue, create and modify job priorities, or even handle job failures and crashes dynamically by automatically creating FireWorks that fix crashed jobs through an FWAction.

2.4.3 Basic Infrastructure

FireWorks has a loosely-coupled, modular infrastructure that is intentionally hackable, allowing users to utilize some FireWorks components without using everything in order to more easily adapt it to their individual application(s).

FireWorks follows a centralized server and worker model, meaning that there are two essential components to a FireWorks installation: the LaunchPad and one or more FireWorkers.

2.4.3.1 LaunchPad

LaunchPad is the central server in FireWorks. LaunchPad's primary function is to manage workflows. When a user adds new workflows, queries for the state of workflows, or re-runs existing workflows, these commands are carried out through the LaunchPad.

2.4.3.2 *FireWorkers*

FireWorks depends upon one or more workers, known as "FireWorkers," to run jobs. The FireWorkers request workflows from the LaunchPad, execute them, and send back information. A FireWorker can be as simple as the same workstation used to host the LaunchPad or as complex as a national supercomputing center with a queuing system.

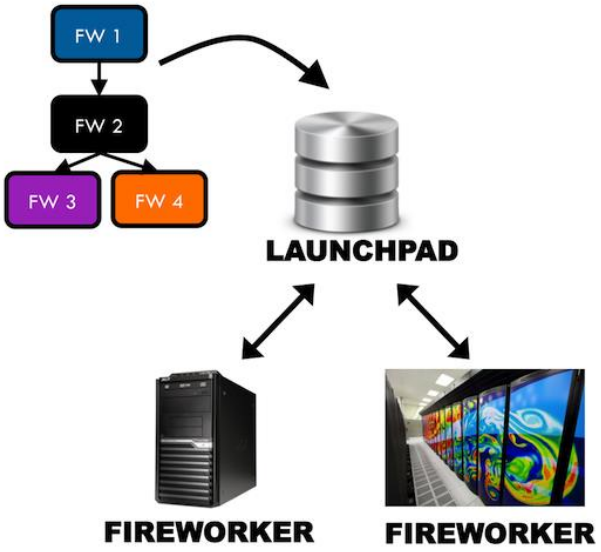

Figure 2. The basic FireWorks infrastructure

These components are largely decoupled, making FireWorks generally easier to use. End users can add new workflows to the LaunchPad without worrying about the details of how and where the workflows will be run, though they have the option to tailor the details of job execution if they wish. This keeps the workflow specifications lightweight, tidy, and easy to learn and use. On the opposite end, administrators can configure worker computers without worrying about where workflows are coming from or what they look like, although they can easily and flexibly assign jobs to certain resources if desired. Running FireWorks on a heterogeneous set of worker computers is simple because it uses essentially the same internal code to run on a simple workstation as it does to run at a large supercomputing center.

3. APPROACH

BaseSite is the newly developed front-end user interface for FireWorks. BaseSite aims to facilitate the use of FireWorks in real-world scenarios, such as the Materials Project, by assisting collaboration, particularly among users who may not have experience or be comfortable with coding. This is accomplished by providing a simple, visual way to monitor the system's status. Though BaseSite was initially designed for use with the Materials Project, it can be utilized for any FireWorks application.

3.1 Design and Implementation

BaseSite is a website that serves as a performance dashboard for monitoring the status of a FireWorks system A dashboard is a visual display of the most important information needed to achieve one or more objectives, consolidated and arranged on a single screen so the information can be monitored at a glance [9]. Effective dashboards are easy to understand and can be utilized by users as well as administrators.

3.1.1 *Framework*

BaseSite was developed using Django [10], an open source, high-level Python Web framework that encourages rapid development and clean, pragmatic design, making it particularly useful for the creation of complex, database driven websites.

3.1.2 *Querying Data*

BaseSite gathers information by using MongoDB queries to directly access the database that is associated with a particular FireWorks system. For example, in the Materials Project application of BaseSite, data is collected from a FireWorks production database hosted at NERSC [11]. These queries are run every time BaseSite is viewed, resulting in a live display of the status of the current database.

3.2 Features

BaseSite includes several different views and features, including a home page, index pages, and various information pages. Different sections of content can be clicked on to easily navigate to more detailed information, or URLs can be used to directly access desired content.

3.2.1 *Home Page*

The BaseSite home page acts as the primary dashboard for information monitoring. It features a count of the total completed FireWorks to date as well as several different charts depicting the current database status and important information about the most recent FireWorks and Workflows.

Current Database Status		
	Fireworks	Workflows
ARCHIVED	110,079	23,189
DEFUSED	5,283	2,501
WAITING	2,534	
READY	765	612
RESERVED	2	
FIZZLED	359	138
RUNNING	19	78
COMPLETED	257,133	64,658
TOTAL	376,174	91,176

Figure 3. A table found on the home page, depicting counts of FireWorks and Workflows based on their current status. "Defused" refers to jobs that were canceled, and "fizzled" refers to jobs that have unexpectedly failed.

3.2.2 *Index Pages*

BaseSite contains two index pages, one for FireWorks and one for Workflows. Each of these pages initially loads with information about the most recently updated FireWorks or Workflows.

Newest Fireworks		
ID	Name	State
376175	O2_Si1--VASP_db_insertion	WAITING
376174	O2_Si1--GGA_static	READY
376173	O2_Si1--VASP_db_insertion	WAITING
376172	O2_Si1--GGA_band_structure_v2	WAITING
376170	O2_Si1--VASP_db_insertion	WAITING
376168	O2_Si1--GGA_Uniform_v2	WAITING
376166	O2_Si1--VASP_db_insertion	READY
376164	O2_Si1--GGA_static_v2	FIZZLED
376171	O2_Si1--VASP_db_insertion	WAITING
376169	O2_Si1--GGA_band_structure_v2	WAITING

1 of 37618 > >>

Figure 4. A table on the FireWorks index page

This information can be further sorted using simple click navigation to view only FireWorks or Workflows of a certain state.

Fizzled Workflows	
ID	Name
362280	Be1 Na2 O6 Si2
361552	Pd3 Rb2 Se4
156491	Cl1 S1
360320	Mn2 Na2 S3
359364	O4 Pb2 Sn1
359673	O3 Sm2
357643	O2 Si1
354169	F6 K2 Zr1
340777	Cl1 H5 N2
149024	F6 Hf1 K2

<< < 9 of 14 > >>

Figure 5. A table displaying information about Fizzled Workflows

3.2.3 Information Pages

Every FireWork or Workflow ID in BaseSite can be clicked on to navigate to a page containing its JSON spec. Users can choose to view the spec with varying levels of detail: less, more, or all.


```
{
    "name": "Mg1_O3_Si1--VASP_db_insertion",
    "fw_id": 354099,
    "state": "COMPLETED",
    "created_on": "2013-06-30T10:42:03.523022"
}
```


Figure 6. The JSON spec for the FireWork with ID 354099, displaying "less" information

The entirety of this data includes all of the input that was specified at the time of the FireWork or Workflow's creation, which directory the run occurred in, how long the run took, and several other important pieces of information.

4. FUTURE WORK

An alpha version of BaseSite is currently available as part of the FireWorks installation. Plans for future work include the addition of different charts and visualizations, simplification of the JSON information pages, improved overall site navigation, and the implementation of a search function to more easily locate data of interest.

5. ACKNOWLEDGMENTS

This work was supported in part by TRUST (Team for Research in Ubiquitous Secure Technology), which receives support from the National Science Foundation (NSF award number CCF-0424422), with additional support from Lawrence Berkeley National Laboratory and the U.S. Department of Energy.

Special thanks to Anubhav Jain and Dan Gunter for their mentorship and guidance throughout the duration of this project.

6. REFERENCES

[1] The Materials Project. http://www.materialsproject.org/.

[2] Jain, A., Ong, S.P., Hautier, G., Chen, W., Richards, W.D., Dacek, S., Cholia, S., Gunter, D., Skinner, D., Ceder, G., and Persson, K.A. 2013. The Materials Project: A materials genome approach to accelerating materials innovation. *Applied Physics Letters Materials*, 2013, 1(1), 011002. DOI= http://dx.doi.org/10.1063/1.4812323/.

[3] Gunter, D., Cholia, S., Jain, A., Kocher, M., Persson, K., Ramakrishnan, L., Ong, S.P., and Ceder, G. 2012. Community Accessible Datastore of High-Throughput Calculations: Experiences from the Materials Project. 5th workshop on Many-Task Computing Grids and Supercomputers (MTAGS), 2012.

[4] Gray, J. 2007. E-Science: A Transformed Scientific Method. NRC-CSTB talk (Mountain View, CA, January 11, 2007).

[5] FireWorks 0.18 documentation. http://pythonhosted.org/FireWorks/.

[6] Python Programming Language. http://www.python.org/.

[7] A Brief Introduction to MongoDB. http://www.10gen.com/static/downloads/ mongodb introduction.pdf.

[8] Introducing JSON. http://www.json.org/.

[9] Few, S. 2006. *Information Dashboard Design: The Effective Visual Communication of Data.*

[10] Django: The Web framework for perfectionists with deadlines. https://www.djangoproject.com/.

[11] National Energy Research Scientific Computing Center: NERSC. http://www.nersc.gov/.