Remote Execution of Scientific Workflow Software on NASA’s High-End Computing Capability Systems

Instructor: Jia Zhang
Authors: Owen Chu, Clyde Li, David Liu, Kate Liu, Norman Xin
Semester: Fall 2012

I. Introduction

The mission of NASA's High-End Computing Capability (HECC) [1] project is to accelerate and enhance scientific discovery and aeronautics research. The motivation for our project is to allow scientists to access the HECC supercomputing environment with minimal knowledge of its operational aspects and modification to their workflows.

Our project aims to provide an infrastructure, designed for integration with VisTrails [2] and potentially other scientific workflow management software to help domain scientists seamlessly leverage HECC's computing and storage resources. In this technical report, we describe our proposed architecture and the implementation of a working prototype to demonstrate the feasibility of our solution.

II. Background

The HECC Project, hosted by NASA, is a world-class computing and storage environment for conducting large-scale scientific research experiments to support NASA's missions. Scientists access the environment by using two-factor (SSH+RSA) logging mechanism to front-end nodes and issue jobs to compute nodes. The NASA Earth Exchange (NEX) [3] facility assists earth scientists in their research by collecting global and high-resolution satellite data and providing high computing systems to share with the geoscience community. Geoscientists use these environmental data sets in conducting their research, such as creating models to predict natural phenomena. Because the information is high-density, comes from all around the world, and encompasses many years, scientists require high computing power to complete their experiments in a reasonable amount of time.

A. NASA HECC Overview

1) Front-End Nodes

The front-end layer of the system contains 14 Pleiades Font-Ends (PFEs) and 2 bridge nodes. These nodes provide environments for users to perform file transfers, file manipulations, and job submissions. Users are required to first log onto secure front-end nodes first with SSH to be able to log on to one of the 14 Pleiades Font-Ends (PFEs) and 2 bridge nodes using RSA authentication.

2) Portable Batch System

Pleiades contains the Portable Batch System (PBS) developed by Altair for all compute job submissions, monitoring, and management. PBS adopts job queues to manage pending work and acts as a scheduler. It dispatches jobs to be run on one or more compute nodes, based on factors such as mission shares (a certain percentage of CPU's on Pleiades are allocated to each NASA mission directorate), job priority, queue priority, and job size. After users log on to the front-end nodes, they are able to issue qsub/qstats command to access the PBS to submit jobs and receive job statuses, respectively.

3) Compute Nodes

Four kinds of computing nodes are currently available on Pleiades [1]. Users can specify the node type and process numbers in the PBS script. An overview of the node specifications is as follows:

Node Type	Number of Nodes	Processors per Node	Processor Speed	Memory Size per Core
Sandy Bridge	1,728	2 eight-core processors	2.6 GHz	2 GB
Westmere	4,608	2 six-core processors	2.93 GHz or 3.06 GHz	2 GB
Nehalem	1,280	2 quad-core processors	2.93 GHz	3 GB
Harpertown	4,096	2 quad-core processors	3 GHz	1 GB

B. Vistrails

VisTrails is a scientific workflow management software package used in data-related research. In our project, value is gained by creating a solution to allow VisTrails to directly submit workflows as jobs to be processed in the NASA's Pleiades system, since high computing power is necessary in processing the scientists' data-intensive requests.

III. Architecture and Design

A. Scope

Our project's scope is the interface between the scientists and the high-performance computing layer of the Pleiades system. The goal is to provide scientists with a client integrated into VisTrails to submit processing requests to be run at NASA's Pleiades systems and receive updates on the status of their processes. We gather these processes in a remote server, which will also contain a scheduler apart from that from the HPC layer.

B. Concerns

The scalability of the server holding the requests is a concern, because it could be a potential architectural bottleneck. Scientists submit processes and receive updates to and from the remote server through VisTrails. Therefore, increased traffic to this remote server could impede or disrupt scientists from being able to submit their processes and receive updates. A potential solution could be to increase the number of remote servers and allow VisTrails to choose the remote server to use based on the number of current requests of each server.

The availability of a MapReduce solution is another concern, because there are currently no openly-available ways to split a VisTrails program into a MapReduce program; without such a plug-in, the VisTrails program can only be run as a batch job and not in parallel. A public Hadoop solution could not be found either. A solution to this concern could be to write an independent MapReduce solution, but it would be very resource-intensive.

C. Principles

The three key principles in our architectural design are the blackboard, client/server, and publisher/subscriber models. The blackboard architecture model will be used to collect different scientists processing jobs to a central server. The goal of this is to decouple the scientists from the HPC servers, and instead, handle the scheduling in a remote scheduling server. This remote server will contain a 'blackboard' of requests to be processed and a scheduler that employs an algorithm to launch scripts from a Pleiades Front-End (PFE) node. The client/server model will be used as well. Specifically, a thin client and fat server will be used to increase responsiveness on the client side. The scheduling will be moved to the designated scheduler server mentioned earlier. The publisher/subscriber model is used to increase responsiveness of the system. Scientists will subscribe to the scheduler server and receive notifications on the status and progress of their processing requests.

D. Constraints

A constraint is the accessibility of PBS through the PFE nodes. Accounts must be granted to access these nodes. Therefore, to begin development of our solution, we need to gain developer access to NASA's computing resources. In prototyping the solution, the different nodes in the Pleiades system and the servers involved in the plug-in can be simulated. However, access of NASA's nodes is required to test the integration of the solution more comprehensively.

E. Design Views

1) Functional View

As illustrated in Figure 1, the functional view consists of a front, middle, and back tier. The front tier provides three approaches for users to leverage NASA's HECC supercomputing resources: VisTrails, Direct Access, and API. (As our project focuses on the integration with VisTrails, the other approaches are only shown for reference and comparison purposes.) The HECC plug-in will be implemented as a VisTrails module and seamlessly adapt the VisTrails workflow to the HECC computing environment.

The middle tier receives compute requests from the front tier, fairly distributes the requests to the Scheduler Servers, and schedules the requests to run in the back tier. The Scheduler Service gathers the back tier's status through the Job Queue Monitor and the Compute Node Monitor, and schedules the requests according to the loading of the backend systems. The Job Status Monitor is responsible for reporting the status of the requests to the HECC plug-in.

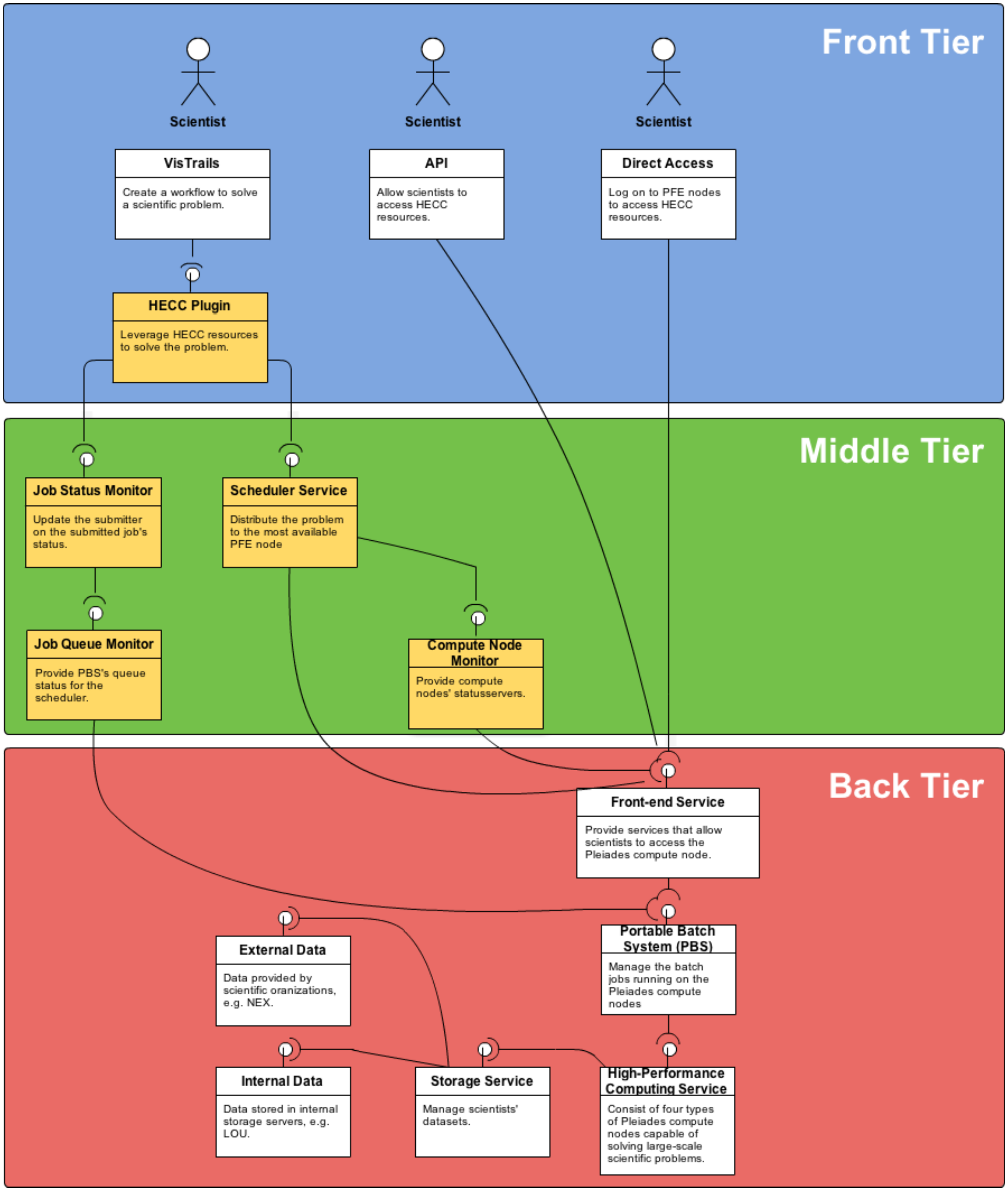

Front Tier
Scientist
Scientist
Scientist
VisTrails
Create a workflow to solve a scientific problem.
API
Allow scientists to access HECC resources.
Direct Access
Log on to PFE nodes to access HECC resources.
HECC Plugin
Leverage HECC resources to solve the problem.
Middle Tier
Job Status Monitor
Update the submitter on the submitted job's status.
Scheduler Service
Distribute the problem to the most available PFE node
Job Queue Monitor
Provide PBS's queue status for the scheduler.
Compute Node Monitor
Provide compute nodes' statusservers.
Back Tier
Front-end Service
Provide services that allow scientists to access the Pleiades compute node.
External Data
Data provided by scientific oranizations, e.g. NEX.
Portable Batch System (PBS)
Manage the batch jobs running on the Pleiades compute nodes
Internal Data
Data stored in internal storage servers, e.g. LOU.
Storage Service
Manage scientists' datasets.
High-Performance Computing Service
Consist of four types of Pleiades compute nodes capable of solving large-scale scientific problems.

Figure 1. Functional View

2) Deployment View

The deployment view contains the following 7 types of nodes:

Node Type	**Description**
VisTrails	The VisTrails node runs the VisTrails application to solve scientific problems. The HECC plug-in enables the VisTrails application to leverage HECC's computing resources.
Scheduler Server	The scheduler server coordinates the usage of HECC computing resources. The server connects to HECC's front-end and bridges servers via the SSH protocol and communicates with the Scheduler Agent to dispatch compute jobs requested by scientists. Scalability could be addressed by adding more scheduler servers and designing a mechanism to synchronize their status.
Pleiades Front-End Server	Pleiades has 14 front-ends servers which allow scientists to log on and submit compute jobs. The scheduler server accesses these servers on behalf of the scientists
Pleiades Bridge Server	Pleiades has 2 bridge servers which allow scientists to log on and submit compute jobs. They contain more memory than front-end nodes. The scheduler server accesses these servers on behalf of the scientists
Pleiades PBS Server	PBS (Portable Batch System) maintain different queues to manage the batch jobs that run on the four type of compute servers.
Pleiades Compute Server	Pleiades has 4 types of compute servers, each with different levels of computing capability.
Pleiades Storage Server	Storage servers allow scientists to save and retrieve their data.

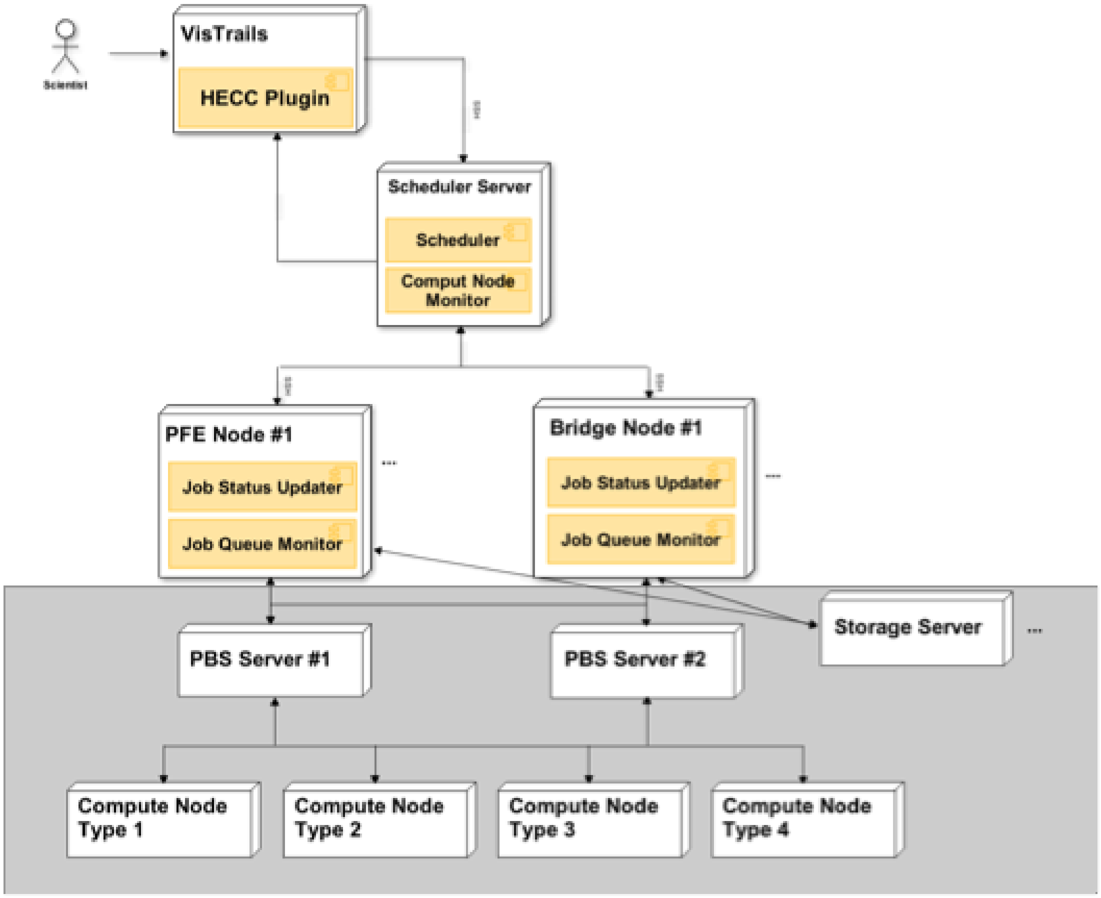


Figure 2. Deployment View

3) Development View

The development view describes how different software components in the system interfaces with each other. This high-level view of the system is mainly for software development and management purposes.

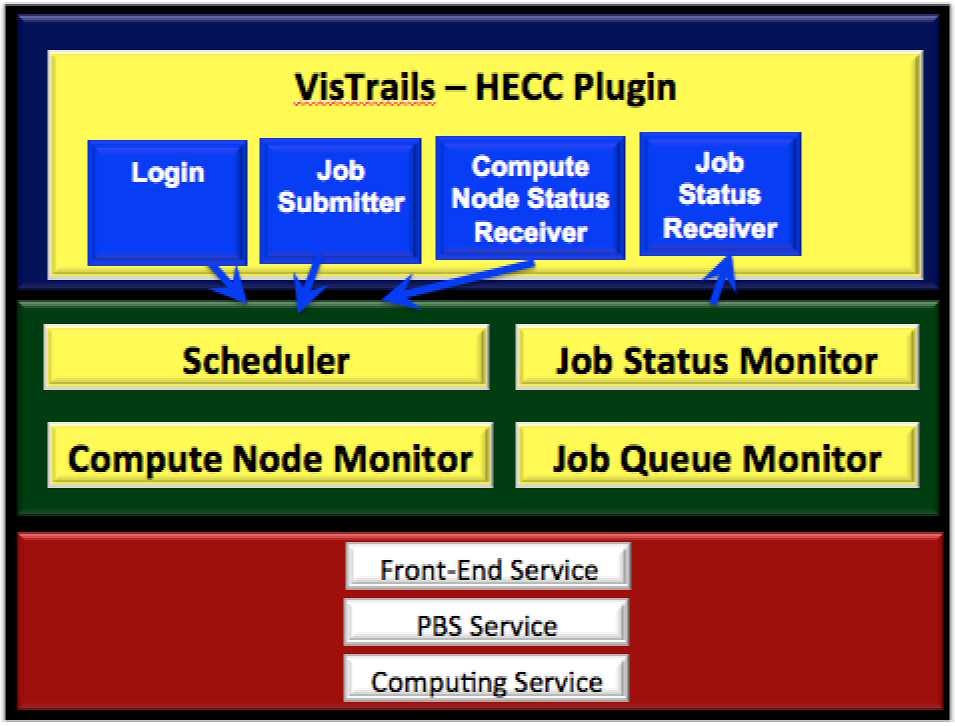


Figure 3. Development View

IV. Prototype Implementation

A prototype of the proposed system is implemented used to demonstrate the design of 3-tier software architecture of HECC plug-in. In spite of the fact that it is still a prototype, it implements the full set of usable features on the client side as a VisTrails plug-in. These features include user login, job status monitoring, and job scheduling. A Scheduler Server is also implemented to receive compute job requests and generate the PBS script. Along with the Scheduler Server, we also simulated the two-level logging process and the job execution to mimic the real situation since we now only have limited access to the NASA HECC environment.

A. VisTrails Plug-In

As described in previous sections, VisTrails is the tool current used by NEX scientists that provides a graphical interface for designing and managing workflows. Due to its plug-in support [4], it has great extensibility for new features. Python is the language for developing VisTrails plug-ins and the GUI framework is built on QT. The implemented VisTrails plug-in for the prototype adds the following new menu items to the current menu:

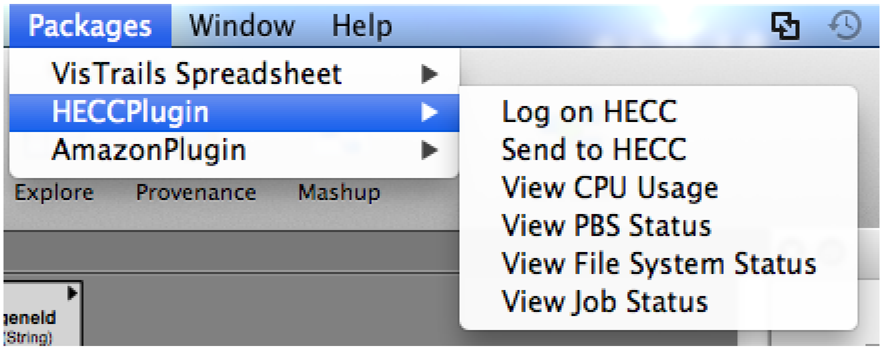


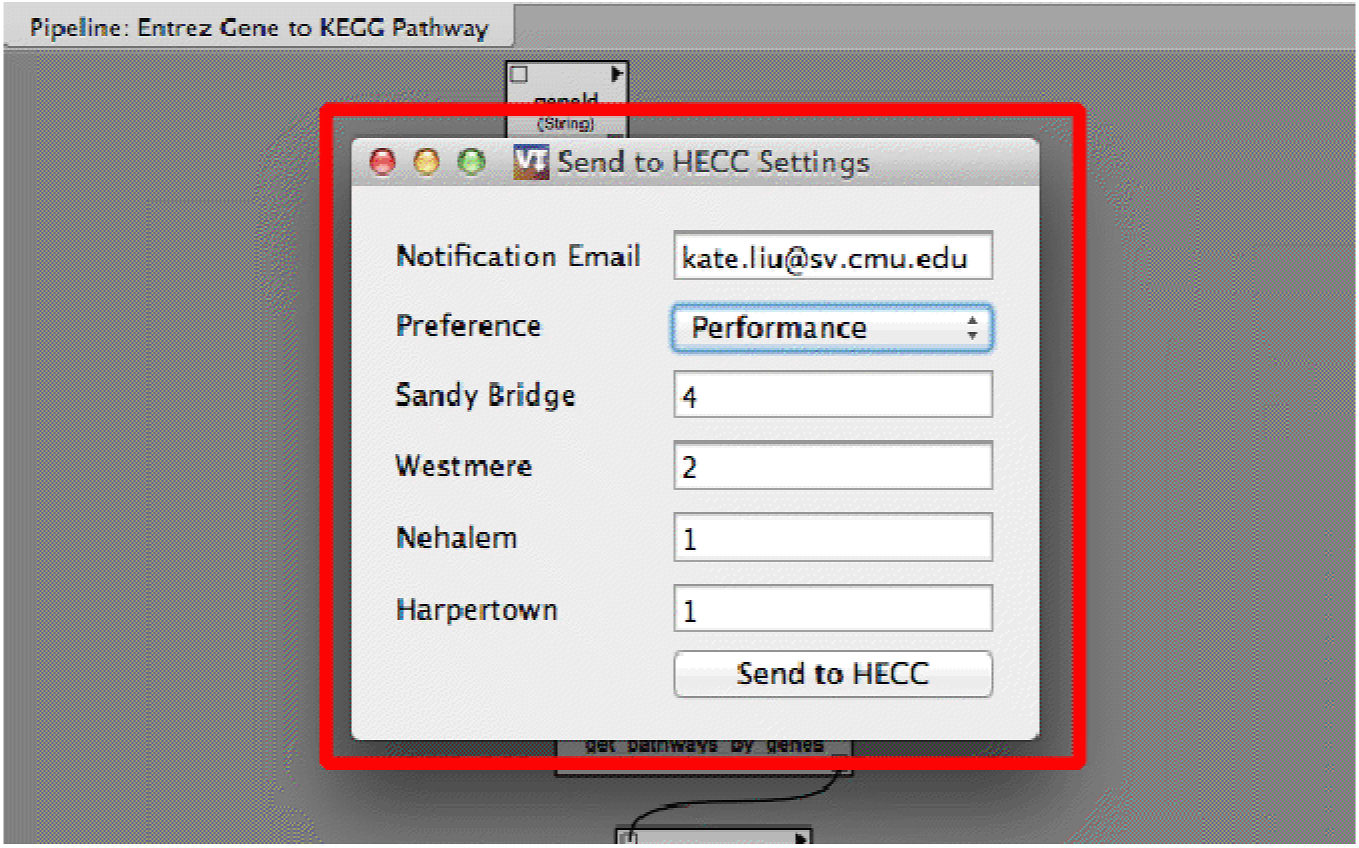
Figure 4. VisTrails Plug-in Menu Items Screenshot

When the workflow design is done, the user can sign on to HECC with the "Log on HECC" menu item. This part is done by calling the functions provided by the VisTrails "remoteLogin" plug-in. (http://www.vistrails.org/index.php/UserContributedPackages). The included pexpect python package further facilitates the program to detect and react with RSA authentication process.

Users are also allowed to use "View CPU Usage", "View PBS Status" and "View File System Status" to fetch the current status of HECC. Currently, these three menu items directly renders the following page on a QWebView component:

- CPUs in use: http://www.nas.nasa.gov/monitoring/hud/realtime/pleiadespanel1.html
- PBS Jobs: http://www.nas.nasa.gov/monitoring/hud/realtime/pleiadespanel2.html
- Filesystem Usage: http://www.nas.nasa.gov/monitoring/hud/realtime/pleiadespanel3.html

Finally, the user can select "Send to HECC" to open a dialog box that provides automatic or manual computing node selection; the two automatic options available are performance and cost. The job can then be sent to be run on HECC by clicking on the "Send to HECC" button. The plug-in makes this possible by uploading the current VisTrails project file and a generated configuration file to the server. An user can send multiple jobs to the Scheduler Server and afterwards, use the "View Job Status" option to retrieve job statuses.



Send to HECC Settings

Notification Email: kate.liu@sv.cmu.edu

Preference: Performance

Sandy Bridge: 4

Westmere: 2

Nehalem: 1

Harpertown: 1

Figure 5. VisTrails Plug-in “Send to HECC” Screenshot

B. Scheduler Server

The prototype of the Scheduler Server is currently deployed on an Amazon EC2 Ubuntu instance. It monitors new workflow jobs sent from VisTrails users and generates PBS scripts according to the user’s configuration. In the prototype, we also run these jobs on the same server as we do not have access to HECC. After the compute job is done, the server then automatically sends a notification email with the link of results to the user.

In general, the prototype implements and covers most of the proposed architecture functions specified by requirements. It reflects usability, security, and extensibility, which are the essential quality attributes for this project.

V. Future Work

To further achieve the integration with NASA HECC system, we are looking forward to have the opportunity to gain more accessibility to NASA's resources. We hope this prototype is a good start and proof of our architectural design for the project. We believe that our work is worthwhile to expand to another level and we can help NASA increase the number of scientists adopting their computing ecosystem.

Appendix A: Source Code

https://github.com/clydeli/HECCAdapter

Appendix B: Installation

Directions to enable VisTrails to run in batch mode:

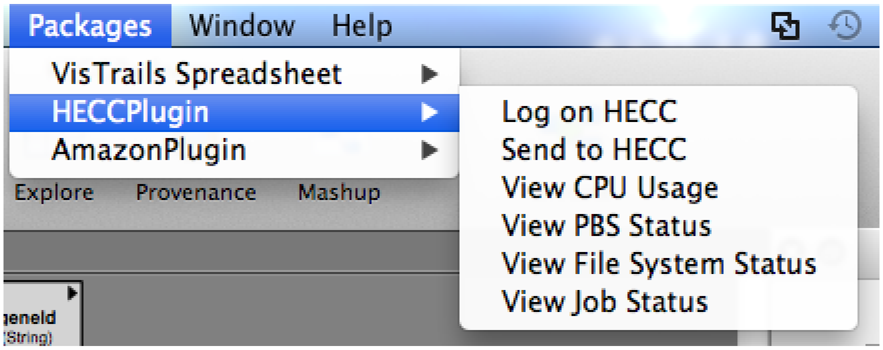
1. sudo apt-get install xvfb python-matplotlib python-suds
2. wget http://downloads.sourceforge.net/project/vistrails/vistrails/v2.0.1/vistrails-src-2.0.1-5e35e2b83b90.tar.gz
3. tar -zxvf vistrails-src-2.0.1-5e35e2b83b90.tar.gz
4. Running a workflow
    a. See scripts/run_vistrails_batch_xvfb.sh for more details
5. Debugging VisTrails installing or runtime issues:
    a. Check VisTrails's log file: ~/.vistrails/vistrails_2_0_1.log

Installing VisTrails and Modules:

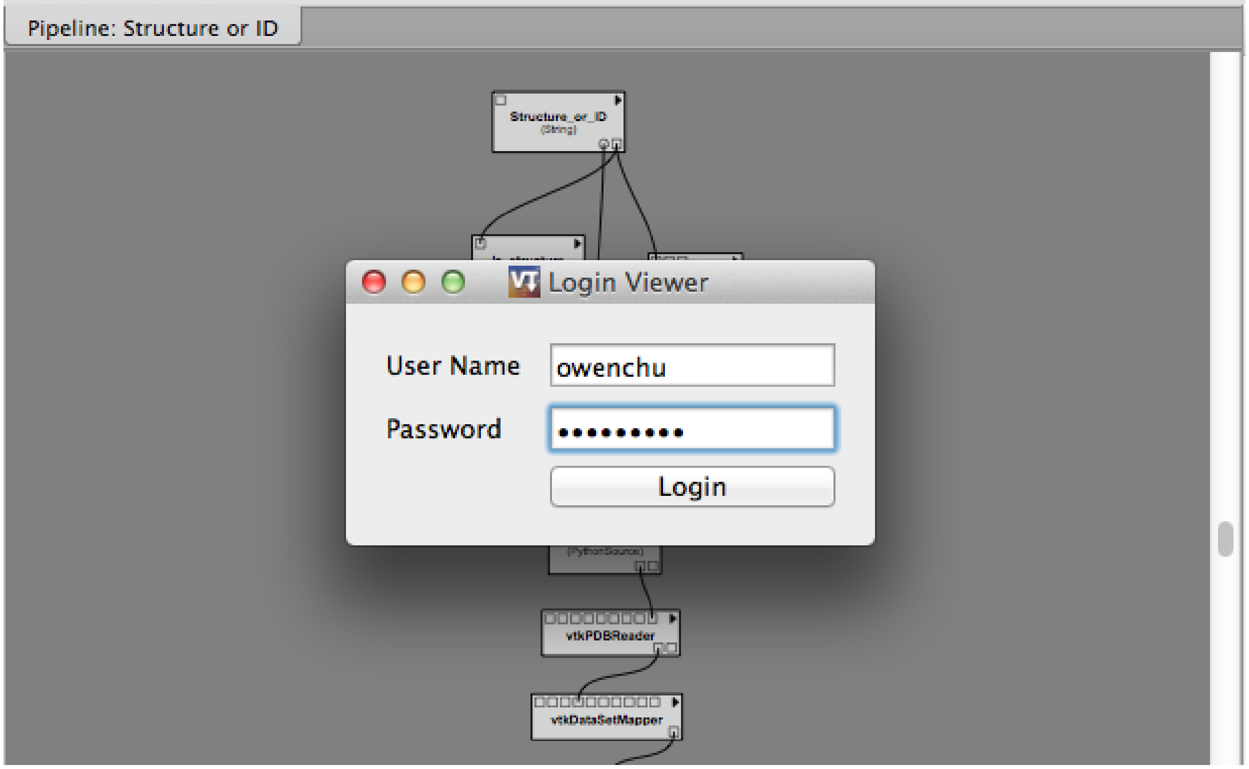
1. http://www.vistrails.org/index.php/Downloads
2. Download the HECCPlugin and AmazonPlugin folders from Git
3. Copy the two folders into ~/.vistrails/userpackages
4. On VisTrails, go to VisTrails > Preferences and click on the Module Packages tab
5. Enable the HECCPlugin and AmazonPlugin modules

Appendix C: Module Navigation

HECC module menu items:



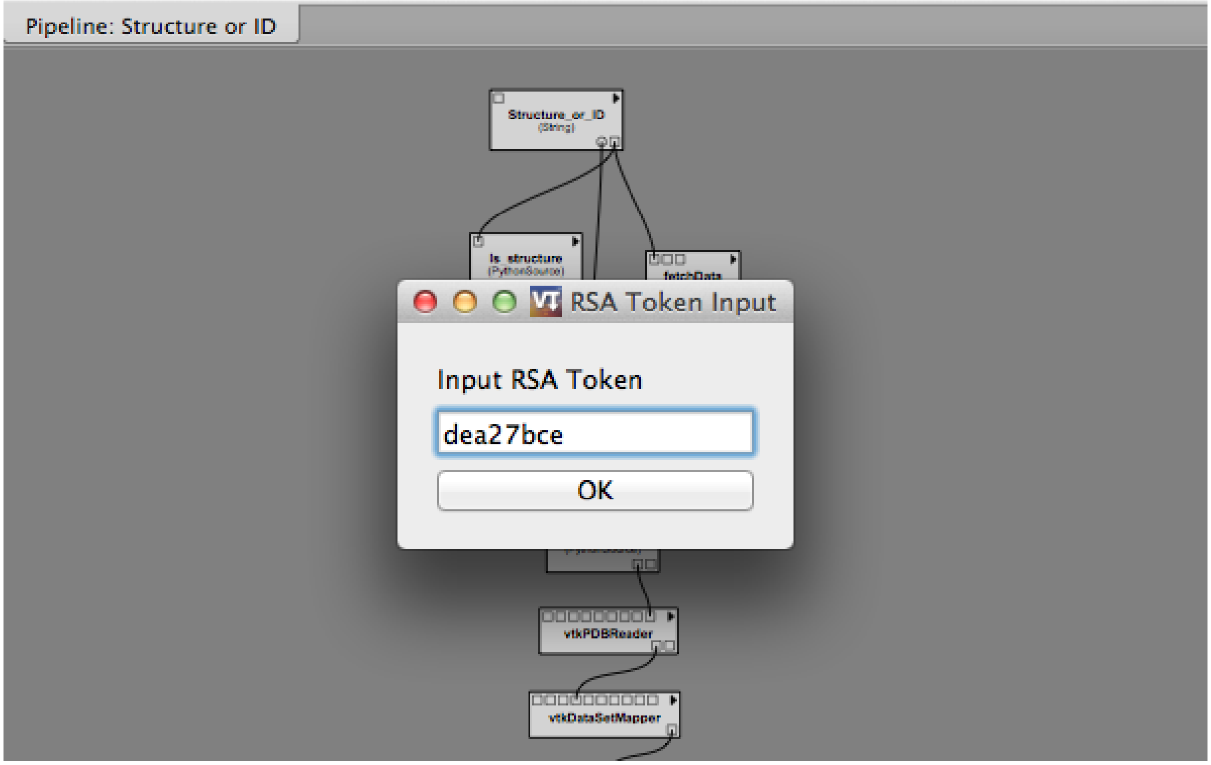
Login window for HECC. This window will show up when the user selects “Log on HECC”.



User Name: owenchu

Password: •••••••••

RSA token window. The user enters the code that appears on their RSA token into this window. This window will show up after the user enters account information from the user login window or selects “Send to HECC”.

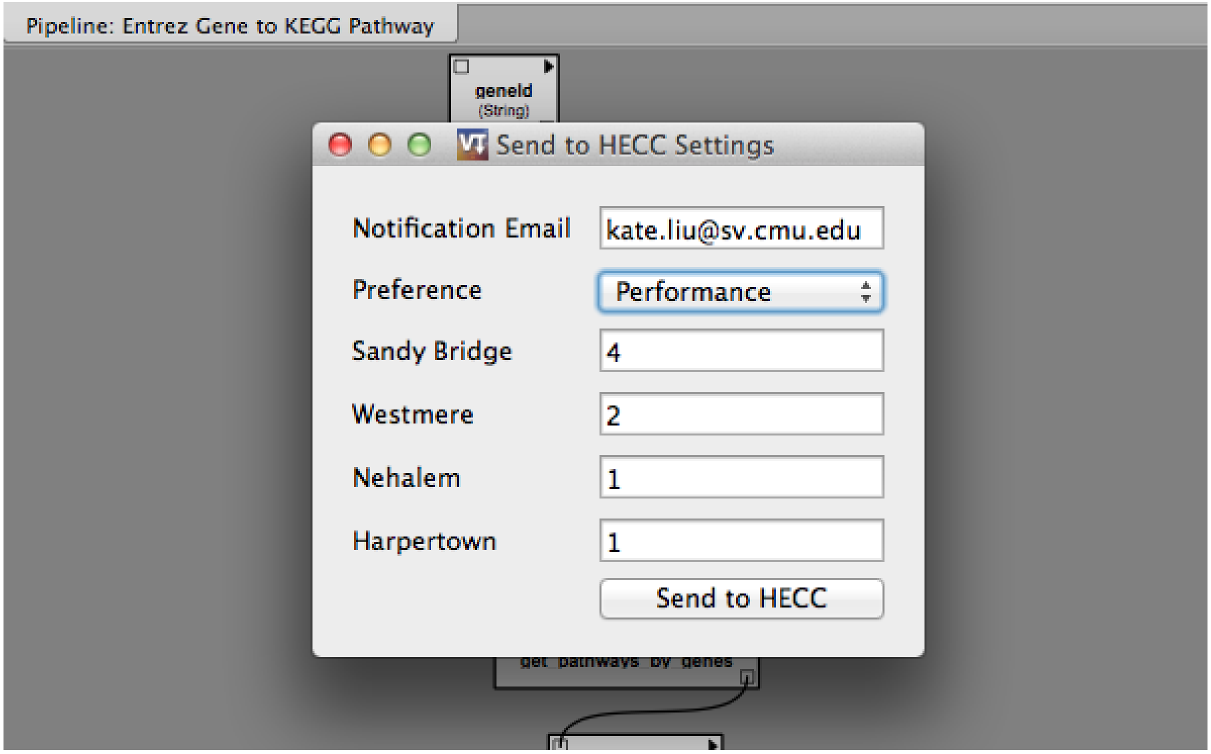


RSA Token Input

Input RSA Token

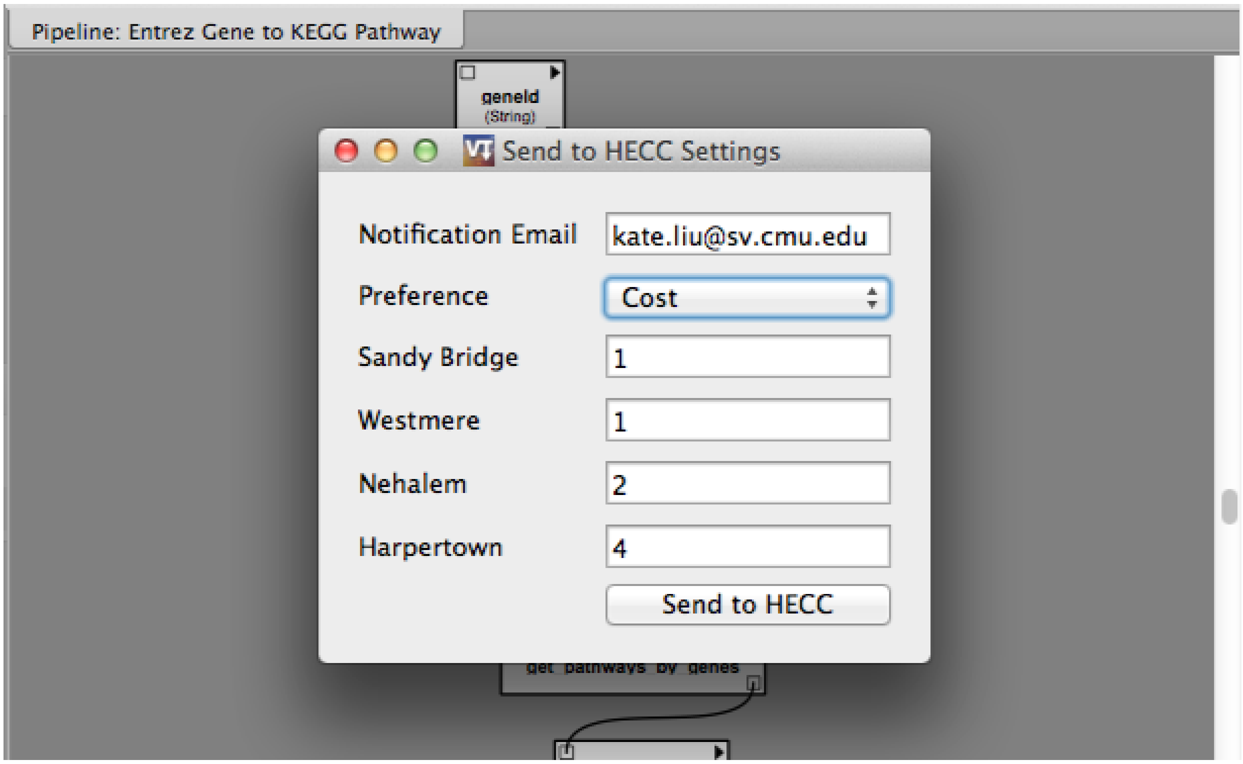
dea27bce

HECC settings window. This window will show up when the user selects “Send to HECC” and puts their RSA token. The user can choose the email address they want to receive the job completion notification. They can also select from 3 different preferences, which will change the number of nodes in the ‘Sandy Bridge’, ‘Westmere’, ‘Nehalem’, and ‘Harpertown’ inputs. The ‘Performance’ preference is selected below. When the user is done, the user can press ‘Send to HECC’ to send the job to be run remotely.

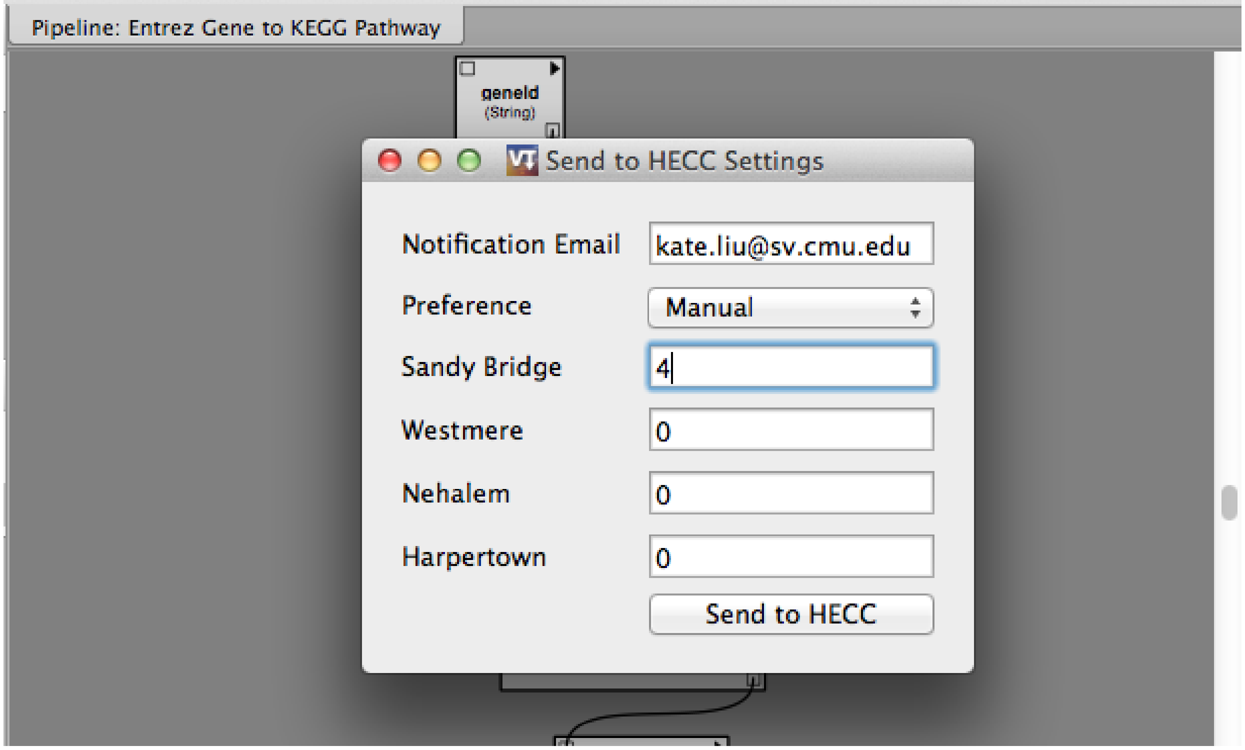


Notification Email	kate.liu@sv.cmu.edu
Preference	Performance
Sandy Bridge	4
Westmere	2
Nehalem	1
Harpertown	1

The ‘Cost’ preference is selected below. The functionality is the same as mentioned above.



Notification Email	kate.liu@sv.cmu.edu
Preference	Cost
Sandy Bridge	1
Westmere	1
Nehalem	2
Harpertown	4



Send to HECC Settings

Notification Email: kate.liu@sv.cmu.edu

Preference: Manual

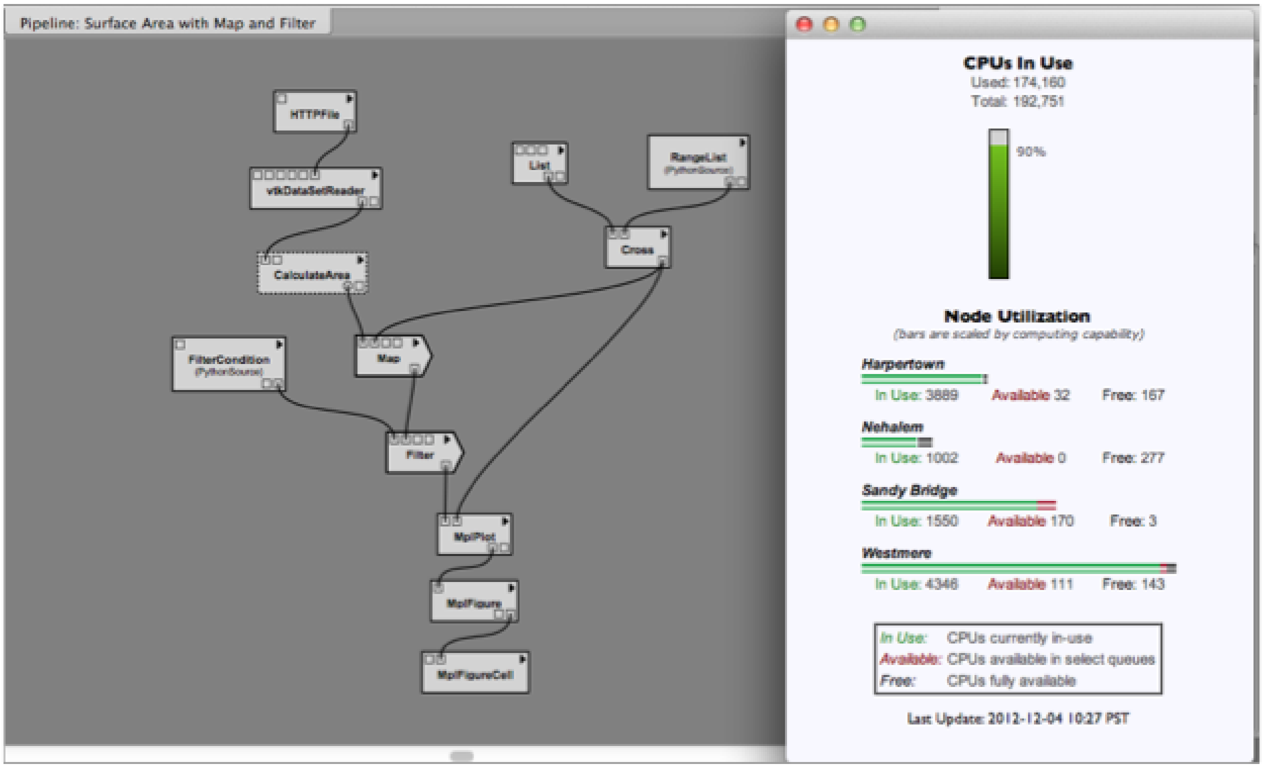
Sandy Bridge: 4

Westmere: 0

Nehalem: 0

Harpertown: 0

This window will show up when the user selects ‘View CPU Usage’. It shows the usages of the different compute nodes in the Pleiades system.



CPUs In Use

Used: 174,160

Total: 192,751

90%

Node Utilization

(bars are scaled by computing capability)

Harpertown

In Use: 3889 Available 32 Free: 167

Nehalem

In Use: 1002 Available 0 Free: 277

Sandy Bridge

In Use: 1550 Available 170 Free: 3

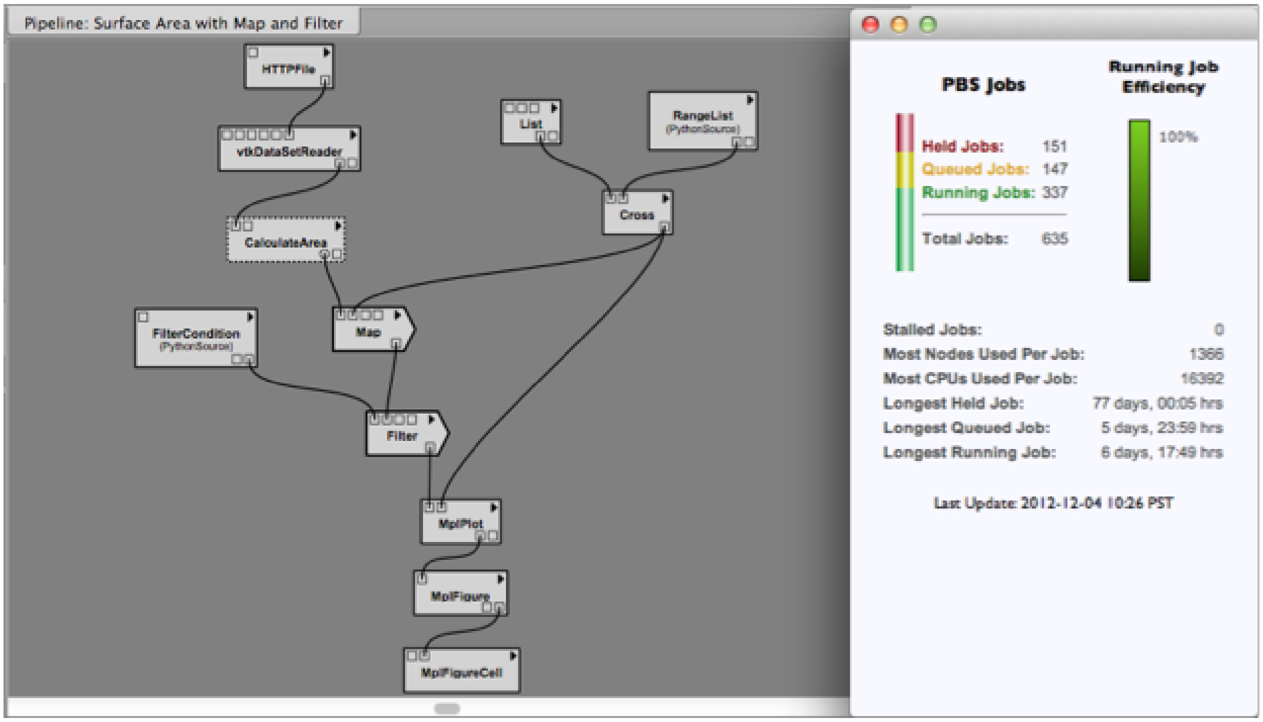
Westmere

In Use: 4346 Available 111 Free: 143

In Use: CPUs currently in-use
Available: CPUs available in select queues
Free: CPUs fully available

Last Update: 2012-12-04 10:27 PST

This window will show up when the user selects ‘View PBS Status’. It shows statuses and information of jobs handled by PBS in the Pleiades system.



PBS Jobs

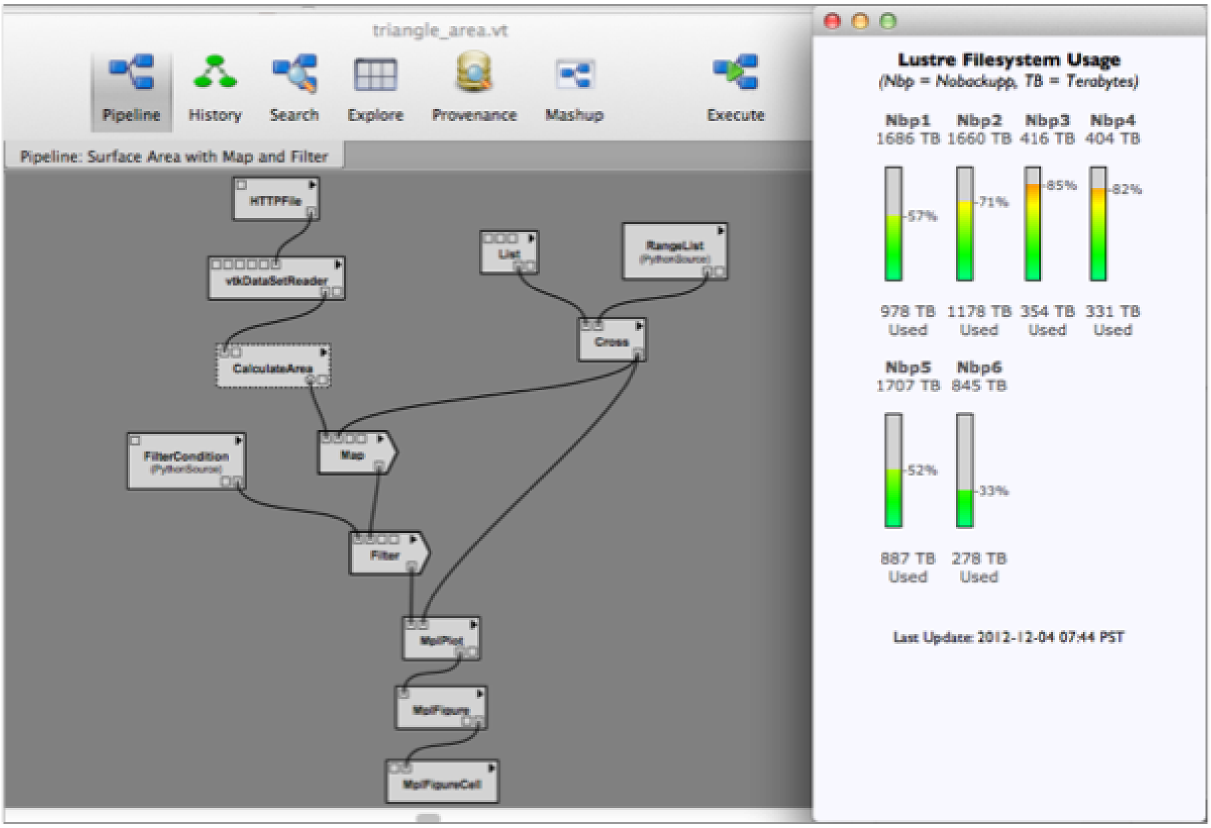
Held Jobs:	151
Queued Jobs:	147
Running Jobs:	337
Total Jobs:	635

Running Job Efficiency

Stalled Jobs:	0
Most Nodes Used Per Job:	1366
Most CPUs Used Per Job:	16392
Longest Held Job:	77 days, 00:05 hrs
Longest Queued Job:	5 days, 23:59 hrs
Longest Running Job:	6 days, 17:49 hrs

Last Update: 2012-12-04 10:26 PST

This window will show up when the user selects ‘View File System Status’. It shows the statuses of the file system nodes.



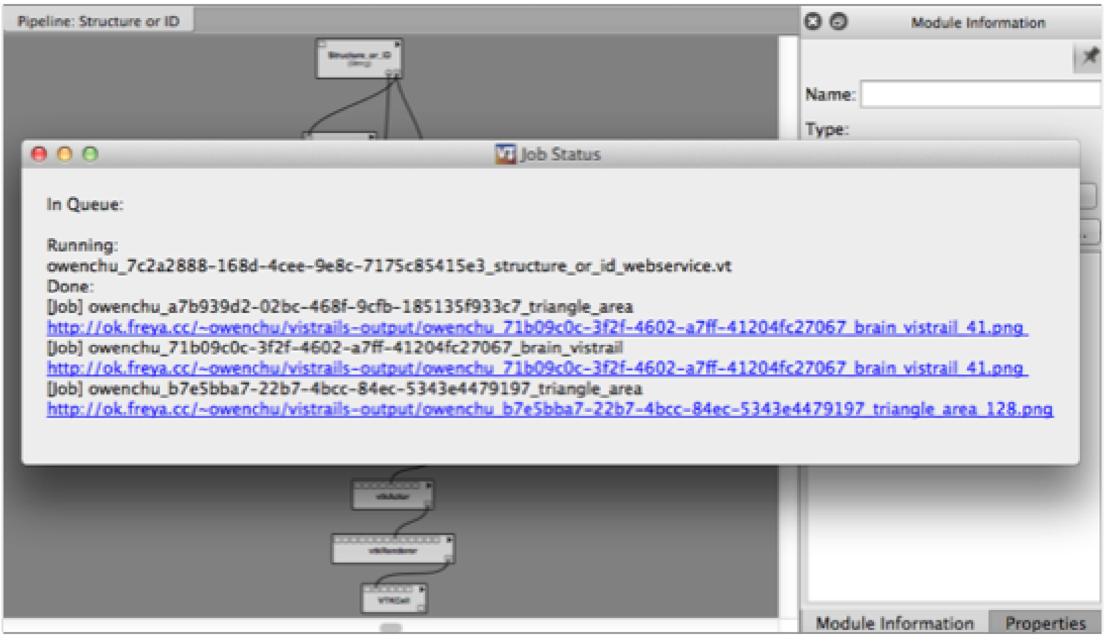
Lustre Filesystem Usage
(Nbp = Nobackupp, TB = Terabytes)

Nbp1	Nbp2	Nbp3	Nbp4
1686 TB	1660 TB	416 TB	404 TB
978 TB Used	1178 TB Used	354 TB Used	331 TB Used

Nbp5	Nbp6
1707 TB	845 TB
887 TB Used	278 TB Used

Last Update: 2012-12-04 07:44 PST

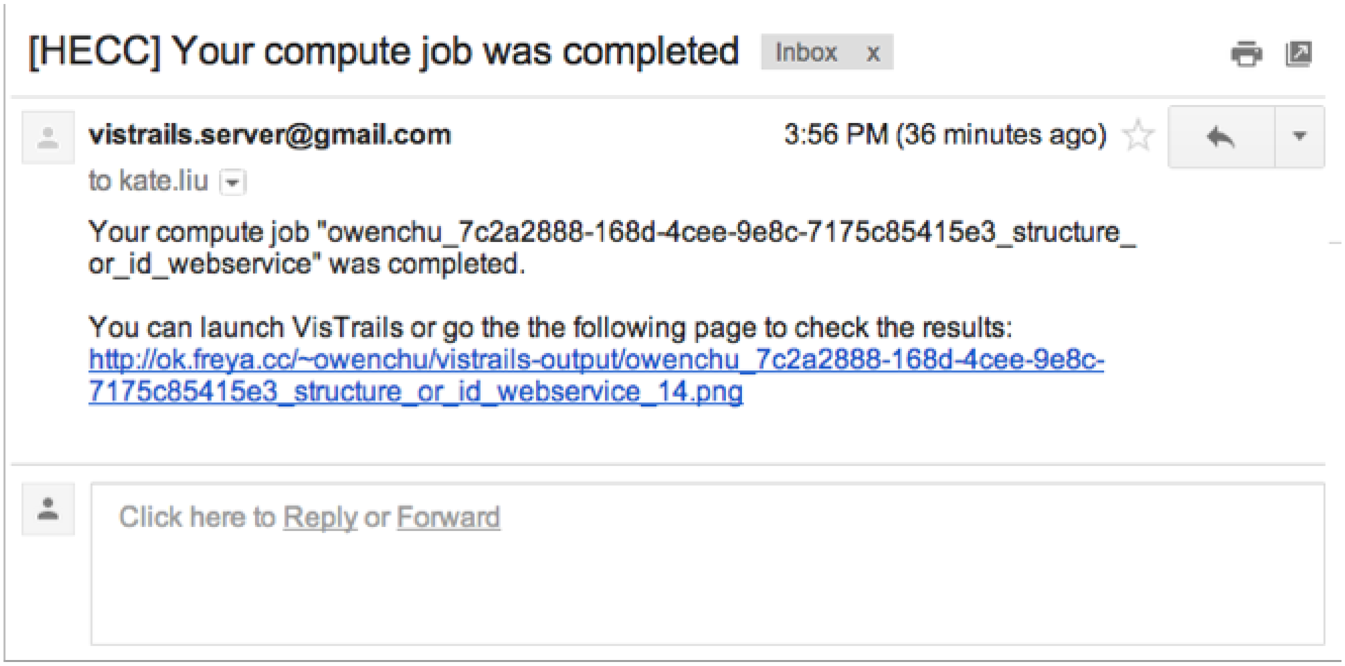
This window shows up when the user selects 'View Job Status' after a job has been submitted to HECC. It provides links to results that are viewable online.



In Queue:

Running:
owenchu_7c2a2888-168d-4cee-9e8c-7175c85415e3_structure_or_id_webservice.vt
Done:
[Job] owenchu_a7b939d2-02bc-468f-9cfb-185135f933c7_triangle_area
http://ok.freya.cc/~owenchu/vistrails-output/owenchu_71b09c0c-3f2f-4602-a7ff-41204fc27067_brain_vistrail_41.png
[Job] owenchu_71b09c0c-3f2f-4602-a7ff-41204fc27067_brain_vistrail
http://ok.freya.cc/~owenchu/vistrails-output/owenchu_71b09c0c-3f2f-4602-a7ff-41204fc27067_brain_vistrail_41.png
[Job] owenchu_b7e5bba7-22b7-4bcc-84ec-5343e4479197_triangle_area
http://ok.freya.cc/~owenchu/vistrails-output/owenchu_b7e5bba7-22b7-4bcc-84ec-5343e4479197_triangle_area_128.png

This is an example of the email a user receives when a job is completed.



[HECC] Your compute job was completed Inbox x

vistrails.server@gmail.com 3:56 PM (36 minutes ago)

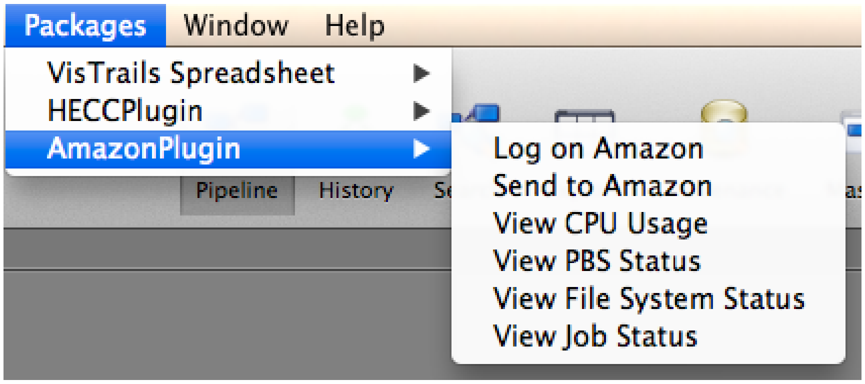
to kate.liu

Your compute job "owenchu_7c2a2888-168d-4cee-9e8c-7175c85415e3_structure_or_id_webservice" was completed.

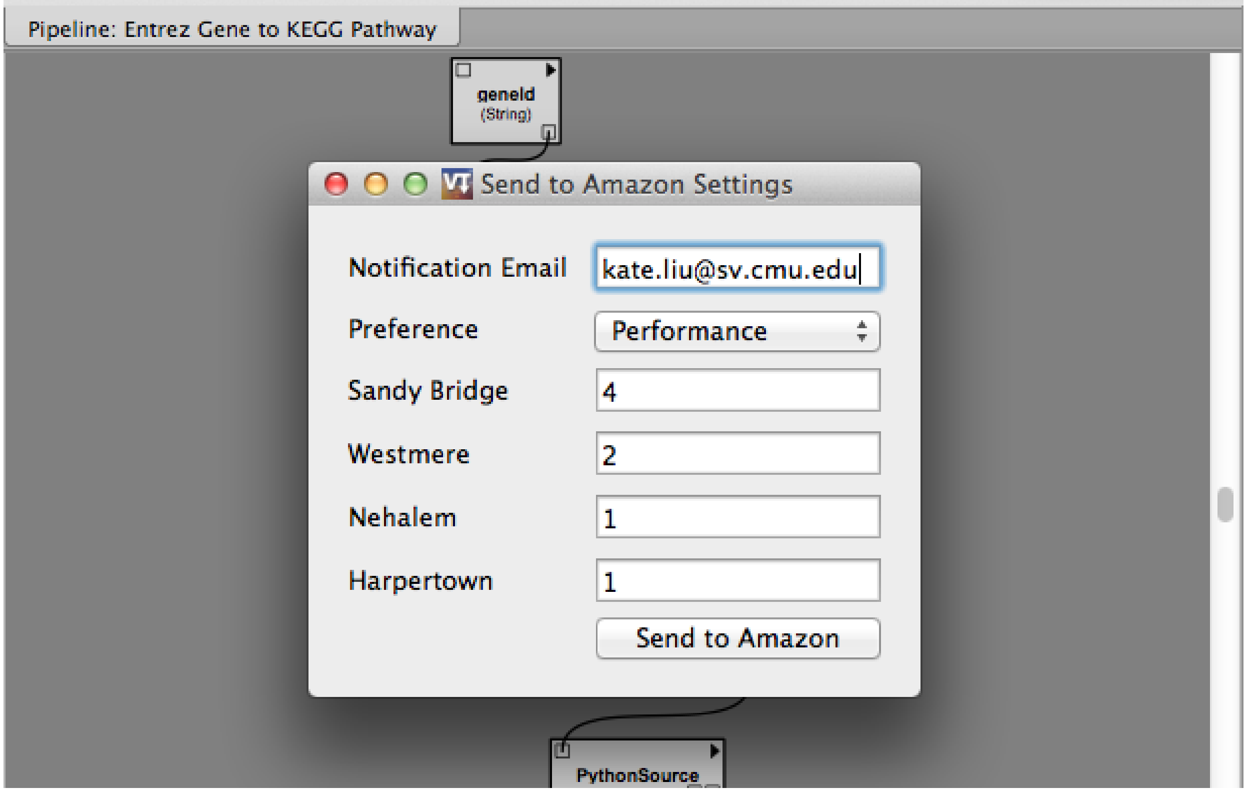
You can launch VisTrails or go the the following page to check the results:
http://ok.freya.cc/~owenchu/vistrails-output/owenchu_7c2a2888-168d-4cee-9e8c-7175c85415e3_structure_or_id_webservice_14.png

Click here to Reply or Forward

Amazon module menu items:



This window shows up when the user selects ‘Send to Amazon’. The functionality is the same as ‘Send to HECC’ as mentioned above, but instead of the job running on NASA’s HECC, the job is run on an Amazon EC2 instance. In this case, a RSA token is not required.



Send to Amazon Settings

Notification Email: kate.liu@sv.cmu.edu

Preference: Performance

Sandy Bridge: 4

Westmere: 2

Nehalem: 1

Harpertown: 1

References

[1] NASA High-End Computing Capability. Web. <http://www.nas.nasa.gov/hecc/>.

[2] VisTrails Wiki. Web. <http://www.vistrails.org/index.php/Main_Page>.

[3] NEX - NASA Earth Exchange. Web. <https://c3.nasa.gov/nex/>.

[4] Writing VisTrails Packages." Web.
<http://www.vistrails.org/usersguide/dev/html/packages.html>.