The HUBzero Platform: Extensions and Impressions

Anna Alber,[1,*] Jarek Nabrzyski,[1,2] and Timothy Wright[1,2]

[1]Center for Research Computing, University of Notre Dame

[2]Department of Computer Science & Engineering, University of Notre Dame

ABSTRACT

Motivation: We our efforts to work with HUBzero in a significant collaboration-oriented project, as well as our impressions of HUBzero after nearly a year of interaction with the platform. Our assessment is a mixed one: while the HUBzero platform does a good job of bringing users and modeling tools together, we found some noteworthy limitations of this recently open-sourced technology. For example: hubs are generally configured to deploy simulation tools of modest means with input and output data not readily shared across user accounts; by default, the security configuration of a HUBzero server is very open; hubs are inflexible in terms of their deployment requirements, and there is a need for more detailed documentation. In answer to the first (and most serious) of these issues, we have extended HUBzero to enable the use of more sophisticated modeling tools, such as openModeller Desktop, and to provide seamless access to external Network File System drive space.

1 INTRODUCTION

The open source HUBzero platform offers an effective way for scientific and educational communities to share information, interact, and run simulations (McLennan, 2010). The latter capability is considered to be HUBzero's "signature service" and, indeed, is not typically available through similar Web platforms.

We are currently employing HUBzero as part of an effort to build a so-called *collaboratory*: a virtual space in which users may share information and innovate. The goal of our work is to support a multidisciplinary community of scientists with a focus on adaptation strategies for biological systems. In addition to publishing and discussing content related to adaptation, users of the HUBzero portal (or "hub") need to leverage relevant simulation tools, such as openModeller and openModeller Desktop (de Souza Muñoz et al., 2011).

Such tools have requirements that may not be satisfied by a typical hub deployment. For example, there may be a need for software libraries that do not normally exist in HUBzero's virtualized Linux environment (OpenVZ [Parallels Holdings Ltd., 2011]). Also, for functionality purposes, users may have to manage and archive a tool's input and output files. As part of this management there will almost always be a need to make some files publically accessible (e.g., simulation results) and others private.

Tangentially related to these requirements is the need to restrict access to a hub's OpenVZ-based *Workspace* tool. Through Workspace a user is given a private, virtual Linux workstation that may be employed for nearly any purpose. Generally, this is intended to offer an environment for users to create and work with simulation tools that are made available in a hub. However, there is potential for accidental and malicious abuse of such a powerful virtual environment; for example, by gratuitous resource use or hacking.

To provide hub access to openModeller and accommodate the aforementioned needs, we have extended HUBzero in three ways. First, we have significantly expanded the OpenVZ environment to support the execution of a tool as complex as openModeller. Next, we have integrated NFS (Network File System) (Smith, 2006) with the file system made available through the HUBzero Workspace tool. Related to this, we have enhanced the process of hub user account creation to include the automatic setup of public and private NFS directories for specific users. Finally, we have made it possible to restrict which hub users are granted access to the Workspace and openModeller tools.

To permit long-running, computationally intense openModeller simulations, we have embarked on a separate project to kick off and manage such jobs outside of the HUBzero environment. The results of these simulations can be made available to our hub through NFS.

The remainder of this paper is organized as follows. Section 2 presents the state-of-the art in collaborative, virtual organization tools. Section 3 discusses the Collaboratory Project and simulations tools that we integrate with the HUBzero infrastructure. In section 4 we present the HUBzero architecture. Section 5 reviews the OpenVZ configuration and what we have done to enhance this configuration. Section 6 addresses our integration of NFS with HUBzero, including our use of public and private directories for each Workspace/openModeller user. Section 7 describes our changes to HUBzero to restrict access to the Workspace and openModeller tools. Section 8 briefly looks at a separate project to handle more intense openModeller jobs outside of our hub. Section 9 outlines our overall impressions of working with HUBzero. Finally, in Section 10 we offer some concluding remarks about this project.

[*]To whom correspondence should be addressed.

Copyright © 2011 for the individual papers by the papers' authors. Copying permitted only for private and academic purposes. This volume is published and copyrighted by its editors.

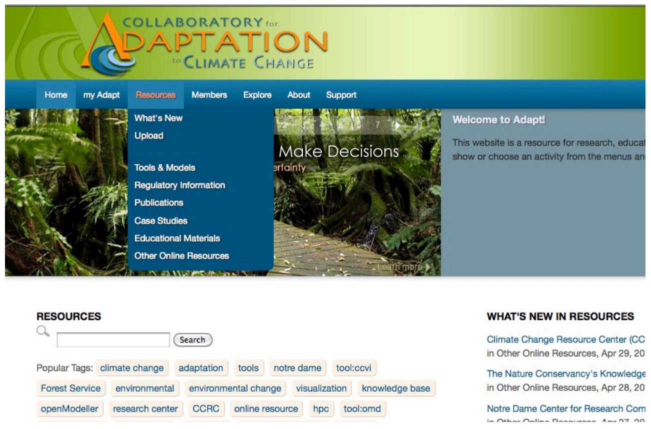
2 RELATED WORK

HUBzero offers many features that researchers need for effective collaboration. Major components include: various interactive simulation tools that can be accessed through almost any Web browser, a repository for online course materials and other publications, mechanisms for uploading new resources, a tool development area, ratings and citations abilities, content tagging mechanisms, wikis and blogs, private and public collaboration areas, usage statistics, news and events, and more. While relatively few platforms are as feature-rich, there are, nevertheless, some worth mentioning.

MyExperiment (De Roure et al., 2009) is a virtual research environment for collaboration and the sharing of experiments, which aims to provide a "workflow bazaar" for any workflow management system. An experiment is represented as an application workflow rather than as an infrastructure or services workflow. The design methodology of myExperiment was inspired by the Web 2.0 approach that was used in systems such as Facebook, MySpace, and Amazon. MyExperiment brings functionality to the user through familiar interfaces and can be combined with other services. Although its primary focus is on workflows, the designers of myExperiment realized that there was an immediate need to associate workflows with other information. Thus, the myExperiment concept is about sharing digital objects that include data, results, provenance information, tags, associated documentation, etc. These individual items can be collected together to form research objects, for example to record an experiment. Unlike Twine (www.twine.com), BioMedExperts (www.biomedexperts.com) or Nature Networking (network.nature.com), myExperiment is not intended to be a general social networking environment for scientists. Instead, the focus is on social networking around shared artifacts. In this way it is more like social bookmarking systems such as CiteULike (www.citeulike.org) and Connotea (www.connotea.org)—but with a much wider and richer remit than published articles, or social content systems like YouTube (www.youtube.com), SlideShare (www.slideshare.net), and Flickr (www.flickr.com). It effectively creates a social network of people and the items that they share. The main difference between myExperiment and HUBzero is that HUBzero is built using portal technologies and provides simulation tools that can be accessed from the browser and executed in the HUBzero space. Furthermore, users of HUBzero can add new resources and simulation tools within the HUBzero space. MyExperiment is built using Web 2.0 Ruby on Rails, rather than a portal framework, includes an API and offers remote execution capabilities.

HUBzero is often compared to the highly successful Open Courseware Initiative from MIT (web.mit.edu/ocw). However, HUBzero is more than just a repository for course materials. It is a place where researchers and educators can meet and accomplish real work. The HUBzero platform offers groups for private collaboration, software development projects in its forge area, event calendars, and many other services designed to connect researchers and build a community. But most importantly, it connects users to the simulation tools they need for research and education. Simulation jobs can be dispatched on national grid resources, including the NSF TeraGrid, the Open Science Grid, and others. HUBzero's middleware hides much of the complexity of distributed and grid computing, handling authentication, authorization, file transfer, and visualization, and letting the researcher focus on research.

Aside from its core features, HUBzero was chosen for our collaboratory project due to its wide adoption by the US research community as well as our easy access to support services that are provided by Purdue University. We believe that MyExperiment could be adapted to our needs, but its focus on workflows wasn't as good a fit as HUBzero. Our project heavily depends on climate and adaptation simulation tools that need to be run within the Collaboratory's system.



Welcome to Adapt!

This website is a resource for research, educat
show or choose an activity from the menus an

WHAT'S NEW IN RESOURCES

Climate Change Resource Center (CC
in Other Online Resources, Apr 29, 20

The Nature Conservancy's Knowledge
in Other Online Resources, Apr 28, 20

Notre Dame Center for Research Com

Fig. 1. The Collaboratory Project's hub home page.

3 OVERVIEW OF THE COLLABORATORY PROJECT AND SIMULATION TOOLS

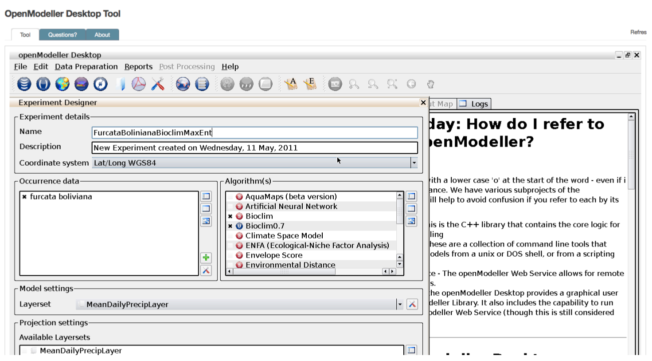
3.1 The Collaboratory Project

Climate change has compelled researchers to look not only for causality, but also for adaptation strategies. Aimed at building a virtual environment for comprehensive and informed decision-making, the Collaboratory Project is comprised of a team of researchers in ecology, computer science, law, and social science. We anticipate that our work will help foster an understanding of complexity in natural systems and charter a path from data to knowledge to insight and, finally, to action. Integrated with on-going and proposed survey research on expert opinion, we also plan to carry out social network analysis (collaboration networks, in particular) and study the impact of a virtual organization on the decision-making process. We anticipate that our efforts will result in various cyber tools that can be adapted by any domain requiring collaborative decision-making.

The role played by our hub (see Figure 1) is to operate as a shared space where researchers from numerous fields related to climate adaptation can run projection and simulation tools, devise new hub tools, and share results with colleagues and the public. Our initial work focused on making computational simulation and projection tools publicly available. However, we soon extended our hub architecture to better enable easy data sharing with added privacy control. As more tools become available, we plan on our

Copyright © 2011 for the individual papers by the papers' authors. Copying permitted only for private and academic purposes. This volume is published and copyrighted by its editors.

hub becoming a central point of contact for experts in climate change and adaptation strategies—a space where a suite of computational tools can be leveraged to simulate the effects of climate change on species and project future distribution patterns. As its community of users grows in both number and diversity, our hub will become more than a repository of publicly available tools and data: it will evolve into a virtual community for climate adaptation research where members contribute back by sharing new tools, publications, use cases, and data.


OpenModeller Desktop Tool
Tool
Questions?
About
Refres
openModeller Desktop
File Edit Data Preparation Reports Post Processing Help
Experiment Designer
ıt Map
Logs

Experiment details

Name: FurcataBolinianaBioclimMaxEnt

Description: New Experiment created on Wednesday, 11 May, 2011

Coordinate system: Lat/Long WGS84

Occurrence data

- ✗ furcata boliviana

Algorithm(s)

- AquaMaps (beta version)
- Artificial Neural Network
- ✗ Bioclim
- ✗ Bioclim0.7
- Climate Space Model
- ENFA (Ecological-Niche Factor Analysis)
- Envelope Score
- Environmental Distance

Model settings

Layerset: MeanDailyPrecipLayer

Projection settings

Available Layersets

- MeanDailyPrecipLayer

day: How do I refer to
enModeller?

ith a lower case 'o' at the start of the word - even if i
ance. We have various subprojects of the
ill help to avoid confusion if you refer to each by its

is is the C++ library that contains the core logic for
ling
hese are a collection of command line tools that
odels from a unix or DOS shell, or from a scripting

e - The openModeller Web Service allows for remote
s.
he openModeller Desktop provides a graphical user
deller Library. It also includes the capability to run
odeller Web Service (though this is still considered

Fig. 2. The openModeller experiment designer with multiple algorithms, available environmental layers, and biological data running on the Collaboratory Project's hub.

3.2 OpenModeller Desktop

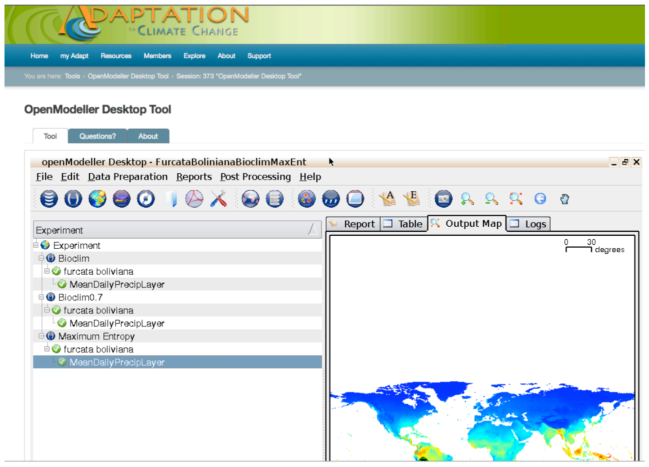
OpenModeller Desktop is an open source, fundamental niche modeling library with embedded Geographic Information System (GIS) using the Quantum GIS (QGIS) libraries. It provides a uniform method for modeling distribution patterns using a variety of widely used algorithms (de Souza Muñoz et al., 2011). The openModeller library is used to generate a projected occupancy map on the basis of environmental parameters considered in the model (e.g., temperature, precipitation, altitude) and biological data with point locations (see Figure 2). The openModeller package can run simulations with multiple species and multiple algorithms, and includes the ability to add newly developed models as plug-ins. This flexibility and versatility is central to the collaborative nature of our project, as it permits users to run a wide variety of projections, and develop/share new algorithms.

Projected occupancy maps (see Figure 3) as well as other output can be shared, but with an option to keep user-specified files private. Ecologists, biologists, policy makers, and experts in other fields can use scientific data and simulation results to make informed decisions about climate change and its effect on species population.

4 HUBZERO ARCHITECTURE

4.1 HUBzero Infrastructure

The HUBzero platform was developed at Purdue University to support nanoHUB.org, an online community for the Network for Computational Nanotechnology (McLennan, 2010). HUBzero is comprised of a Joomla! content management system, a ticketing mechanism, version control, a wiki system based on Trac Open Source Project, and middleware that enables the execution of hub tools inside a user's Web browser. To provide an easy-to-use portal system for the scientific community, HUBzero also incorporates features for publishing articles and presentations, a Q&A area, a wish list feature, and user group management. All hub tools run in restricted virtual containers that control access to the underlying file system, networking, and other processes. Each user is granted access to their own home directory with quota


ADAPTATION to CLIMATE CHANGE
Home
my Adapt
Resources
Members
Explore
About
Support
OpenModeller Desktop Tool
Tool
Questions?
About
openModeller Desktop - FurcataBolinianaBioclimMaxEnt
File Edit Data Preparation Reports Post Processing Help
Report
Table
Output Map
Logs
0 30 degrees

You are here: Tools › OpenModeller Desktop Tool › Session: 373 "OpenModeller Desktop Tool"

Experiment

- Experiment
  - Bioclim
    - furcata boliviana
      - MeanDailyPrecipLayer
  - Bioclim0.7
    - furcata boliviana
      - MeanDailyPrecipLayer
  - Maximum Entropy
    - furcata boliviana
      - MeanDailyPrecipLayer

Fig. 3. An openModeller simulation experiment with multiple models with output map inside user's browser

limitations and a Debian Linux X Window environment.

4.2 Site Walk-through

Our hub users can access various resources about climate adaptation strategy ranging from legal documents to simulation tools. Users can also become contributors by uploading articles, documents, and event information; creating new tools; and participating in forum discussions. As new tools are added to the hub, they contribute to the overall capacity for analysis and lead to more data; in turn, these data become input for steps in the adaptation decision-making process.

Adding a new tool to the hub begins with a registration process on the site, which alerts the hub administrator to create a code repository for the tool. The owner of the tool then iterates though a software engineering process until the tool is ready for publication to the hub's community. There is, of course, extensive documentation available on the hub to assist newcomers with all HUBzero features, including tool creation.

User groups may also be created to facilitate discussions and the exchange of ideas through the hub. This mechanism offers different levels of privacy, ranging from public to "by invitation only," and permits users to more finely manage their discussions and the sharing of data.

5 OPENVZ CONFIGURATION

5.1 Containers and VNC

HUBzero's middleware relies on OpenVZ to operate multiple, isolated containers (or virtual environments [VEs]). Because

Copyright © 2011 for the individual papers by the papers' authors. Copying permitted only for private and academic purposes. This volume is published and copyrighted by its editors.

OpenVZ utilizes Linux, any computational tools deployed within a hub must also be able to run under Linux. When a hub user launches a tool, the corresponding application is executed within a lightweight VE.

In addition to OpenVZ, HUBzero employs VNC (Virtual Network Computing [Richardson, Stafford-Fraser, Wood, & Hopper, 1998]) as a means of interacting with the interfaces of tools through a Web browser. Hub tool interfaces are displayed by connecting a TightVNC Java Applet to a RealVNC-enabled X Window server (Kisseberth, 2010). Both the applet and server are modified to work with the HUBzero system; the applet, of course, runs on a user's Web browser, while the server executes inside an OpenVZ VE.

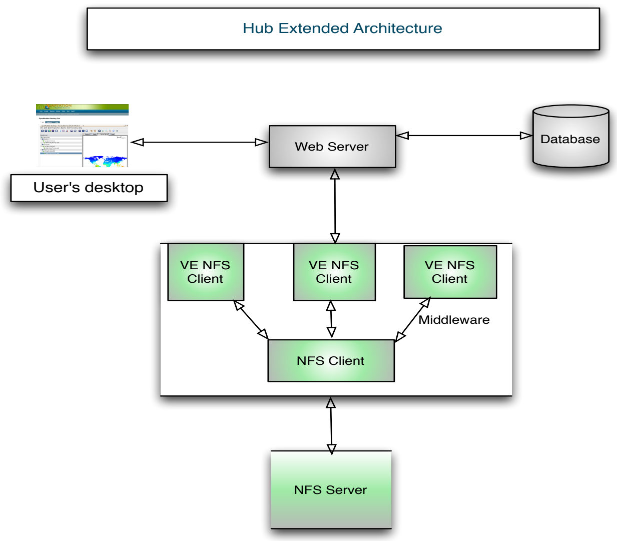


Fig. 4. The Collaboratory Project's HUBzero infrastructure.

5.2 Extending the OpenVZ Environment

The standard open source installation of HUBzero includes a basic OpenVZ configuration that we chose to enrich with additional software packages. For example, Subversion can be added to provide a more functional development environment for hub tools (although, in the most recent release of HUBzero, Subversion is already included). We found significant value in adding a graphical editor (*gedit*), Web browser (*kazehakase*), a compression package, and other essential utilities.

The installation of gedit was not limited to just package deployment. Additional changes to HUBzero's File Alteration Monitor (FAM) scripts are required before gedit can be installed without errors. FAM notifies applications when specific files or directories are changed (Silicon Graphics International, 2010).

In order to successfully execute a hub tool inside a VE, libraries on which the tool relies must also be added to the OpenVZ environment. For our hub, we needed to provide various C/C++ libraries (starting with the installation of the Debian *build-essential* package) and an extensive list of additional packages such as *qgis*, *gdal*, *sqllite*, *expat*, *grass* and *qt*. In addition, through a locally mounted directory accessible within our hub's VEs, we provided access to precompiled libraries for openModeller (in general, the libraries found inside our VEs differ from those of the underlying HUBzero host system).

We found it beneficial to test new tools with the hub Workspace tool. Through Workspace, error messages are displayed and provide hints about any missing libraries or problems with invocation scripts. For example, the binary package we manually created for openModeller contained several libraries that were not carried along internally during HUBzero's process of promoting the openModeller tool from initial registration to installation. In this case, we had to manually copy the missing libraries to their destination. This issue may have been the result of the HUBzero project not foreseeing the use of tools as extensive and complex as openModeller.

For other hub tools developed in Java, we configured a more complete Java environment for use by OpenVZ. In addition to the installation of the *sun-java6-jdk* package, we had to make changes to *x86_64-linux.gnu.conf* file for Java libraries.

5.3 X Window Configuration

Beyond providing access to all the necessary utilities and libraries for our openModeller hub tool, we also added an X Window window manager, *IceWM*, to properly display openModeller Desktop within the HUBzero TightVNC Java applet. With IceWM in place, openModeller Desktop appeared and behaved as it normally would on a graphical workstation. We employed the same window manager configuration used by the HUBzero Workspace tool, although we changed themes and preference files to better handle the openModeller icons and window behavior.

It is noteworthy that a special script is needed to invoke a hub tool; for openModeller certain requisites had to be in place within this script. In particular, we had to ensure that various openModeller libraries were pointed to and the IceWM window manager was started.

As a result of our efforts, users familiar with openModeller Desktop who visit our hub do not have to learn a new interface. We have also embarked on an effort to construct new openModeller plugins for our hub tool. These extensions will be made available for hub users as they become ready.

6 NFS CONFIGURATION

In a hub, files (as opposed to Web content) are typically accessible through the Workspace tool, which provides a Linux workstation within a VE. Our extensions to HUBzero employ NFS thin clients that operate within our hub's VEs and on our hub server.

6.1 Extended Storage

NFS allows sharing the resource of an external file system with many thin clients. In our case, these clients execute inside hub VEs and permit users to access an external file system as though it were local to the hub's server. As a result we can 1) use less disk space on our HUBzero server and provide access to external, scalable data storage; and 2) leverage the external NFS storage to permit public and private directories for hub users.

Initially, we made an effort to avoid running the NFS client on our server, but found that this made collecting and processing user data difficult and did not easily integrate with HUBzero's existing configurations. Also, it became cumbersome to automate the

Copyright © 2011 for the individual papers by the papers' authors. Copying permitted only for private and academic purposes. This volume is published and copyrighted by its editors.

creation of public and private user account directories under NFS (HUBzero's middleware already has access to the user information we need for the dynamic creation of public/private directories under NFS file system).

With our present architecture, an NFS client executes on both our hub server and inside the VEs. Under this configuration, our NFS space is mounted when the OpenVZ service starts. We made our NFS mount persistent for all VEs such that each VE becomes an NFS client that is accessible by the user via the Workspace tool. This permits us to access user data on the host machine while simultaneously providing access to the NFS space from inside the hub VEs. With this, NFS can be used for storing and sharing openModeller input and results.

6.2 Security Considerations

While data sharing is an important task, the ability to control data access is also important. To accommodate our researchers' need to manage how their data are shared, we automated our hub to create a public and a private directory for each user who is permitted to use the Workspace tool. When a given user launches the Workspace tool, HUBzero automatically checks to see if this is a new user, and, if they are new, creates their public and private directories in NFS space. If the user is later deleted, a maintenance routine is run to reclaim the defunct user's storage. A user's public NFS directory is accessible by all Workspace tool users as a read-only folder, while their private NFS directory is accessible only to them.

7 WORKSPACE AND TOOL RESTRICTIONS

By default, the HUBzero platform makes the Workspace tool available to any registered user. With such a configuration, our system's resources (including NFS) would be open to use and, possibly, abuse by anyone who registered with our hub. Given the CPU cycles and large data sets involved in simulations, the misuse of hub resources is a significant risk.

To better manage resource usage, we modified HUBzero so that new users are granted access to the Workspace tool only upon request and after being vetted. When an unauthorized (but registered) user tries to launch Workspace, they are given an option to request access through a trouble ticket. This request is dispatched to the hub administrator who may then determine whether or not to approve the desired access. If approved, the user will be able to employ Workspace with access to both their home directory and NFS public/private space.

Our extensions to restrict the Workspace tool required a number of changes to the tool's configuration. First, a new group was created to manage the use of Workspace. Next, we created a new resource through the HUBzero administrative interface, associating the aforementioned group name with this resource. Finally, we published the new resource and manually updated the hub database so that Workspace access became limited to the newly created group.[1] In a related enhancement, we also extended the HUBzero ticketing system logic to include a ticket status for Workspace access approval (as touched upon above).

Access control for tools similar to openModeller can be handled during the tool publishing process and restricted to a previously created group.

It is worth noting that testing HUBzero extensions can become somewhat cumbersome due to a requirement that each registered user have only one unique email address. For example, with our added restrictions to the Workspace tool, we needed to test multiple users who are members of various groups. Each user account, of course, requires a unique email address in order to be registered. However, this creates significant limitations for fabricating test accounts, since a legitimate email address must be devised for each account. Thus, we altered our hub so that this requirement, which we find useful in general, is relaxed for specific accounts that we set aside for testing purposes only.

8 HANDLING OPENMODELLER RESOURCE CHALLENGES

The expectations of our hub's user community are that simulation results are forthcoming with little delay. Even for small, simple jobs, however, openModeller's memory and CPU overhead is not insignificant. Moreover, the openModeller Desktop user interface can contribute a fair amount to this overhead, too. In general, such resource consumption can have a deleterious affect on any system if enough users simultaneously execute large openModeller jobs.

To better handle this type of workload, we have begun implementing a stand-alone system to execute multiple numbers of simultaneous openModeller jobs. Although job processing is handled outside of HUBzero's environment, job submission will take place through a webpage hosted on our hub, while simulation results will be accessible through the same NFS space that is made available to users of our hub's Workspace tool.

9 OVERALL IMPRESSIONS OF HUBZERO

After having been under development for a number of years, HUBzero was released to open source in 2010. This happened to coincide with the start of our collaboratory project and, for the reasons already noted, we decided to try HUBzero and deploy our own hub rather than pay for a hosting plan through Purdue University.

Because we have only recently made our project's hub available to the Internet, we have yet to accumulate enough traffic and benchmarking to offer a worthwhile opinion of HUBzero's capabilities while in production use and under load. Nevertheless, we can comment on deploying and securing HUBzero, as well as the support offered by the HUBzero project.

9.1 Deployment

We have deployed the open source HUBzero platform onto three different servers (one test and two production systems) multiple times. Each server operates Debian Lenny as a virtual machine (VM) running under the Kernel-based Virtual Machine (KVM) hypervisor (Solomon, 2011) with Red Hat Enterprise Linux 6 as the host. We found it helpful to deploy HUBzero as a VM since we could make snapshots of successful installations and return to these if things went awry. Also, operating a hub as a VM is a good way to manage the risk of incompatible hardware—for example, at

[1] This method was provided to us through communications with the HUBzero project team.

Copyright © 2011 for the individual papers by the papers' authors. Copying permitted only for private and academic purposes. This volume is published and copyrighted by its editors.

one point we attempted to install Debian Lenny on a new physical machine and ran into an unresolvable problem with the RAID adapter. With a VM acting as a layer of abstraction from underlying hardware, such issues generally cannot happen. To counteract the potential for loss of speed, especially with respect to I/O operations, our production host server is outfitted with 32 cores, 128 gigabytes of RAM, and 1.6 terabytes of hard disk space; all cores, 120 gigabytes of RAM, and 1.5 terabytes of disk are made available to the HUBzero VM.

The HUBzero installation process at the time of this writing consists of 77 steps spread over 25 major areas. This complexity often results in errors that are challenging to debug. One option the HUBzero project might consider implementing is a checkpoint system for their install process. With this, at key intervals it would be possible to know that a given installation is correct. To this end, we have compiled our own internal list of checkpoints, especially with regard to Workspace configuration. In a related matter, although we have not yet tested it, the HUBzero project has recently devised a beta version of a simplified installation process.

Over time our familiarity and expertise improved regarding HUBzero installations. Our team now maintains an internal wiki to help maintain institutional knowledge on this front and keep solutions to common problems close at hand. It now takes us roughly eight hours to setup a new HUBzero server (without adding our extensions).

9.2 Speed and Resource Consumption

An informal benchmarking of simultaneous openModeller Desktop tool executions offered some encouraging results. In our test (see Figure 5), each openModeller Desktop tool processed through a 149-megabyte data set, resulting in no perceptible delay for 18 simultaneous sessions. The maximum CPU utilization was 56% and the total running time was 76 seconds. As the number of sessions was decreased, we observed that the total maximum running time changed only slightly (e.g., 70 seconds for six simultaneous sessions); we believe this was due to the abundance of available resources for the tested number of sessions. For comparison, we found the maximum CPU utilization was 37% for 12 simultaneous sessions and 19% for six sessions. Memory usage was not significant given the amount of data processed (it remained below 6% of our hub's total memory throughout the test). On a previous HUBzero VM (four CPU cores, four gigabytes of RAM, and 500 gigabytes of hard disk) we had carried out a similar test of the openModeller Desktop tool, but with fewer simultaneous sessions. There, for 15 simultaneous sessions we noted the memory usage peaked at 31%, while CPU utilization reached 94%; running time was considerably greater at about three minutes.

9.3 Security

The "out of the box" security posture of HUBzero is best described as open. To begin with, a newly installed hub will start life with several critical services exposed to its network. These include: SSH, SMTP, MySQL, and LDAP. Each of these should be appropriately restricted—in the case of SMTP, MySQL, and LDAP, these can be safely restricted to the hub itself. Next, other than "root," it is not possible to install a user account on a hub. This makes it very difficult to know who is using the root account (since anyone logging into a hub server must do so as root) and significantly reduces the utility of OS activity audit trails. Related to this, the root account is exposed to password guessing through SSH login attempts. Finally, efforts to install file space monitoring tools such as Tripwire appear to cause malfunctions in HUBzero. (Although, we cannot offer much detail about Tripwire's erroneous interactions with our hub deployments, as we elected not to spend time diagnosing this issue.)

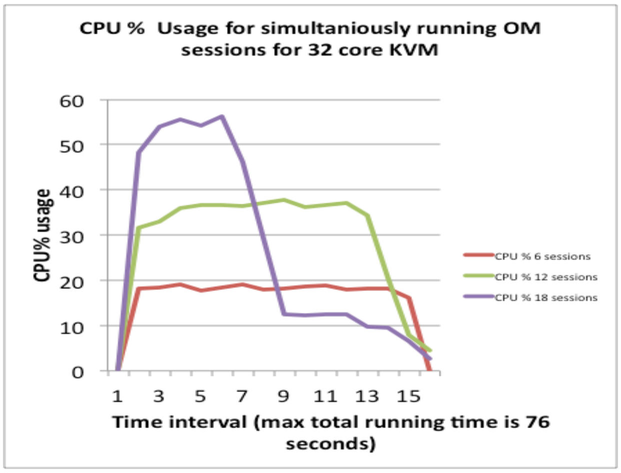


Fig. 5. CPU utilization for 6, 12, and 18 simultaneously executing openModeller jobs.

9.4 Support

The support available through the HUBzero project has improved significantly over time. We typically see responses to our email queries within 24 hours and, in some cases, even on weekends. Often, the issues we raise are already known to the HUBzero project team and do not require much troubleshooting on their part. Both a knowledge base and a "Questions & Answers" resource are available through the HUBzero project website. Although anyone can post a question, only a handful of institutions appear to be deploying their own hubs (as opposed to purchasing a hosting plan through Purdue), so there is relatively little information to be found in existing support resources.

10 CONCLUSION

The abilities to execute simulation/modeling tools and share information are paramount to our collaboratory project. As such, we found the HUBzero platform to be a proper fit for sharing computational tools and data. While the HUBzero platform does a good job of bringing users and modeling tools together, we extended its functionality to enable the use of sophisticated, computationally intensive modeling tools. Also, we discovered that a default hub requires additional security measures to protect its network services (e.g., SMTP, MySQL, and LDAP) and restrict access to its OpenVZ environment. In terms of resource consumption, we tested relatively modest sets of data with the openModeller Desktop tool and will continue evaluating resource consumption for larger data sets as they become available. Finally, we provided seamless access to external Network File System drive space to store and share input and output data.

Copyright © 2011 for the individual papers by the papers' authors. Copying permitted only for private and academic purposes. This volume is published and copyrighted by its editors.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1029584.

REFERENCES

De Roure, D., Goble, C., & Stevens, R. (2009). The design and realisation of the virtual research environment for social sharing of workflows. *Future Generation Computer Systems, 25*(5), 561-567.

de Souza Muñoz, M., De Giovanni, R., de Siqueira, M., Sutton, T., Brewer, P., Pereira, R., et al. (2011). openModeller: A generic approach to species' potential distribution modelling. *GeoInformatica, 15*(1), 111-135. doi:10.1007/s10707-009-0090-7

Kisseberth, Nicholas J. (2010), "Hub set-up - Setting up Middleware," http://hubzero.org/resources/186

McLennan, M., & Kennell, R. (2010). HUBzero: A platform for dissemination and collaboration in computational science and engineering. *Computing in Science Engineering, 12*(2), 48-53.

Parallels Holdings Ltd. (2011). *OpenVZ project website.* Retrieved 3/19, 2011, from http://wiki.openvz.org/Main_Page

Richardson, T., Stafford-Fraser, Q., Wood, K. R., & Hopper, A. (1998). Virtual network computing. *Internet Computing, IEEE,* 2(1), 33-38.

Silicon Graphics International. (2010). *File alteration monitor*. Retrieved 4/5, 2011, from http://oss.sgi.com/projects/fam/

Smith, M. C. (2006). *Linux NFS project website*. Retrieved 3/19, 2011, from http://nfs.sourceforge.net/

Solomon, H. (2011). *KVM - the linux kernel-based virtual machine*. Retrieved 3/22, 2011, from http://www.linux-kvm.com/

Copyright © 2011 for the individual papers by the papers' authors. Copying permitted only for private and academic purposes. This volume is published and copyrighted by its editors.