

SIG Proceedings Paper in LaTeX Format*

Extended Abstract[†]

Ben Trovato[‡]
Institute for Clarity in
Documentation
P.O. Box 1212
Dublin, Ohio 43017-6221
trovato@corporation.com

G.K.M. Tobin[§]
Institute for Clarity in
Documentation
P.O. Box 1212
Dublin, Ohio 43017-6221
webmaster@marysville-ohio.com

Lars Thørväld[¶]
The Thørväld Group
1 Thørväld Circle
Hekla, Iceland
larst@affiliation.org

Lawrence P. Leipuner
Brookhaven Laboratories
P.O. Box 5000
lleipuner@researchlabs.org

Sean Fogarty
NASA Ames Research Center
Moffett Field, California 94035
fogartys@amesres.org

Charles Palmer
Palmer Research Laboratories
8600 Datapoint Drive
San Antonio, Texas 78229
cpalmer@prl.com

John Smith
The Thørväld Group
jsmith@affiliation.org

Julius P. Kumquat
The Kumquat Consortium
jpkumquat@consortium.net

ABSTRACT

This paper provides a sample of a L^AT_EX document which conforms, somewhat loosely, to the formatting guidelines for ACM SIG Proceedings¹.

CCS CONCEPTS

•**Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; •**Networks** → Network reliability;

KEYWORDS

ACM proceedings, L^AT_EX, text tagging

ACM Reference format:

Ben Trovato, G.K.M. Tobin, Lars Thørväld, Lawrence P. Leipuner, Sean Fogarty, Charles Palmer, John Smith, and Julius P. Kumquat. 1997. SIG Proceedings Paper in LaTeX Format. In *Proceedings of ACM Woodstock conference, El Paso, Texas USA, July 1997 (WOODSTOCK'97)*, 4 pages.
DOI: 10.475/123.4

*Produces the permission block, and copyright information

[†]The full version of the author's guide is available as `acmart.pdf` document

[‡]Dr. Trovato insisted his name be first.

[§]The secretary disavows any knowledge of this author's actions.

[¶]This author is the one who did all the really hard work.

¹This is an abstract footnote

Permission to make digital or hard copies of part or all 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. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

WOODSTOCK'97, El Paso, Texas USA

© 2016 Copyright held by the owner/author(s). 123-4567-24-567/08/06...\$15.00

DOI: 10.475/123.4

1 INTRODUCTION

1.1 Background

Cloud computing has marked significant developments and possibilities in the industry. It focuses on offering services for the different needs of the modern society. There are three categories of cloud computing services namely, Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). Organizations provide SaaS depending on the demand. Google Apps is one example of SaaS that can be used to manage email and create documents, etc. PaaS offers developers a platform where they can build and deploy applications. IaaS provides storage, servers and handling computer clusters. These tools are primarily created to serve computational needs [?]. A cloud computing platform dynamically allocates, configures, reconfigures and deallocates servers as requested or on demand. This approach ensures the elasticity of cloud computing [?].

Most scientific applications require high performance computing (HPC) which needs CPU intensive computations and large data storage. To be able to host such applications, several computers interconnected in a network such as clusters are needed. This makes scientific computing very costly. Fortunately, with the advancement in cloud computing, these tools can be deployed in the cloud without worrying about the costs of hardware purchase and maintenance [?]. To determine the cost of hosting scientific computing in cloud, the Scientific Computing Cloud (SciCloud) project is conducted in the University of Tartu. Results have shown that transmission delays are the major concern in pursuing HPC applications in the cloud[?]. To address this problem, a study is conducted by Hermocilla[?] in Peak-Two Cloud (P2C). One of the features introduced by P2C is vCluster. vCluster is a tool that enables a user to deploy a working (MPI) cluster on demand and to terminate it after execution. It uses a

master-slave design implementation. However, vCluster is a command-line based application which has limited capacities for data manipulation, analysis, and presentation. Creating a web application that would host vCluster with additional features like graphical representation to visualize trends, and a Graphical User Interface(GUI) to offer convenient access. This paper will present SkyLab, a web application that aims to serve and to improve vCluster functionalities.

1.2 Significance of the Study

This study will help us visualize how HPC applications can be hosted in the cloud and understand why is this technology timely relevant. Even with capable hardware, computations might take hours or even days which makes it a hassle for users. Thus, there is a need for a more convenient way of access for HPC applications. Furthermore, this would encourage students to explore and research on HPC applications. HPC applications usually have a plain numerical output. A means of presenting data graphically would make the interpretation of data more efficient.

The study would contribute to research on hosting HPC applications in the cloud. Problems encountered and solutions offered throughout the study will give insights to future researchers. Furthermore, the output application of the study can be used by the academe.

1.3 Scope and Limitations

The study focuses on vCluster and is not mainly concerned on P2C as a whole. The study initially bases on the current implementation of vCluster and eventually improve it and offer additional features[?].

1.4 Objectives

This study aims to develop SkyLab, a web application that would function as a front-end for vCluster. Specifically, it should be able to:

- (1) allow users to select tools and execute them via web interface;
- (2) create an extensible platform that would accommodate additional tools; and
- (3) display output data of tools.

2 REVIEW OF RELATED LITERATURE

2.1 Performance Concerns for HPC Applications in the Cloud

Shajulin [?] has enumerated five performance concerns for HPC applications in the cloud. First concern is resource provisioning issues which includes memory management,

energy management, response time, and resource outages. Hardware virtualization for multiple guest operating systems running concurrently on a host computer creates a concern for huge overheads, delays, or memory leakages caused by the process of allocating memory, remapping it to virtual

machines, and cleaning it up. The scenario of multiple guest OS can also cause an OS to fail to make hardware use energy-efficient because the presence of multiple hardware virtualizations prevent the main OS from having full control over actual hardware. Configuration failures, update failures, or even unknown issues cause resource outages. Second, most programming models used in the cloud have inconsistencies on error recovery and storage handling by design and by execution. If these inconsistencies are not addressed, the system components might fail because they are not well-integrated. Third, enforcing terms of Service Level Agreement (SLA) is bound to performance issues due to loss of control over resource provisioning or deploying cloud services. Fourth, there is a lack of standard conventions in clouds. This means that there is no guarantee that the current configuration of the an application will be compatible with other providers. The user might need to modify the program for a specific provider. Fifth, security measures challenge performance since increasing security protocols decreases response time.

In a study by Jackson et al.[?] conducted on Amazon EC2, benchmarks are used to imitate the workload of a typical HPC application. Results of the study show that the percentage of the communication time of the application is strongly correlated to its overall performance. Communication time and frequency of global communications are inversely proportional to the application's performance. Lastly, performance variability is introduced by the added layers of abstraction caused by hardware virtualization.

Another study conducted by Juve et al.[?], analyzed running scientific workflows in Amazon EC2. It has been found that getting resources is the primary cost factor to consider in executing workflow tasks while storage has relatively cheaper costs. Storing data in the cloud rather than transferring it to each workflow effectively reduces the high cost of data transfer. These results show that the cloud is a good alternative for running scientific workflow applications but for cloud providers to be able compete with existing physical clusters in terms of performance of HPC applications would need high-speed networks and parallel file systems[?]. While this may mean that cloud computing is yet to support large-scale HPC applications, it can still be considered as an alternative for deploying simple HPC tasks on demand[?]. Despite the performance trade-off, EC2 offers ease-of-use and cheaper costs which are marginal factors in consideration for academic purposes[?].

2.2 Related Work

Ganglia is a system designed to monitor high performance computing systems. It uses a hierarchical model in managing the system of clusters. It uses optimized data structures and communication algorithms to achieve scalability with high concurrency. It is claimed to be used by over 500 clusters around the world. This implies that the system is tested and trusted to be used for real-world applications[?].

One of the main inspirations for developing SkyLab is the Yabi system. It provides a web interface with support

for workflow environments with focus on introducing HPC applications to non-technical audience. Users can create and reuse workflows, and manage large amounts of data while system administrators can configure tools via the web interface as well. It is currently in use by multiple institutions, and is maintained as an open-source project[?].

Another related project is Web Interface for mpiBLAST (WimpiBLAST). It supports mpiBLAST, a parallel implementation of Basic Local Alignment Search Tool (BLAST). BLAST is a software used for sequence homology similarity search in large databases of gene sequences. mpiBLAST can utilize HPC clusters to achieve faster computing speeds but it requires knowledge in using MPI commands to benefit from its advantages. WimpiBLAST addresses this problem by providing the user a web interface to simplify the steps to use mpiBLAST[?].

3 METHODOLOGY

3.1 Materials

Programming Language: Python

Web Framework: Django

DBMS: MariaDB

Cloud Infrastructure: Peak-Two Cloud

3.2 System Architecture

SkyLab functions as a web front-end for vCluster which is built on top of Peak-Two Cloud.

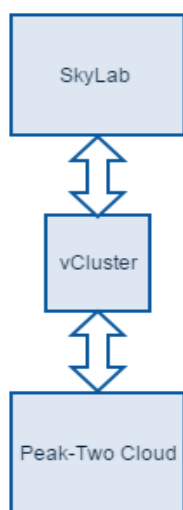


Figure 1: System Architecture of SkyLab

3.3 Requirements Specification

3.3.1 Functional Requirements.

- Users can browse and select tools with an interface. It includes, but not limited to:

AutoDock

It is a software used to simulate protein-ligand docking[?].

AutoDock Vina

It is a software similar to AutoDock 4 but on the average, it provides faster and more accurate computations[?].

DOCK

It is used to predict the small molecule-target interactions[?].

Quantum ESPRESSO

It is an integrated software suite of tools for ab-initio molecular dynamics (MD) simulations and electronic structure calculations[?].

GAMESS

It is used for ab initio molecular quantum chemistry. [?]

Ray

It uses parallel genome assemblies for parallel DNA sequencing [?].

- Users can upload input data to the server.
- Users can view the results of the used tools.
- The system is integrated with vCluster functionalities.

3.3.2 Non-functional Requirements.

- Users must be authenticated to be able to use the system.
- The system must be easy to use and user friendly.
- The system must be highly maintainable for future development.

3.4 Evaluation of Methodology

The use cases given ensure that the user will be able to select tools and execute them with the system. Also, the output files of user tasks will be served by the system. The system code is required maintainable for future inclusion of other tools. The methodology provided is sufficient to achieve the given objectives of the study. The created software will be subjected to user acceptance test for further evaluation.

4 RESULTS AND DISCUSSION

4.1 System Design

4.1.1 MPI clusters. The system spawns a thread (MPIThread) for each active cluster which handles the connection to the assigned cluster via Secure Shell (SSH). Creation and deletion of clusters is done by using vCluster commands while tool activation is done by using p2c-tools. The thread also manages task queuing and execution.

A cluster is either classified as public or private. If it is set to public, every user in the system can use it. On the other hand, for private clusters, the cluster will only be visible to the owner. The owner has the option to share the cluster to other users via the share key generated for the said cluster.

4.1.2 Tool sets. The system searches for Python packages inside the assigned modules folder and install it on server

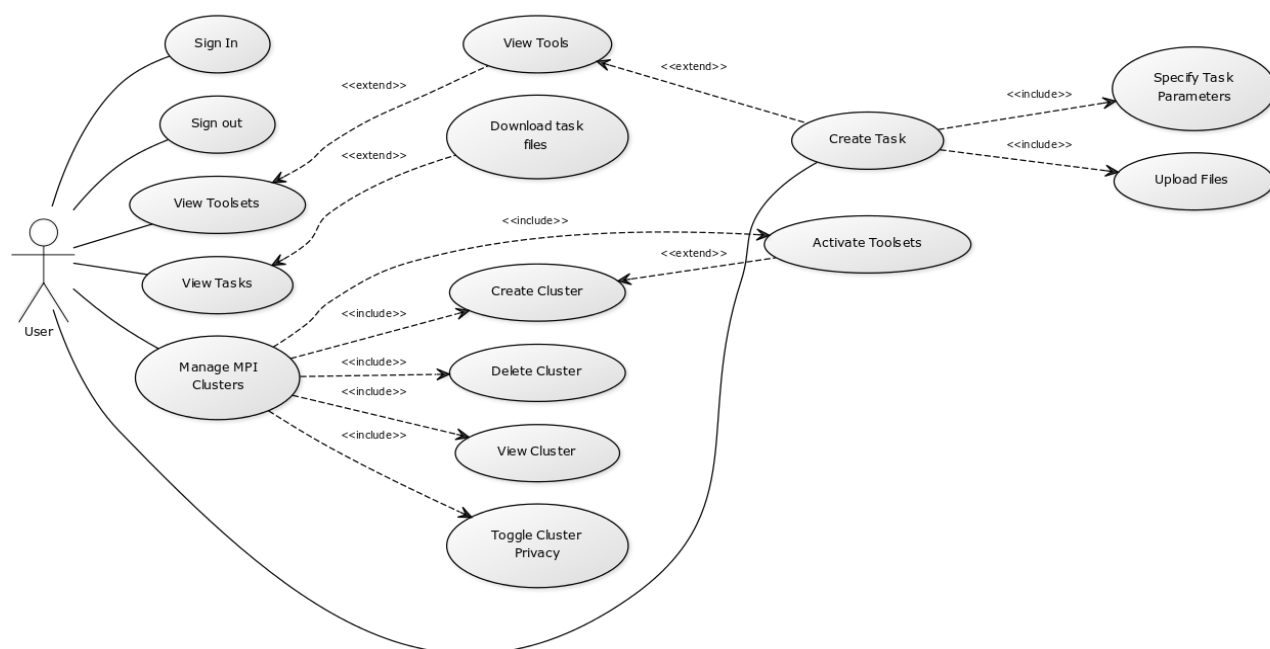


Figure 2: Use Case Diagram of SkyLab

start. The tool sets will then be available for use with the system. The package must have a Python module named *install.py* which contains function calls for integrating the package with the system. The package must also contain the corresponding views and executable classes for each sub-tool.

4.1.3 Tasks. The system creates a task object for each task input by the user. A signal will then be sent and it is then received by the corresponding MPIThread which queues the task for execution. When a task is executed, it calls the assigned executable class with the given parameters. On connection error, the task waits exponentially before retrying. If the server crashes while running task execution, the task is just restarted.

Default task execution flow via executable class:

- (1) Needed remote and local directories for execution are cleared or created.
- (2) Input files are uploaded to cluster.
- (3) List of commands given are executed.
- (4) Output files are sent back to the server.
- (5) Remote task folder is deleted.
- (6) Output files are served by the server.

4.2 System Features

The system's interface offers different functionalities that simplifies MPI cluster and task management.

- The user is authenticated by logging in with his @up.edu.ph Google account.
- The user can create an MPI cluster with optional tool activations.

Create MPI Cluster

Cluster name*

This is required to be unique. e.g. chem_205_games_12_12345

Cluster size*

Toolsets

☐ AutoDock 4

☐ AutoDock Vina

☐ Dock 6

☐ GAMESS

☐ Impi

☐ Quantum ESPRESSO

☐ Ray

Select toolsets to be activated. Optional

☐ Public

This option makes the cluster visible to all users.

Execute

Figure 3: MPI creation form

- The user can monitor visible public and private MPI clusters.

MPI Clusters

Show 10 entries

Search:

Cluster name	Nodes	IP address	Tasks queued	Status	Visibility	Date created
testcluster1	2	10.0.3.243	1	Connecting	Private	10/25/16 12:44 AM

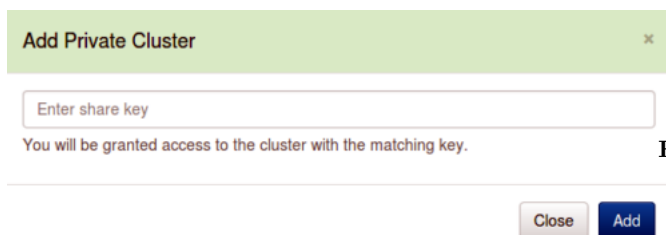
Showing 1 to 1 of 1 entries

Create MPI cluster Add private cluster

Previous 1 Next

Figure 4: MPI cluster table

- The user can make a private cluster visible by entering a valid share key.



Add Private Cluster

Enter share key

You will be granted access to the cluster with the matching key.

Close Add

Figure 5: Add private cluster form

- The user can view details about a MPI cluster. If the user is the cluster's owner he has the option to delete the cluster.

testcluster1

Cluster name	Nodes	IP address	Tasks queued	Status	Share key	Visibility	Date created
testcluster1	2	10.0.3.243	1	Connecting	SC010	Public OFF	10/25/2016 12:44 a.m.

Delete this cluster

Toolsets

Toolset	Description	Status
AutoDock 4	AutoDock is a suite of automated docking tools. It is designed to predict how small molecules, such as substrates or drug candidates, bind to a receptor of known 3D structure.	Activated
Dock 6	The new features of DOCK 6 include: additional scoring options during minimization; DOCK 3.5 scoring-including Delphi electrostatics, ligand conformational entropy corrections, ligand desolvation, etc...	Activated
GAMESS	The General Atomic and Molecular Electronic Structure System (GAMESS) is a general ab initio quantum chemistry package. Briefly, GAMESS can compute SCF wavefunctions ranging from RHF, ROHF, UHF, GVB, ...	Activated
Ray	Ray is a parallel software that computes de novo genome assemblies with next-generation sequencing data.	Activated
AutoDock Vina	AutoDock Vina is an open-source program for doing molecular docking. It was designed and implemented by Dr. Oleg Trott in the Molecular Graphics Lab at The Scripps Research Institute.	Activated
Quantum ESPRESSO	Quantum Espresso is an integrated suite of Open-Source computer codes for electronic-structure calculations and materials modeling at the nanoscale. It is based on density-functional theory, plane wave...	Activated
Impi	Image processing tool that runs in parallel via MPI	Activated

Figure 6: MPI detail view

- The user can select from a list which tool does he want to use.

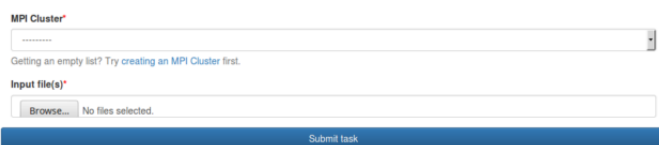
Toolsets

AutoDock 4 AutoDock is a suite of automated docking tools. It is designed to predict how small molecules, such as substrates or drug candidates, bind to a receptor of known 3D structure.
AutoDock Vina AutoDock Vina is an open-source program for doing molecular docking. It was designed and implemented by Dr. Oleg Trott in the Molecular Graphics Lab at The Scripps Research Institute.
Dock 6 The new features of DOCK 6 include: additional scoring options during minimization; DOCK 3.5 scoring-including Delphi electrostatics, ligand conformational entropy corrections, ligand desolvation, etc...
GAMESS The General Atomic and Molecular Electronic Structure System (GAMESS) is a general ab initio quantum chemistry package. Briefly, GAMESS can compute SCF wavefunctions ranging from RHF, ROHF, UHF, GVB, ...
Impi Image processing tool that runs in parallel via MPI
Quantum ESPRESSO Quantum Espresso is an integrated suite of Open-Source computer codes for electronic-structure calculations and materials modeling at the nanoscale. It is based on density-functional theory, plane wave...
Ray Ray is a parallel software that computes de novo genome assemblies with next-generation sequencing data.

Figure 7: Tool set list view

- The user can submit a task by filling up a tool's task creation form.

GAMESS



MPI Cluster*

Getting an empty list? Try creating an MPI Cluster first.

Input file(s)*

Browse... No files selected.

Submit task

Figure 8: GAMESS task creation form

- The user can monitor created tasks.

Tasks

Show 10 entries

Task number	Tool	MPI Cluster	Task status	Date created	Last updated
38	Impi (Impi)	testcluster1	Success	11/01/2016 2:56 a.m.	11/01/2016 4:01 a.m.
37	Impi (Impi)	testcluster1	Success	10/31/2016 3:04 p.m.	10/31/2016 3:13 p.m.
34	GAMESS (GAMESS)	testcluster1	Success	10/29/2016 6:27 p.m.	10/29/2016 7:12 p.m.
33	GAMESS (GAMESS)	testcluster1	Success	10/29/2016 6:08 p.m.	10/29/2016 6:20 p.m.
32	GAMESS (GAMESS)	testcluster1	Success	10/29/2016 6:04 p.m.	10/29/2016 6:08 p.m.
31	GAMESS (GAMESS)	testcluster1	Success	10/28/2016 2:24 a.m.	10/28/2016 2:24 a.m.
30	GAMESS (GAMESS)	testcluster1	Success	10/25/2016 11:45 p.m.	10/28/2016 1:46 a.m.
29	GAMESS (GAMESS)	testcluster1	Success	10/25/2016 6:30 a.m.	10/25/2016 6:34 a.m.

Showing 1 to 8 of 8 entries

Create new task

Figure 9: Task table view

- The user can view results of tasks. JSmol renders the compatible output files[?].

Task 29

GAMESS @ testcluster1 Success

Input Files:

glyc_makelp.inp

Output Files:

HQO.log
glyc_makelp.log
task_29_scratch_files.zip

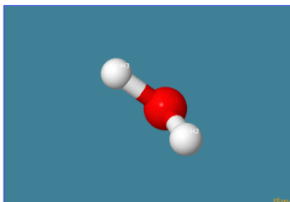


Figure 10: Task detail view

4.3 User Acceptance

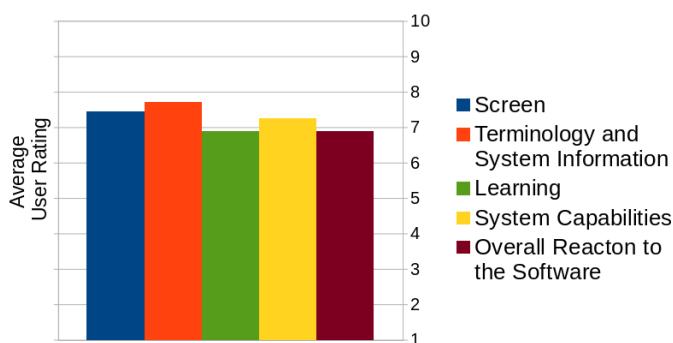


Figure 11: Results of QUIS for SkyLab

The system has been evaluated by 56 respondents by answering a survey based on Questionnaire for User Interface Satisfaction (QUIS) [?]. Respondents are students who are unfamiliar with both HPC tools and the concept of MPI systems. Respondents are asked to test features of SkyLab by following a set of instructions and using input files provided. On the average, the users rated their overall experience with SkyLab to 6.9/10. The users listed the simplicity of the user interface to be the most positive aspect of the system while the slow speed of task processing is said to be the most negative. Majority of the tools supported by SkyLab have inherently long processing time which is not known to the respondents. The system does not focus on optimizing the said tools to achieve better performance but rather it focuses on simplifying the user's task submission process.

5 CONCLUSION AND FUTURE WORK

The system created allows users to manage MPI clusters and submit tasks without the need for technical expertise in scripting. This makes the advantages of HPC available to non-technical users. This is achieved by parsing form inputs to generate commands for task execution. Task files can be download from the server and output files are displayed with the help of JSmol[?]. The system is also configured to install tool sets found in the modules folder making it possible to accommodate additional tools. Based on the user acceptance test conducted, the users found the system to be acceptable in terms of the criteria provided, in general.

The system achieved its main objectives but its features can still be improved and additional features can be introduced. Improved input parameter checking and error handling will make the system more robust. There are still use cases of tools that are yet to be supported. Input file generation can make the process more interactive and more customizable. Workflow design support will enable users to run complex tasks. Support for custom MPI programs will make it easier for developers to utilize the system as a test environment. Task scheduling and resource management algorithms can be used to efficiently handle resource-intensive or time consuming tasks. For example, a cluster can borrow resources from idle clusters. These recommendations will provide the users a better experience in using the system for academic and research purposes.