Implementation and Usability Evaluation of a Cloud Platform for Scientific Computing as a Service (SCaaS)

Prasad Saripalli
Runaware Inc.
Coral Springs, FL.
prasadsa@runaware.com

Curt Oldenburg
Lawrence Berkeley National Lab
Berkeley, CA.
CMOldenburg@lbl.gov

Ben Walters; Radheshyam N.
Runaware Inc
Coral Springs, FL.
ben@runaware.com

ABSTRACT

Scientific computing requires simulation and visualization involving large data sets among collaborating teams. Cloud platforms offer a promising solution via ScaaS. We report on the architecture, implementation and User Experience (UX) evaluation of one such SCaaS platform implementing TOUGH2V2.0, a numerical simulator for sub-surface fluid and heat flow, offered as a service. Results from example simulations, with virtualization of workloads in a multi-tenant, Virtual Machine (VM)-based cloud platform, are presented. These include fluid production from a geothermal reservoir, diffusive and advective spreading of contaminants, radial flow from a CO2 injection well and gas diffusion of a chemical through porous media. Prepackaged VM pools deployed autonomically ensure that sessions are provisioned elastically on demand. Users can access data-intensive visualizations via a web-browser. Authentication, user state and sessions are managed securely via an Access Gateway, to autonomically redirect and manage the workflows when multiple concurrent users are accessing their own sessions. Usability in the cloud and the traditional desktop are comparatively assessed, using several UX metrics. Simulated network conditions of different quality were imposed using a WAN emulator. Usability was found to be good for all the simulations under even moderately degraded network quality, as long as latency was not well above 100 ms. Hosting of a complex scientific computing application on an actual, global Enterprise cloud platform (as opposed to earlier remoting platforms) and its usability assessment, both presented for the first time, are the essential contributions of this work.

Keywords

Scientific Computing; Software as a Service; Cloud computing; Usability

1. INTRODUCTION

Scientific Computing (SC) of many natural and man-made phenomena to validate the underlying models, and understand or discover the underlying physics requires numerical simulation of several physico-chemical and biological processes, and visualization of results at unprecedented levels of detail, involving terabytes of data and costly computations [1]. Real-time simulations at hierarchical scales, and simulation of a large number of scenarios at varying scales, which are often required to properly model dynamic phenomena, render this process especially challenging and costly. The sheer volume of data requiring massive amounts of storage, memory and processing power coupled with the stringent usability requirements necessary to optimally facilitate the cognitive insights during knowledge discovery contribute to this challenge [2]. Further, large-scale scientific computing projects are often collaborative in nature, and require tools for managing the metadata across multiple user groups, including pedigree and provenance tracking [3]. For such reasons, SC tools and expertise remain inaccessible to many practitioners who could benefit from their use, in diverse applications such as global climate change, water, energy, and health management, to cite a few examples.

Dedicated High Performance computing (HPC) infrastructures such as clusters and pools of networked machines, managed by cycle-stealing middleware such as Condor [4], are an early example of collaborative computing in support of SC. Grid computing may be considered a predecessor to Cloud computing [5] which enabled access to the computational power required to perform large experiments via a network of machines. Vecchiola et al [6] provided a comprehensive overview of HPC applications for SC, and presented Aneka, a Cloud platform for developing distributed applications on the Cloud, and demonstrated its use for the classification of gene expression data and the execution of fMRI brain imaging workflow. Grid computing was shown to be helpful to running large scale, collaborative SC projects, which include the Open Science Grid [7], Grid for ESciencE (EGEE) [8] and TeraGRID [9]. For example, TeraGrid is used by 4000 users at over 200 universities that advance research in molecular bioscience, ocean science, earth science, mathematics, neuroscience, design and manufacturing, and other disciplines. Grid computing enabled SC workflows with new capabilities such as dynamic discovery of services, the ability of relying on a larger number of resources belonging to different administrative domains and of finding the best set of machines meeting the requirements of applications.

Differences between Grid and Cloud computing are explained in detail in [5] emphasizing the fact that Grid has been focused on SC applications whereas the Cloud started as an Enterprise focused effort. In the recent past, researchers have begun to apply cloud computing for SC in projects such as Nimbus [10], Stratus [11], Cumulus [12], OpenNEbula [13] and the RESERVOIR project [14]. Rehr et al. [15] reported on the use of cloud computing for running serial and parallelized versions of the widely used x-ray spectroscopy and electronic structure code FEFF on the Amazon Elastic Compute Cloud. Very recently, SGI announced Cyclone [16], the world's first large-scale, on-demand cloud computing service specifically dedicated to technical applications. Cyclone is immediately available and enables two service models: Software as a Service (SaaS) and Infrastructure as a Service (IaaS). The U.S. Office of Science has announced Magellan, a multi-organizational collaborator test bed to explore

978-0-7695-4592-9/11 $26.00 © 2011 IEEE
DOI 10.1109/UCC.2011.58

the effectiveness of scientific cloud computing. Such efforts underscore the importance and timeliness of SCaaS.

Cloud platforms offer a promising solution to address the needs of modern scientific computing initiatives, namely, providing SC as a Service (SCaaS). Under this model, the SC software is hosted on a cloud platform using web- and application servers supported by storage, network, and management servers in a network of Data Centers (DC), to which the users connect via a web browser. A key advantage of SCaaS model is enabling access to high-end SC resources to users, who typically cannot afford to set up and maintain such infrastructure due to the high cost and expertise required. Users can access such resources via SCaaS, without any software installation or hardware purchase. Domain scientists from remote field locations, using even a low form-factor device such as a netbook, would also be able to access high-end simulation results, which can greatly aid field research and operations. High Availability (HA) and Disaster Recovery (DR) are facilitated due to the inherent replication and back-ups available on the cloud. Distributed applications and collaboratories, facilitating multiple scientists to collaborate on a project, are easily enabled via SCaaS. There is a cost advantage to the end users due to the economies of scale cloud computing provides, using consolidation, virtualization, elastic load management, cost sharing via subscriptions and multi- tenancy.

Further, new computing models such as SCaaS are likely to spur innovation within the SC domain itself, which spans across several fields such as biological, chemical, mathematical, energy, health, materials and environmental sciences. For example, taking advantage of the ease and economy of access, users can comparatively simulate and assess a given phenomenon using multiple simulation models, which improves the cognitive insights and the degree of confidence with which conclusions are drawn about hypotheses. Such benchmarking also improves the quality and robustness of the SC tools (simulation and visualization models) themselves. High end SC software, owing to their inaccessibility, often suffer from poor usability and functionality testing, which is known in Enterprise Software Engineering practice as 'Beta Testing'. SCaaS' improved accessibility can facilitate both benchmarking and Beta testing of SC software. In summary, SCaaS has the ability to extend the capabilities and reach of SC to much larger user-bases, at low costs.

2. SCaaS REQUIREMENTS

In addition to the usual scenarios SC software facilitates, newer usage scenarios become possible in the cloud environment. Scientific Collaboratories, where multiple scientists collaborate on a given workflow, and distributed software development and learning environments, where software development and training for SC are conducted on a cloud platform, are examples of such new scenarios [3]. The SCaaS model is inherently well-suited for such collaboratories. In a SCaaS environment, there are multiple Concurrent Users (CCU) who would access an instance of the same SC application and run their specific workflows simultaneously. For example, there could be multiple users from different geographic locations, accessing an oil reservoir simulator, to run carbon sequestration (injection of CO2 deep underground) simulations in a given reservoir. Each such simulation requires a dedicated session to the particular user, with their session and application data. Such data need to be persistent, when they stop work and resume to that simulation later. Additionally, these multiple users may be collaborating on a given workflow. For example, users from one location may be using the SC software for geologic characterization; a second group could be conducting seismic modeling and a third group conducting CO2 fate and transport modeling. In this case, it would be necessary to pool and merge input data as well as output data from multiple concurrent sessions into a common workflow. Results from seismic, geophysical and fate and transport model sessions could be juxtaposed into a single visualization to understand the current state of the fluids in the reservoir, for example. Such dynamic integration of results from multiple CCU's becomes possible due to SCaaS.

Expertise as a Service is another attractive feature of SCaaS. A small team of experts centrally located close to the SC application development group can serve as the Expert to facilitate the use and further development of SC tools by the wider user-base. An IT Admin would be responsible for packaging the SC software for access as a service. Each such scenario entails its own set of requirements. In all such usage scenarios, users need to be able to run the software the same way as they run it on their local desktops. The promise of Software as a Service (SaaS) is best fulfilled if the UX on the cloud platform closely approximates the local desktop UX. This is a close approximation because network conditions could impact the quality of the remote connection from the user's desktop to the SC software session running in a remote DC, and hence the SCaaS usability.

Requirements in these scenarios are best described referring to SCaaS Personas. In addition to the end-user persona, typically a scientist or engineer using the SC application, SCaaS also involves the IT Admin and Expert personas. It should be noted that SCaaS Admin and Expert typically are employees of the company or lab maintaining the SCaaS platform.

USER

1. User will be able to login to any of their SCaaS application modules with a single, one-time sign on.
2. Users can launch their SCaaS application environment within 20 seconds or less and be able to use the same.
3. User can perform input, scripting, coding, and visualization of SC workflows within the remote SCaaS environment the same way as they would on a local desktop.
4. Simulation results during the run, which typically takes hours to days, should be automatically and periodically saved to a location specified by the user and made available as needed.
5. User will be able to upload input files and access the results in typical data formats. They will be able to read, write to, copy, and save files and content from the SCaaS environment to their desktops and vice-versa.
6. Users can start, pause, and resume the simulations at their own pace, or as led by Expert in other scenarios.

EXPERT

1. Expert will be able to access the input files and simulations of any of the multiple concurrent users, modify the same, and provide feedback to the users.
2. Expert will be able to multicast presentation of a simulation workflow or demo to the user group.

SCaaS ADMIN

1. SCaaS Admin will be able to create, provision, manage, and delete virtual SCaaS environments for multiple CCUs, using the SCaaS Administrator Console.
2. SCaaS Admin will be able to collect and provide usability metrics, logs, and video capture, etc. back to the Expert and content developers.
3. SCaaS Admin will be able to monitor, track, and meter the usage of a given SC application in a non-repudiated manner, and clean-up stray or abandoned user sessions and workflows. Non-repudiation is important to SC for intellectual property protection.
4. SCaaS Admin will be able to meet the stringent authentication and security requirements to protect the IP, confidentiality and security requirements of SCaaS.
5. SCaaS Admin will not be able to view, copy, or save any of the simulation content if required to be kept confidential by some users.

The SCaaS platform presented in this paper was designed and implemented to meet these requirements. Design of experiments and discussion of results in this paper are focused on the experience of SCaaS application's use by the USER persona, which is most critical to the success of SCaaS model.

3. PLATFORM ARCHITECTURE

Meeting the above requirements in a multi-tenant platform requires isolation of the SC environments of each tenant at various layers of the computing stack (OS, application, network and storage, for example). A tenant here represents a single or distributed software application installation on the SCaaS platform, available for use by its authorized set of users. Multi-tenancy means multiple such applications would co-exist on the SCaaS platform, all being able to serve the remote users with an instance of the application on demand. This requires isolation of the application environments. Further, access to a given SC software by its multiple concurrent users requires isolation of the users' application sessions, computational and network resources. Both types of isolation of multi-tenant applications described here are achieved on the present SCaaS platform using virtualization, with 4 essential components: (i) a system virtualization layer created on the physical (DC) servers using a Virtualization software such as Microsoft Hyper-V, Citrix Xen Server or VMware ESX server; (ii) a presentation virtualization layer to create remote application sessions using Citrix Xen App or Microsoft Terminal Server; (iii) a custom-built deployment/management middleware layer comprising of a Management Server (MS), Management Console (MC) and Management Agent (MA) to orchestrate the SCaaS workflows, and a Web interface layer to present the SCaaS apps to the end users via the Internet, via a variety of Web 2.0 technologies, including Adobe Flash, Java and AciveX clients. To our knowledge, Eucalyptus [17] is the only other such available management middleware infrastructure for cloud computing.

User session isolation is achieved on the SCaaS platform using presentation virtualization, also known as 'graphics remoting', using protocols such as Microsoft's Remote Desktop Protocol (RDP) and Citrix' Independent Computing Architecture (ICA) protocol. When multiple concurrent users are accessing different applications, which may require different OS, systems virtualization is used to meet this need. In another scenario, if the SC application being used is a distributed system, which requires multiple applications and/or client and server roles to be instantiated with communication among them, each user may require their own OS instance. In such cases, the SCaaS platform creates Virtual Machines (VMs) which can run a wide variety of guest OS including Linux, Windows, Solaris and Mac OS, using VM solutions such as Hype-V, Xen Server and VMWare ESX server. If the workflow involves simulation of a distributed system, VMs networked as dedicated VLANs can create such environment for each participant. These VMs and virtual application sessions are hosted on high-end servers in the DC, which can provision large amounts of resources including RAM, CPU and storage on demand. Specialized graphics cards can be used to run GPU-aided SC workflows as well.

VMs are also an elegant solution for the problem of legacy application compatibility. SC applications in several domains today comprise of legacy systems and platforms, often more than a decade old, and legacy applications of significant complexity. This represents a major financial and engineering investment. Incompatibility of such legacy systems and applications (App Compat problem) typically is a major blocker to the adoption of new technology. This is likely to be a major blocker for the adoption of SCaaS as well, because majority of applications in diverse SC domains have a long legacy (several years to decades old) and run on a variety of OS, Linux being the predominant one. Keeping both the application compatibility and OS heterogeneity needs in mind, the present SCaaS platform is built on a mature systems and presentation virtualization platform, to address both needs in a rich way. It enables the running of virtual desktops, software development tools, Linux/Win32 apps, LOB and legacy apps which may be even incompatible with some operating systems, and the modern (.Net/java) web-apps as well, with seamless interoperability. Further, this platform could also be used as a cloud appliance to setup an organization's own private SCaaS cloud as well.

Figure 1. MS Console with pre-packaged VMs ready to deploy

Pools of VMs and remote application sessions are packaged, provisioned, managed and retired autonomically on the fly, depending on the user's workflow scope and requirements, using the Management Server (MS). Autonomy here refers to the system's ability to provision, pause and retire resources such as VMs and virtual application sessions necessary to complete a given SC workflow, without human intervention via scripting and copying etc. The MS is designed to support the configuration,

deployment and management of the remoting infrastructure for hosting processes, which could support Win32 applications, web applications as well as full virtual desktops. It enables the IT Admin to easily and autonomically create, provision, manage, save and delete such containers in a cloud environment. It has a configuration manager component to support the initial configuration of virtual application sessions as well as VMs, loading and arbitrating the various services in the service layer above the platform layer, logging and instrumentation support.

Shown in Fig. 1 is the IT Admin's view of the MS console, where several VM images are available to be created and deployed on the fly. The SCaaS Management Console is the user interface developed in Microsoft WPF to manage the servers and to send specific commands to specific servers (MA) and also to schedule specific commands on a specific server (MA) so that the commands are executed at the scheduled time without user intervention. It is installed on the IT Admin's desktop, to access and manage the SaaS VMs, apps and workflows. Also, in MC user can view the server status and the results of the commands that are executed and scheduled on the MA.

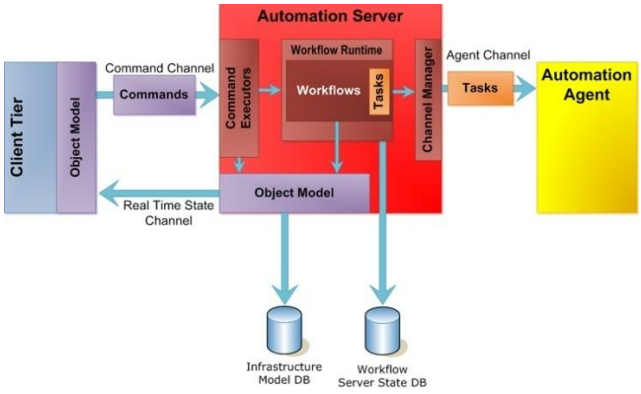

Figure 2. SCaaS Management Server Architecture

The service layer is where the services specific to each SC application's workflow are hosted. Authentication service, module navigation, metadata management, mail support and session tracking services are examples of such services, implemented on this platform in support of SCaaS. Configuration manager is designed to support the initial configuration of a large number of concurrent user sessions, support for loading and arbitrating the various services in the service layer above the platform layer, logging and instrumentation support. Its service layer provides a number of services specific to SC workflow. This platform's modular, componentized architecture allows easy extensibility for future scenarios on the SCaaS platform.

Shown in Fig. 2 is an architectural diagram representing the MS, which runs as a part of the SCaaS platform, as shown in Fig. 3. Management Agent component is the one which resides on every server which requires maintenance using RAP. MA is a Windows Installable which will need to be installed on every server role instance which serves the SaaS workflows. MA accepts the commands from MS and executes the command on that server and provides the output back to MS which in turn updates the MC. Management Server is the primary component which acts as a broker service between the MA and MC. MS consists of database, certificate authority, RunawareManagentServer windows service, and authorization manager. MS receives the commands from MC issued by the user from MC to a MA with other details and stores the data in database and sends the command to the respective MA with the format the MA can understand.

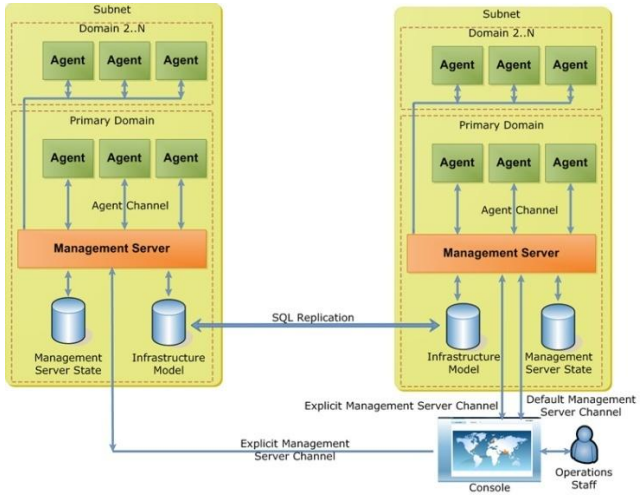

Figure 3. MS and Agents as a part of the SCaaS platform

This architecture is implemented using Microsoft .Net Workflow Foundation, as a set of configurable workflows. For example, invocation of a SC application session for N number of remote users is a work flow. The Automation Agent is responsible for collecting such work flows as inputs, and passing them on to the Automation Server. To achieve this, the Agent component, running simultaneously in the server (DC) and the remote user clients, orchestrates the necessary input data collection, and passes it on the Automation Server. The Automation Server takes several appropriate actions to create and support the workflow, such as a VM creation, application session creation and remote session brokering.

The SC application goes through an Application Lifecycle Workflow. This workflow captures application details from the users or experts, and the work items are assigned. Once the application is packaged and tested, IT Admin deploys the application into Staging area and send the test links to the Testing team. Production staff rolls out the application from Staging to the Production servers before the link is published to the end users. The application goes through the stages or packaging, staging, production (beta/main) and disablement (non-live).

In Fig. 3, a SCaaS DC outlay is shown, indicating how the MS server and agents fit into such architecture. In a global scale SCaaS deployment, multiple DCs are commissioned to support workflows in closer regions, without undue network latency. The system shown in Fig. 3, which manages the SCaaS platform presented here, would autonomically divert the traffic to a DC location in the U.S., EU or Asia, depending on its proximity to the user. Replication is used as shown, to ensure that state is synchronized at the DC and application levels periodically. An

additional design goal for this architecture was autonomic backup and redundancy, in support of High Availability (HA) and Disaster Recovery (DR).

4. SIMULATIONS

4.1 TOUGH2 Computational Model

TOUGH2V2.0, a scientific simulator for the simulation of flow and transport through geosystems [17], is used to demonstrate the SCaaS platform's architecture and its implementation. TOUGH2 is a general-purpose numerical simulation program for multi-phase fluid and heat flow in porous and fractured media, developed at the Lawrence Berkeley National Laboratory (LBNL). This code represents a complex, high-end scienctific computing model and has been used for many applications, including geothermal reservoir engineering, remediation of contaminated hazardous waste fields and nuclear waste disposal problems. PetraSim is a GUI for TOUGH2 codes to enable modelers to create grids and input files, run simulations and display results as visualizations, in an easy to use graphical user environment [9]. PetraSim was developed by Thunderhead Engineering Consultants Inc., and is available on the Web [18, 19]. TOUGH2 was especially selected for this work, to investigate how a high-end scientific computing code would behave on the cloud in terms of basic functionality and usability.

4.2 WANem Network Simulator

Network conditions of different quality were simulated in the cloud environment using WANem, a Wide Area Network Emulator [20]. WANem is a Wide Area Network Emulator, which simulates the network behavior of a Wide Area Network/Internet, during application development and testing over a LAN environment. Typically application developers develop applications on a LAN while the intended purpose for the same could be, clients accessing the same over the WAN or even the Internet. WANem thus allows the application development team to setup a transparent application gateway which can be used to simulate WAN characteristics like Network Delay, Packet Loss, Packet Corruption, Disconnections, Packet re-ordering, Jitter, etc. WANem can be used to simulate WAN conditions for Data/Voice traffic and is released under the widely acceptable GPL v2 license. WANem provides emulation of Wide Area Network characteristics and thus allows data/voice applications to be tested in a realistic WAN environment prior to production

5. RESULTS AND DISCUSSION

A set of four simulations were conducted and visualization of their results analyzed, using TOUGH2V2.0 as a Service. The workflows included the production from a Geothermal Reservoir with Hypersaline Brine and CO_2, Diffusive and advective spreading of volatile organic contaminants (VOCs) in the vadose zone, Radial Flow from a CO_2 Injection Well and Gas diffusion of an organic chemical with phase partitioning through porous media. One complete set of simulations was conducted using these examples in PetraSim, installed on a HP Desktop system running Windows 7 OS. The same set was run separately on the SCaaS platform. On this platform, PetraSim was installed within a VM running Windows Server 2008 OS installed on the blade server in a XenDesktop environment and remotely accessed from the above desktop using Citrix ICA protocol. All of the simulations were accessible and usable with a good quality of user experience, not significantly different from the corresponding local desktop experience.

Shown in Fig. 4 A and B are results from the simulation of a 3D five-spot geothermal production simulation on a local desktop (A) and in the SCaaS environment (B). This is a simulation for a five-spot pattern of injection and production of a geothermal reservoir. Visualization shows temperature contours and water flow vectors by the end of 36.5 years.

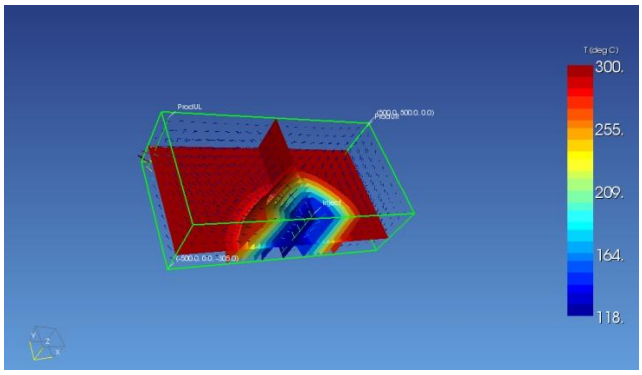

Figure 4 A. 3D Temperature contours in the reservoir: Desktop installation

Along the side of the temperature contours is scale which indicates the temperature gradient from 118°C to 300°C. In such visualizations, user expects to visually differentiate the saturation value of the parameter being contoured (temperature in this case), correlate the differences in this value across the visual field, identify and understand all interesting trends. This was facilitated well in all the visualizations presented, both within and outside of the SCaaS environment. For example, the red hot regions of temperature and directionality of flow vectors are clearly visible in all cases (Fig. 4 and Fig. 5).

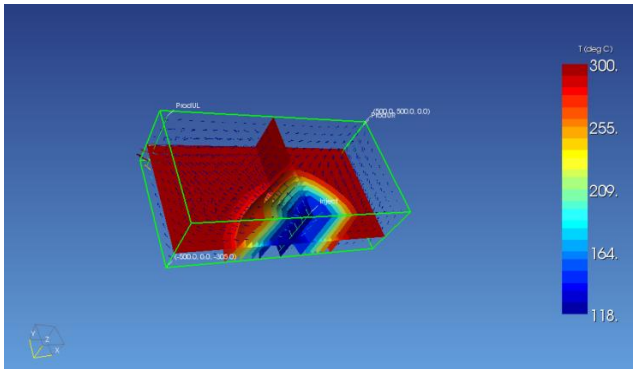

Figure 4 B. 3D Temperature contours in the reservoir: SCaaS installation

Similarly, shown in Fig 5 A and B are results from the simulation of a 2D five-spot geothermal production simulation on a local desktop (A) and in the SCaaS environment (B). This is a simulation for a 2D five-spot pattern of injection and production of a geothermal reservoir. Visualization shows temperature contours by the end of 36.5 years. Along the side of the

temperature contours is the scale which indicates the temperature gradient from 118°C to 299°C.

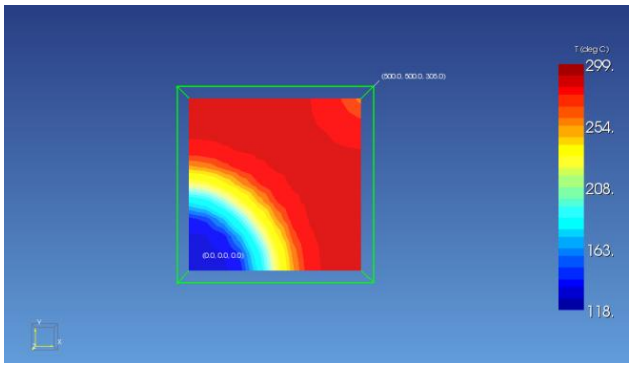

Figure 5A. 5 spot geothermal simulation: Desktop installation

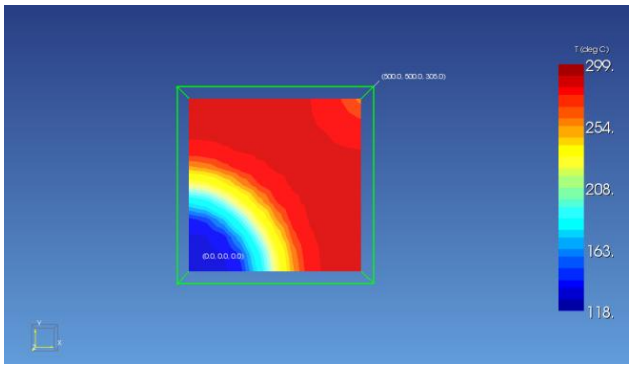

Figure 5B. 5 spot geothermal simulation: SCaaS installation

In both cases, the user was able to clearly visualize the temperature gradients and the water flow vectors. No perceptible differences were apparent between the simulations. Hence the simulation and visualization of the geothermal model on the SCaaS platform were accomplished with good usability.

Final visual quality is a very important aspect of presentation virtualization, and hence SCaaS platforms. To further stress the SCaaS platform, the simulations shown in Fig. 4B and 5B were repeated, introducing a network Delay of 100ms via the WANem simulator. No significant degradation in the UX was found in the quality of simulation or visualization displays, and the user was able to complete the runs well. What this implies is that the SCaaS platform presented here presents the SC application's view to the user with good graphic quality, even when there is several 100 ms of Delay over the network. However, the final presented graphic quality is only one important aspect of SCaaS usability. There are several other parameters which together influence the SCaaS usability. We have defined eight such usability metrics for SCaaS, summarized in Table 1. Among these, UM7 represents the graphic's color quality explained above, whereas the remaining metrics refer to the dynamic aspects of usability of an application, such as the time taken to launch the application and rotate or pan a graphic etc. These dynamic metrics are very critical to the overall satisfaction and goodness of UX.

Keeping this in view, from among the larger set of simulations, two representative simulations were selected for a detailed usability evaluation, under progressively degrading network conditions simulated using WANem. The first simulation (S1) demonstrates a basic reservoir analysis, including an initial state calculation and then a production analysis, illustrating the use of color contours to define the model surface, so that the surface represents a varying topography. Surface boundary conditions, material and chemical properties were specified using the recommended values from PetraSim user manual. The second simulation (S2) represents One Dimensional Gas Diffusion of an Organic Chemical, which is a Non-Aqueous Phase Liquid (NAPL) with Phase Partitioning. Material and chemical properties were suitably chosen, using the recommended values from the PetraSim user manual. All of the reservoir cells were initialized to a two-phase water and air state, without any initial chlorobenzene saturation. A detailed analysis of the flow and chemical transport aspects will not be presented as this paper is focused on demonstrating the architecture, implementation, and usability aspects of SCaaS.

5.1 Usability Assessment

In the context of SC applications, graphics performance maps to two categories of UX: (1) visual and mechanical fidelity of routine graphical input and response events carried out by a typical user, and (2) the speed and agility with which graphics intense applications are rendered and displayed on the desktop.

Usability of TOUGH2V2.0 in the cloud environment were assessed vis-à-vis the same on traditional desktop (local) environments using several UX metrics (Table 1). These metrics included the time to launch the application, rotational movement of graphics during visualization, translational movement of graphics during visualization, mouse movement, typing (keyboard) agility, and Windows movement (minimizing and maximizing child windows etc). Quality of color display in graphics is also assessed. Each of these metrics is assigned a value on a scale of 0 to 10, 0 being very poor quality implying the system is unusable, and 10 being excellent quality, corresponding to using the application on a local desktop.

These simulations were repeated imposing the WANem simulator, in the network path between the user's desktop and the SCaaS DC server, under varying conditions of network quality. Network Delay, Packet loss, Packet corruption and Jitter were varied as shown in Table 2. Delay (sometimes called Latency) is the one way delay from one node to another. Latency is proportional to the physical distance covered by the route taken by a packet from source to destination. Real networks show variation in delay, which is referred to as Jitter. For e.g., if Delay is 100ms and Jitter is set to 10 ms, the delay applied is 100 + 10 ms or 100 – 10 ms in random. Sometimes the networks drop packets for various reasons, congestion and signal loss being the common factors. In WANem, packet loss is specified in percent. A loss value of 0.1 causes 1 out of 1000 packets to be randomly dropped.

First a set of simulations was run in the local desktop environment, outside of the SCaaS platform. Then, PetraSim was packaged in the cloud (SCaaS) environment, and a desktop running Windows 7 OS was used to connect to PetraSim running on a HP ProLiant BL G6 Server. To simulate the real WAN environment connecting the local desktop to the cloud, WANem network simulator was placed in the network path. The same set of simulations was run in the SCaaS environment, by varying Delay form 0 to 2000 ms. Shown in Table 3 and 4 are the usability indices for S1 and S2 respectively, keeping the Delay variable constant at 100 ms. A usability test engineer evaluated

each metric on a 1-10 scale as shown in Table 1. These findings are recorded for simulations S1 and S2, in Tables 3 and 4.

As discussed earlier, the user (scientist) expectation of usability here would be that (i) the simulations can be completed with satisfactory agility and response from all desktop operations and screen (windows) movements, and (ii) during visualization of results, trends and differences across the visual field in the parametric values being visualized via contouring etc. can be easily perceived, understood and interpreted.

Table 1. Summary of Usability Metrics for SCaaS

UM #	Metric	Explanation
UM1	Application Launching Time	Quickly launches (10); takes more than 45 sec to launch (0)
UM2	Rotational Movement	Very smooth (10); very slow and intermittently frozen (0)
UM3	Translational Movement	Very smooth (10); very slow and intermittently frozen (0)
UM4	Mouse Movement	Very smooth (10); very slow and intermittently frozen (0)
UM5	Keyboard Typing	Agile without lag (10); lags beyond next keystroke (0)
UM6	Windows Movement	Very smooth (10); very slow and intermittently frozen (0)
UM7	Color quality	Color is perfect (10); clear fading and pixel corruption (0)
UM8	Window Pan	Very smooth (10); very slow and intermittently frozen (0)

This scale was designed using TOUGH2V2 running in PetraSim on the local desktop, observing multiple representative simulations. A UM value above 6 represents a satisfactory to excellent UX, whereas a value less than 6 represents unacceptable UX, experienced as slow, heavy and broken movement of graphical windows, and slow mouse and keystroke response.

All eight UM consistently showed degradation as the Delay increased. This is reasonable, because presentation virtualization on the SCaaS platform works by remoting graphical output from the DC servers to the client. In this model, when a user first launches an application running on the SCaaS servers, a client-side component establishes and maintains the connection between the local client and the SCaaS DC server. All of the input from the user, such as keystrokes and mouse movements, is transmitted to the server. All output from the server such as application display information and print streams are sent back to the client, typically as bitmaps, which are rendered as the application's output on the user's (client) screen. Microsoft's Remote Desktop Protocol (RDP) and Citrix ICA protocol are some prominent examples of presentation virtualization. This process involves a roundtrip transmittal of packets between the client and server frequently (for every mouse click and key stroke). As such, network latency or delay can have a detrimental impact on the agility and quality of graphics display.

Results from Table 2 confirm this. For example, time to launch the application increases with Delay because both the launch request to the server and the presentation response back to the client experience delay. However, usability was observed to be unacceptable, i.e., causing frustration to the user, only when Delay was around 100 ms or higher. When Delay increases beyond this threshold, all of the UMs registered a marked slowness or drag, rendering the UX increasingly unacceptable and painful. The UMs U2 to U8, excepting U7 are all impacted by the same latency effect in the same way (graphic transmission and rendering are delayed). This explains the consistency in their degradation trend with Delay. It appears that a network delay of more than 100 ms should be avoided for deploying applications on SCaaS platforms. This is achievable in today's business networks.

Table 2A. Influence of Delay on SCaaS Usability

Delay	UM1	UM2		UM3		UM4	
	Time	S1	S2	S1	S2	S1	S2
0	4	7	8	7.5	8.5	9	9
50	7	6	7	6.5	7.5	8	8
100	13	5	6	5.5	6.5	5	5
250	18	3	4	3.5	4	4	4
500	25	1	2	1.5	2	3	3
1000	42	0	1	0	1	1	1
2000	>2min	0	0	0	0	0	0

Shown in Fig 6 A and B are the visualizations of heat upflow and the cooling downflow of a basic reservoir at the end of 19989.27 years along with the temperature gradient at a corruption of 0% and 4%. It can be seen that the graphics look very similar, indicating that the visualization quality is not impacted by Delay or the fact that the application is running in a SCaaS environment.

Table 2B. Influence of Delay on SCaaS Usability (Contd.)

Delay	UM5	UM6		UM7		UM8	
	S1	S2	S1	S2	S1	S2	S1
0	9	9	9	9	9	9	9
50	8	8	9	9	8	8	9
100	7	7	8	8	7	7	8
250	6	6	7	7	6	6	7
500	4	4	7	7	4	4	7
1000	3	3	4	4	3	3	4
2000	2	2	2	2	2	2	2

These 3D contour data (A) corresponds to the simulation where packet corruption is not introduced using WANem whereas (B) represents the same simulation but affected by a packet corruption of 4%. It can be seen that the visual and color quality on SCaaS Platform is slightly degraded (B) with packet corruption, in the form of foreign banding and slight fading of color in some regions. However, this does not perceptibly influence the overall insights drawn from the visualization.

Table 3A. Influence of Network Quality on SCaaS Usability

Delay = 100	UM1	UM2		UM3		UM4	
Loss	**Time**	**S1**	**S2**	**S1**	**S2**	**S1**	**S2**
0	13	5	6	5.5	6.5	5	5
2	17	4.5	5.5	5	6	4.5	4.5
4	21	5	6	5.5	6.5	4	4
Jitter	**Time**	**S1**	**S2**	**S1**	**S2**	**S1**	**S2**
0	15	5	6	5.5	6.6	5	5
2	20	4	5	5	6	4.5	4.5
4	24	3	4	4	5	3	3
Corrupt	**Time**	**S1**	**S2**	**S1**	**S2**	**S1**	**S2**
0	15	5	6	5.5	6.5	5	5
2	21	5	6	5.5	6.5	5	5
4	25	4.5	5.5	5	6	4.5	4.5

From Table 3A, it was also observed that only the packet corruption parameter has such detrimental effect on visualization, whereas the other parameters such as Delay and Jitter did not have a similar influence. It should be noted that the application launch time delay is 13 ms until the Delay is 100 ms, and is not perceptibly impacting the usability of the application. However, after a Delay of 100 ms, the usability is significantly worse, because the application launch time increases to more than 20 sec. Accordingly, for further simulations, Delay was held constant at 100 ms, and usability is assessed by varying packet loss, corruption and jitter between 0 to 4%. Here, packet loss is defined as the % random drop of network packets. For example, a loss value of 0.1 causes 1 out of 1000 packets to be randomly dropped. Similarly, packet corruption is defined as the deliberate altering of a % of packet's data with spurious data.

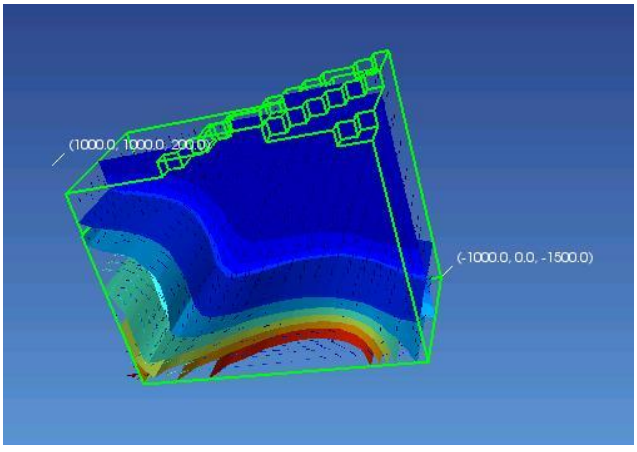

Figure 6 A. Color Quality (U7) of the simulation without the introduction of corruption through WANem

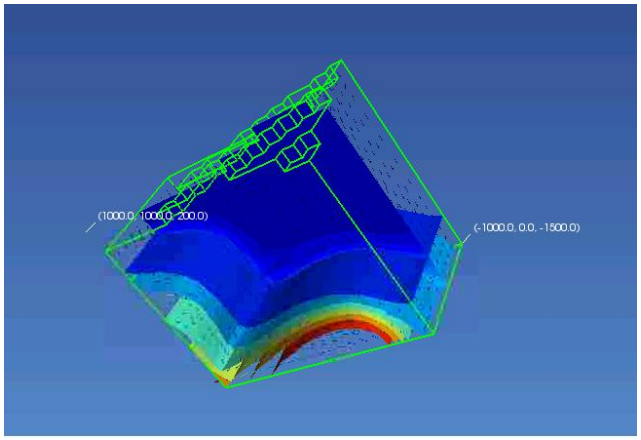

Figure 6 B. Influence of Packet Corruption on Color Quality

Table 3B. Influence of Network Quality on Usability (contd.)

Delay = 100	UM5		UM6		UM7		UM8	
Loss	S1	S2	S1	S2	S1	S2	S1	S2
0	7	7	8	8	9	9	5	5
2	7	7	7.5	7.5	9	9	4.5	4.5
4	6.5	6.5	7	7	9	9	4	4
Jitter	**S1**	**S2**	**S1**	**S2**	**S1**	**S2**	**S1**	**S2**
0	7	7	8	8	9	9	5	5
2	4.5	4.5	7	7	9	9	4.5	4.5
4	3	3	6	6	9	9	4	4
Corrupt	**S1**	**S2**	**S1**	**S2**	**S1**	**S2**	**S1**	**S2**
0	7	7	8	8	9	9	5	5
2	7	7	8	8	9	9	5	5
4	6.5	6.5	7.5	7.5	8.5	8.5	4.5	4.5

Results from these simulations showing the effect of additional degradation in network quality, introduced via increasing Packet loss, Jitter and Corruption, are shown in Table 3. To obtain this data, simulations S1 and S2 were repeated, keeping the network Delay constant at 100 ms, and varying the 3 aforementioned network parameters separately. Due to the constant 100 ms Delay imposed on all simulations, all UM registered some degradation in usability, even when the Loss, Jitter or Corruption parameters are set to zero. Among them, the Application Launch Time (UM1) was clearly impacted similarly by all 3 network parameters. At the maximum values of Loss, Jitter and Corruption used, the application PetraSim on the SCaaS platform launched about 80% slower than in their absence, taking more than 20 seconds. Application launch time of more than a few seconds begins to degrade UX, whereas more than 10 seconds causes user fatigue. Among the other metrics, U2 (Rotational Movement), UM3 (Translational Movement) and UM8 (Windows Panning or rapid movements across the screen) showed more marked degradation.

This is to be expected, because in all of these actions, the user is rapidly changing the position of the graphic across the screen, and

each such movement requires one roundtrip transport of data over the wire, between the client and DC. Increasing packet loss and packet corruption does have a moderate additional negative influence on these UM, whereas Jitter has a more pronounced impact. Further, increasing jitter shows a clearly worse impact on the Mouse movement and Keyboard typing metrics. Since both of these involve the display and visualization of a very small (pointed) area in the pixel space, jitter, which is the variability in latency or delay, shows the corresponding jitteriness which the user finds unacceptable. When the same jitter is observed over a larger area (i.e. a whole window) as in the case of the other UM (such as UM2, 3, 6 and 8), its impact on UX is found to be more tolerable.

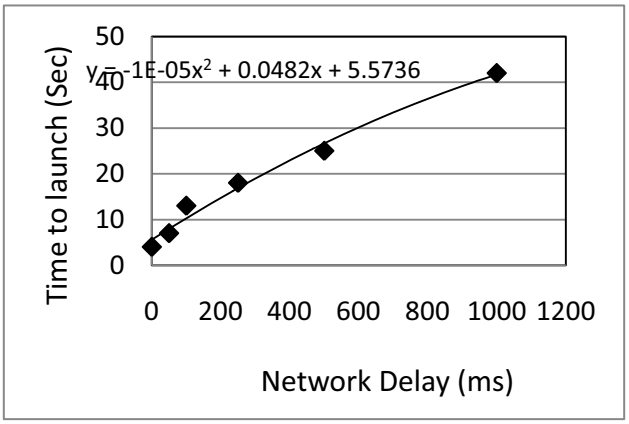

Figure 7. Influence of Delay on App launch time (U1)

Furthermore, the graphics color quality metric (UM7) is influenced by an increase in the Corruption parameter, whereas the effect of Jitter and Loss on color quality is not as pronounced. It is well known in computer graphics literature that techniques such as anti-aliasing can address packet loss (loss of data) effectively, whereas packet corruption represents the presence of spurious data, which is rendered 'as is'. This effect can be seen in Fig. 6B as well, as fading and foreign banding of color. However, user's eye is able to make out the overall color contour trends for the given scientific simulation adequately, and as such the effect of corruption on UM7 is not reported as dramatic (Table 3).

Shown in Fig. 7 is the influence of Delay on the Application launch time (UM1). The SCaaS platform is usable with good quality on all UM metrics as long as the Delay does not exceed 100 ms and the Loss, Jitter and Corruption parameters are less than 2%. In summary, Application Launch Time is most adversely impacted by degradation in network quality as represented by Loss, Jitter and Corruption, followed by the Mouse Movement, Keystroke Display and Color Quality.

6. CONCLUSIONS

Architecture, implementation and the evaluation of functionality and usability (UX) of a major reservoir engineering a geosystems fate and transport model (TOUGH2) on a global, Enterprise cloud platform were presented for the first time. A significant contribution of this work is that the selected scientific application can be deployed and used on a true, global cloud platform without any functionality issues. Further, the SCaaS platform should maintain network latency less than 100 ms, for acceptable usability. A more comprehensive usability study, expanding the parameter matrix shown in this study and using a larger number of sample set (*i.e.* number of SC simulations etc) is necessary to conclusively establish the influence of network quality on SCaaS platforms' usability.

6.1 Limitations and Future Work

It would be of high interest to examine the impact of the cloud infrastructure itself, location management, consistent presentation of workloads among multiple tenants, elastic provisioning in response to demand and data distribution among distributed workflows etc [21, 22, 23]. We have focused first on the necessary fundamentals of functionality and usability in a true cloud context for the first time, proof of which in the present deployment are novel, because the architecture and implementation of the cloud provisioning, deployment and management workflows itself constitute a necessary, novel first step for SCaaS. Further, autonomic provisioning, deployment and management of what was able to run only as a locally installable application into a true SaaS application, using a cloud workflow management engine during runtime especially built for this purpose, is novel and necessary. Among the commercially available Saas or SoA platforms, there is no platform which can host a complex, Locally Installable Applications (LIA) as a service on an IaaS platform, and make it accessible via the Internet independent of the end user device characteristics. For example, it requires significant new middleware and workflow implementation to present SCaaS of the type presented here on Amazon AWS. Google App Engine and Microsoft Azure, which focus on Web apps, are not readily suited to present LIA such as TOUGH2 as a service. This is bulk of the core contribution of the present work, which finds immediate, novel applications among the reservoir engineering community. Furthermore, the UX evaluation methodology presented here is first of its kind for any remoting systems which assessed latency effects on remoting [23], and certainly for a SaaS application in SC. We have presented an assessment of the additional, more advanced cloud principles of the same platform, such as the elastic provisioning in response to demand via load prediction and hotspot detection, elsewhere [25]. It is necessary to assess the influence of the remaining cloud fundamentals such as elasticity, security and multi-tenancy on SCaaS, which is a limitation of the present work.

7. ACKNOWLEDGMENTS

We are thankful to the production staff at Runaware Corporation for help with the SCaaS deployment, to Thunderhead Engineering for PetraSim evaluation license, and the anonymous reviewers whose feedback helped revise this work.

8. REFERENCES

[1] E.W. Bethel, H. Childs, A. Mascarenhas, V. Pascucci, Prabhat. "Scientific Data Management Challenges in High Performance Visual Data Analysis," In Scientific Data Management: Challenges, Existing Technology, and Deployment, Chapman Hall/CRC Press, 2008.

[2] C.D. Hansen, C.R. Johnson, V. Pascucci, C.T. Silva. "Visualization for Data-Intensive Science," In The Fourth Paradigm of Scientific Discovery, Edited by Dan Fay, Kristin Tolle and Tony Hey, Microsoft Research, 2009.

[3] Electronic Notebooks: An Interface Component for Semantic Records Systems, James D. Myers, Michael Peterson, Prasad Saripalli, Tara Talbott, 227th American Chemical Society (ACS) National Meeting, Anaheim, CA, March 28-April 1, 2004.

[4] D. Thain, T. Tannenbaum, and M. Livny, "Distributed computing in practice: The condor experience," Concurrency and Computation: Practice and Experience, vol. 17, pp. 323–356, February, 2005.

[5] Foster, I. Yong Zhao Raicu, I. Lu, S. (2008), pp. 1-10; Cloud Computing and Grid Computing 360-Degree Compared, Grid Comp. Environments Workshop. GCE '08.

[6] Christian Vecchiola, Suraj Pandey, Rajkumar Buyya: High-Performance Cloud Computing: A View of Scientific Applications. ISPAN 2009: 4-16.

[7] R. Pordes, D. Petravick, B. Kramer, D. Olson, M. Livny, A. Roy, P. Avery, K. Blackburn, T. Wenaus, F. Wurthwein, I. Foster, R. Gardner, M. Wilde, A. Blatecky, J. McGee, and R. Quick, "The open science Grid," Journal of Physics: Conference Series, vol. 78, no.1, 2007, pp. 012–057.

[8] F. Gagliardi, M.E. Begin, "EGEE - Providing a Production Quality Grid for e-science," Local to Global Data Interoperability - Challenges and Technologies, Sardinia, Italy, June, 2005.

[9] C. E. Catlett, "TeraGrid: A Foundation for US Cyberinfrastructure," in Network and Parallel Computing, LCNS vol. 3779, H. Jin, D. Reed, and W. Jiang Eds., Springer Berlin /Heidelberg, 2005.

[10] Nimbus Project http://workspace.globus.org/clouds/nimbus.html/, access on February 9, 2010.

[11] Stratus Project [URL]. http://www.acis.ufl.edu/vws/, access on February 9, 2010.

[12] OpenNEbula Project [URL]. http://www.opennebula.org/ , access on February 9, 2010.

[13] Reservoir Project [URL]. http://www-03.ibm.com/press/us/en/pressrelease/23448.wss/, access on February 9, 2010.

[14] L. Wang, J. T. Kunze, M. Castellanos, A.C. Kramer, D. Karl, W. Scientific Cloud Computing: Early Definition and Experience in: High Performance Computing and Communications, 2008. HPCC '08. 10th IEEE International Conference; Sept. 2008, pp: 825 – 830.

[15] Scientific Computing in the Cloud J. J. Rehr, J. P. Gardner, M. Prange, L. Svec and F. Vila (2008) Department of Physics, University of Washington, Seattle, WA. Report http://arxiv.org/abs/0901.0029

[16] http://www.sgi.com/company_info/newsroom/press_releases/2010/february/cyclone.html , accessed on Jan 12, 2010.

[17] K. Pruess, C. Oldenburg, and G. Moridis. TOUGH2 User's Guide, Version 2.0. November, 1999. Earth Sciences Division, Lawrence Berkeley National Laboratory. Berkeley CA USA . LBNL-43134.

[18] PetraSim: A Graphical User Interface for the TOUGH2 Family of Multiphase Flow and Transport Codes, Ground Water; Volume 46 Issue 4, Pages 525 – 528 (2008).

[19] PetraSim. www.thunderheadeng.com/petrasim/index.html, access on January 14, 2010.

[20] http://wanem.sourceforge.net/ accessed on January 9, 2010.

[21] Saabeel, W.; Verduijn, T.; Hagdorn, L. & Kumar, K. (2002), A Model for Virtual Organization: A structure and Process Perspective, in 'Electronic Journal of Organizational Virtualness'.

[22] Gubala, Tomasz, Bartosz Bali, Maciej Malawski, Marek Kasztelnik, Matthias Assel, Daniel Har, Tomasz Barty, et al. 2008. ViroLab Virtual Laboratory. In Proc. 7th Cracow Grid Workshop, 35-40. Krakau: ACC CYFRONET AGH.

[23] Kipp, Alexander, Lutz Schubert, Matthias Assel, and Terrence Fernando. 2008. Dynamism and Data Management in Distributed, Collaborative Working Environments. In Proc. 8th International Conference on the Design of Cooperative Systems, 16-22.

[24] Miller, R. B. (1968). Response time in man-computer conversational transactions. Proc. AFIPS Fall Joint Computer Conference Vol. 33, 267-277.

[25] Prasad Saripalli, GVR Kiran, Ravi Shankar R, Harish Narware and Nitin Bindal (2010) 'Load Prediction and Hot Spot Detection Models for Autonomic Cloud Computing,' (under review) USENIX-HotICE, 2011.