Computational Engineering in the Cloud: Benefits and Challenges

Lorin Hochstein, USC Information Sciences Institute, USA

Brian Schott, Nimbis Services, USA

Robert B. Graybill, USC Information Sciences Institute, USA

ABSTRACT

Cloud computing services, which allow users to lease time on remote computer systems, must be particularly attractive to smaller engineering organizations that use engineering simulation software. Such organizations have occasional need for substantial computing power but may lack the budget and in-house expertise to purchase and maintain such resources locally. The case study presented in this paper examines the potential benefits and practical challenges that a medium-sized manufacturing firm faced when attempting to leverage computing resources in a cloud computing environment to do model-based simulation. Results show substantial reductions in execution time for the problem of interest, but several socio-technical barriers exist that may hinder more widespread adoption of cloud computing within engineering.

Keywords: *Case Study, Cloud Computing, Computational Engineering, Computational Science, Finite Element Analysis, Finite Element Method, Model-Based Simulation*

INTRODUCTION

Cloud computing has recently emerged as a service model where users obtain short-term access to large-scale computational resources, potentially at lower cost than purchasing and administering computing hardware (Armbrust et al., 2009). Most of the initial drive and interest in cloud computing has been in the IT community. More recently, there has been growing interest on using cloud computing as a platform for computational science and engineering (Shainer et al., 2010)

DOI: 10.4018/joeuc.2011100103

The term *computational science* refers to the application of computers for solving scientific problems, particularly the use of computer simulations to predict physical phenomena (Post & Votta, 2005). The term *computational engineering*, analogously, refers to the application of computers in solving engineering problems (Post, 2009). While there are many similarities between computational science and computational engineering, important differences also exist. Computational engineers use computers to do *virtual prototyping*, analyzing the behavior and failure modes of proposed designs through model-based simulations. Virtual prototyping has the potential to reduce both engineering

development time and cost by reducing the amount of physical prototyping required to do a design validation, as well as opening up possibilities for design optimizations. Commercially available engineering packages put these simulation techniques within reach of the end-user engineer, although a high degree of domain expertise is required to set up and interpret the results of such simulations.

Engineering simulations are extremely computationally intensive, with simulations taking anywhere from hours to days or weeks, depending on the type of simulation required. Many of these commercial engineering packages have support for running on high-performance computing (HPC) systems, and a survey of larger engineering firms indicates that such firms take advantage of HPC (Joseph et al., 2004). However, such systems are expensive to maintain and require additional IT expertise, rendering them inaccessible to many smaller engineering firms.

In this paper, we describe a feasibility study undertaken by the authors to help determine whether the use of remote HPC resources for modeling-based simulation would have a positive return on investment for a small-to-medium-sized manufacturing company. This paper describes our experiences, including the benefits of reduced processing time, as well as practical challenges that we faced while supporting computational engineers in using remote HPC resources.

RELATED WORK

Armbrust et al. (2009) provide a broad overview of the costs, benefits, and challenges of cloud computing. Although they do not focus specifically on scientific and engineering applications, they discuss several issues that appear in this study, such as batch processing of parallel processing applications, compute-intensive desktop applications, data transfer bottlenecks, and data licensing issues.

Cloud computing for computational science and engineering is a very young but increasingly active area, as evidenced by new workshops emerging in 2010 such as the first Workshop on Science Cloud Computing (ScienceCloud) (http://dsl.cs.uchicago.edu/ScienceCloud2010/) and Cloud Futures 2010: Advancing Research with Cloud Computing Workshop (Faculty Connection, 2007). Some early experience reports have begun to emerge. Hoffa et al. (2008) explored the use of cloud computing for executing a scientific workflow in the field of astronomy. Lauret and Keahy used cloud computing resources to quickly perform a preliminary analysis of a nuclear physics experiment in time to submit a conference paper (Heavy, 2009).

The MapReduce programming model has dominated much of the early interest in cloud computing applications. Dean and Ghemawat (2004) introduced the MapReduce model of processing datasets on large clusters, which has been implemented by the open-source Hadoop project (Bialecki et al., 2007). Much of the current cloud computing research focuses on data-intensive applications that map well to this model, such as indexing of spatial databases (Cary et al., 2009), processing very large graphs (Zhao et al., 2009; Cohen, 2009), and indexing of very large text corpora for information retrieval (Callan & Kulkarni, 2009)

There are also ongoing research projects to develop software infrastructure to support the creation of private clouds for scientific use. These efforts include Eucalyptus (Nurmi et al., 2009), OpenNebula (Sotomayor et al., 2008), Nimbus (Keahy et al., 2009), and Cumulus (Wang et al., 2008).

THE COMPUTATIONAL ENGINEER AS END USER

There is substantial overlap between computational scientists and computational engineers. Both of them model physical phenomena and simulate these models using computers to estimate the physical behavior of interest. Many of the underlying physical and mathematical computational machinery are the same (e.g.,

finite element methods, sparse linear solvers). However, the goals and contexts of these two groups differ.

Case studies of computational science projects to date suggest that most such code is "research code", written in a research lab such as a university or government-sponsored research environment. Examples include Hochstein and Basili's (2008) case studies of five university-based projects funded by the U.S. Department of Energy, Carver et al.'s (2007) retrospective case studies of five projects sponsored by various U.S. government agencies, and Easterbrook and Johns' (2009) ethnographic study of software development practices in a large U.K. government-funded climate research lab. Computational scientists will typically either write their own research code, or use someone else's, but it is rare to find commercially available computational science software that does model-based simulation (*Gaussian*, a commercial software package for computational chemistry, is a notable exception). By contrast, in computational engineering there is a significant market for commercial software that performs model-based simulation, such as ANSYS, FLUENT, MD NASTRAN, NEI NASTRAN, ABAQUS, STAR-CD, LS-DYNA and COMSOL Multiphysics. These tools enjoy wide commercial usage, and many are now able to take advantage of HPC systems. The effective result is that the computational engineer is much more of a traditional end-user, seldom called upon to write software directly.

Because commercial software packages must support a wide range of simulations, and because significant domain knowledge is required to understand the different simulation options, they are very complex pieces of software. A computational engineer must devote considerable time and effort to master the use of one of these commercial software packages. While many packages have a modern graphical user environment, they also have extensive command languages to support playback and batch processing. For example, the ANSYS software package implements a scripting language called the ANSYS Parametric Design Language (APDL), which has a FORTRAN-like syntax. In effect, these commercial tools can be thought of as domain-specific, graphical programming environments. Figure 1 shows some ANSYS commands, based on a tutorial by Quon and Bhaskaran (2002).

The end-goals of the activities of computational scientists are described in the case studies performed by Hochstein and Basili (2008), Carver at el. (2008) and Easterbrook and Johns' (2009), as well as by Segal (2005) and Basili et al. (2008). Based on our observations, the goals of computational engineers differ. In general, a computational scientist is interested in learning about particular physical phenomena for its own sake. The computational engineer is interested in validating or optimizing the performance of a particular product to be manufactured and sold. These differences in goals can affect the time constraints that they work under (e.g., product life cycles vs. publication deadlines). In addition, being in a corporate environment impacts the exchange of information with others outside of the organization. Even information exchange with customers and suppliers about computer-based models can be limited due to concerns about trade secrets.

Another significant difference between computational science and computational engineering is the role of verification and validation. In many computational science applications, the motivation for using computer simulation is that running physical experiments as an alternative are either prohibitively expensive, or practically impossible. By contrast, in computational engineering, computer simulation is part of a larger process that will involve the construction and testing of physical prototypes. The computational scientist may never discover how well his or her simulations approximate reality. For better or for worse, the computational engineer will eventually find out.

Figure 1. Example ANSYS commands for defining a bicycle crank

```
RESUME, crank,db,
/PREP7
! Add the Mesh Facet 200 element
ET,1,MESH200
! Add the Brick 8node 45 element
ET,2,SOLID45
! For Mesh200, we use 3D quadrilateral with 4 nodes
! K1: QUAD 4-NODE)
KEYOPT,1,1,6
KEYOPT,1,2,0
! Specify structural, linear, elastic, isotropic material
! Young's modulus (EX)=2.8E7, Poisson's Ratio (PRXY)=0.3
MPTEMP,,,,,,,,
MPTEMP,1,0
MPDATA,EX,1,,2.8E7
MPDATA,PRXY,1,,0.3
! Rectangle
BLC4,-3.3465,0,3.3465,1.299
! Big circle
CYL4,-3.3465,.6495,.6495
! Smaller circle
CYL4,-3.3465,.6495,.25
```

STUDY CONTEXT

Study Motivation

The case study described here is one of a series of studies funded by the Defense Advanced Research Projects Agency (DARPA) to determine whether manufacturing companies in the U.S. Department of Defense's supply chain can benefit from using high-performance computing for model-based simulation. There is increasing concern among policymakers in the United States about the decline of the U.S. manufacturing industry over the past several decades (National Academies, 2007). This is of particular concern to the U.S. Department of Defense because of the potential national security risks involved in becoming dependent on foreign manufacturers.

Because of the structure of the global economy, American manufacturers cannot compete with foreign manufacturers on labor costs. Therefore, American manufacturers are required to innovate to remain competitive in a global marketplace (Helper, 2009). The application of model-based simulation in engineering design has been proposed as one such innovation to improve the competitiveness of the manufacturing sector (Glotzer, 2009), with higher-fidelity simulations performed using high-performance computing (Council on Competitiveness, 2009). The motivation for this study is to understand the practical benefits and obstacles that a manufacturing company would face when trying to leverage high-performance computing resources.

The Organization

To conduct this study, we partnered with an engineering company (hereafter referred to as "Company X") that supplies fuel injection systems for commercial and military applications. The partnership was facilitated by a previous relationship we had with one of their customers. A medium-sized business, Company X employs about 200 people, with three computational engineers on staff. These computational engineers take the CAD models from the design engineers, construct finite element models, and then use computational tools to predict whether the part will fail under various conditions it may be subjected to during operation. Before beginning the study, Company X had no previous plans to pursue HPC.

Their primary software tool is ANSYS Mechanical, a commercial finite-element analysis solver that can simulate structural and thermal stresses. For more sophisticated simulations, the engineers also use CD-adapco's STAR-CD, a computational fluid dynamics solver. Using STAR-CD can increase the accuracy of the simulation outputs under certain scenarios

(e.g., to obtain more accurate estimates of heat flows due to convection).

Cloud Computing Resources

To identify a suitable cloud computing provider, we had two considerations: performance, and an environment that complies with the U.S. International Traffic in Arms Regulations (ITAR). Most modern HPC systems are *clusters*: independent computers (or *nodes*) that are connected together via a network. In that sense, any cloud computing resource has the ability to provide users with a cluster by requesting several *nodes* on demand. However, for many model-based simulation problems, the speed of the network interconnect is a critical part of the performance of software that will run in parallel on the system. Because of the need for a high-speed interconnect, it is unclear how well performance will be on cloud computing providers such as Amazon's Elastic Compute Cloud (Amazon, 2009), because there is no guarantee that multiple nodes requested will be on the same local network.

ITAR is a set of United States regulations, enforced by the Department of State, which restricts certain U.S. exports to protect national security. Computer models of designs for military vehicles typically fall under ITAR, which restricts who can access data. In particular, only those the State Department classifies as "U.S. persons" are permitted to handle such data. This severely restricts the use of cloud computing resources for processing of ITAR data, because such organizations would have to certify that no "foreign persons" have electronic or physical access to the data. Since Company X manufactured components for both commercial and military vehicles, support for ITAR processing was an important attribute for selecting a resource provider.

After examining several cloud computing providers, we chose IBM's Computing on Demand (CoD) service because they were the only provider able to provide us with both a high-speed interconnect and an ITAR-compliant environment. We rented a 14-node dual-processor, dual-core, Intel-based Linux cluster, connected via a high-performance Infiniband network between the nodes.

DEMONSTRATION PROBLEM

The computational engineers selected one of their fuel nozzle designs to serve as the driver problem for the feasibility study. The finite element model for the nozzle study has approximately six million degrees of freedom (MDOF), which is a relatively large model for Company X. The corresponding ANSYS database file takes up about 1.7GB of space on the file system. They chose this particular design because it was one they happen to be working on at the time of the study, and because the structure was more complex than their typical models, which made it a good candidate for doing the simulation on an HPC system.

Transient Thermal Structural Response

The main simulation of interest was a transient thermal structural response problem, also known as a thermal shock problem, and is shown in Figure 2. As the nozzle operates, fluids at changing temperatures move through it. These moving fluids result in convective heat transfer that causes changes in temperature in the nozzle. The purpose of the simulation is to estimate the structural stresses that the fuel nozzle undergoes over an interval of time because of these temperature changes. Using information from the customer, the computational engineer estimates the thermal surface loads at several time points within the time interval of interest, which serve as boundary conditions for the simulation.

The complete simulation is a two-stage process. The first stage is a transient thermal simulation, where the ANSYS software calculates how the temperature distribution of the nozzle varies over time given the boundary conditions. The second stage is a static structural simulation, where the ANSYS software calculates the structural stresses in the nozzle at a particular point in time using the tempera-

ture distribution at that time point from the thermal analysis. The engineers chose 81 time points in the interval for doing structural analyses. Note that the structural analysis at each time point is independent: once the temperature distribution is known, no other information from previous time points is required.

The workflow for this problem is shown in Figure 3. Each time ANSYS is invoked, it takes as input a database file containing a finite element model (in this case, nozzle.db for all invocations), and a command file that specifies what analyses are to be done. By convention, we put the word "driver" in the file name for these command files. In addition to the database file and command file, there may be other inputs depending on the problem.

In depicting the thermal analysis, we have separated it out into two stages to show the flow of the various data files. In practice, the bc-driver.txt and thermal-driver.txt file can be concatenated and this can be run as a single ANSYS invocation.

The thermal boundary conditions files contain information about the expected thermal loads of the fuel nozzle over time, based on information from the customer. For this simulation, the engineer selected 40 time points over the interval of interest for doing the transient thermal analysis. Within the model, the engineer assigns variable names to areas of interest (e.g., ZONE1), and the boundary condition files specify temperature and heat loading conditions to these regions. Before ANSYS can run this simulation, it must translate those constraints from named regions to constraints on the individual elements within the model. While the thermal boundary condition files are quite small (40-70 LOC, or 1.1KB), the corresponding load files are much larger (170 – 300 KLOC, or 135 MB). Most of the potential advantage of HPC comes from the structural analyses, which can be executed independently. Generating the load step files can also be done in parallel, although this step is not as time consuming as the structural analysis. Since ANSYS has HPC support, there is also potential benefit of running the thermal analysis on an HPC system. In addition, it is also possible to generate the load step files in parallel.

Dynamic Stress Analysis

The engineers also selected a dynamic stress analysis scenario as a challenge problem for the study. This is a frequency analysis of the nozzle at operational temperatures, also known as a model analysis. The engineer specifies a frequency range, and the ANSYS software subjects the model to structural vibrations at frequencies within the range. In such analyses, the engineers wish to verify that the natural frequencies, or modes, of the system, do not fall within the expected vibrations that the nozzle will be subjected to during its operation. These types of simulations are extremely computational intensive, and Company X would not typically attempt this type of simulation using such a large model.

As shown in Figure 4, the workflow of the dynamic stress analysis is much simpler: a single database file and command file, generating a single results file.

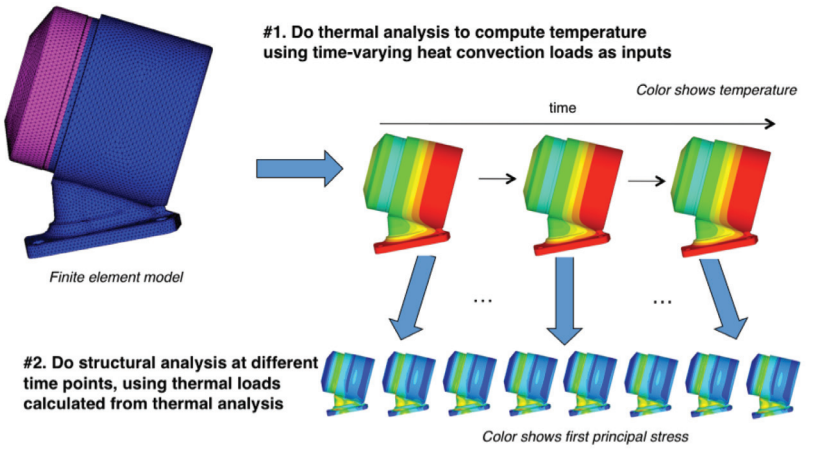
Challenges of Running in a Cloud Environment

The previous section demonstrates the potential benefits of running simulations on HPC systems. In this section, we discuss the challenges related to running HPC simulations within a cloud computing environment.

Licensing Issues

Most users have encountered a software license agreement, typically in the form of an end-user license agreement: a series of legalistic text that describes how we are permitted to use the software. As end-users, most of us simply click the "Accept" button without a passing glance. However, in corporate environments, software license agreements may be treated just like any other contract, requiring approval and possible modification of license agreements by legal departments.

Figure 2. Transient thermal structural response



When we began this study, we were unprepared for the complexity of dealing with license agreements with respect to the ANSYS software. In our case, the cost was not an issue, as ANSYS provided us access to the software without charge for this project. But they had never dealt with the complexity of the situation where the license holder (University of Southern California) was different from the intended users (engineers at company X), which was also different from the site where the software would be installed (IBM Computing on Demand).

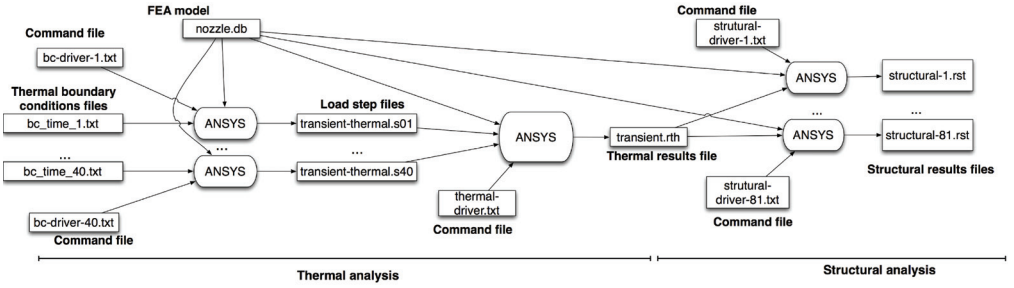
Complicating issues even further, the legal department at the University of Southern California desired some modifications to the language of the license agreement. Resolving the license agreement issues to the satisfaction of legal departments at all four organizations (ANSYS, USC, Company X, IBM) took months of document exchanges. Table 1 shows an abbreviated timeline of the events involved in obtaining access to the IBM Computing on Demand resources and the ANSYS software and licenses. Note the span of time between when ANSYS agreed to donate licenses for this project to the time that software was obtained (five months), and the time between establishing what IBM computing services are requested and when the services became available (four months).

Software and Hardware Configuration Issues

IBM's Computing on Demand uses the Infrastructure-as-a-Service (IaaS) model; IBM initializes the nodes with a supported Linux distribution of the user's choice, and then provides the user with root access. It is then up to the user to install and configure the applications they wish to run on the cloud. In our case, that involved installing ANSYS Mechanical, as well as the TORQUE open-source resource manager for launching parallel jobs on the node.

The fourteen IBM compute nodes we rented had both gigabit Ethernet and Infiniband network interfaces. ANSYS can be run in HPC mode to take advantage of multiple processors, implementing parallelism through the use of the Message Passing Interface (MPI) library. The term "MPI" does not refer to a specific library, but rather a standardized library interface (Dongarra et al., 1996) for FORTRAN, C and C++, for which there are multiple implementations. ANSYS ships with two MPI implementations, one by HP (the default) and another by Intel. For ANSYS to take advantage of the high-speed Infiniband interconnect, it requires Infiniband support in both the operating system and the MPI implementation. We initially wanted to install CentOS Linux 5 as the operating sys-

Figure 3. Workflow for transient thermal structural response



tem, but the drivers were not compatible with the Infiniband interfaces, so we were forced to install CentOS Linux 4, instead. According to the documentation, both the HP and Intel MPI implementations had support for Infiniband. However, in practice, the HP-MPI library was not compatible with the Infiniband interconnects installed on our nodes, which caused ANSYS to crash each time we attempted to run it over Infiniband. Switching to the Intel MPI library eventually resolved the problem.

When we had the software configured and began doing ANSYS runs, we soon discovered errors due to the hard drives on the compute nodes filling up. The default configuration had only 67 GB of space, which was inadequate to store the input files, output files, and intermediate files from our ANSYS runs. We upgraded the storage on the local nodes to support the simulations.

Network and VPN Configuration Issues

To access remote resources in the cloud, the computational engineers must be able to log into the remote machines from inside Company X. However, the company's network configuration was not designed with this type of remote access in mind. In general, the IT personnel work to secure the network to prevent unauthorized machines from joining the network.

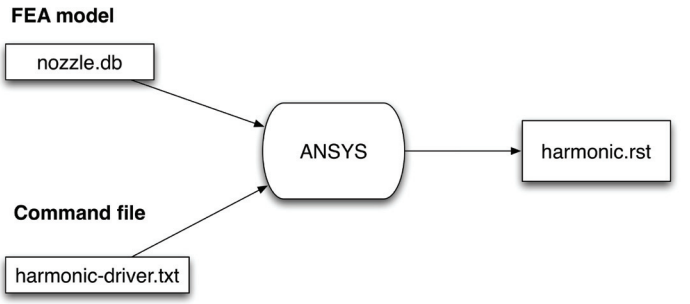
Company X's network only allows outbound connections to the web, and only through a web proxy. The IT department had to change the firewall rules to allow outbound SSH connections from certain machines so that computational engineers could log into a test cluster at ISI.

To access IBM Computing on Demand resources, users connect through a virtual private network (VPN). We had installed the ANSYS license server on the head node of the test cluster at ISI, so that we could have an HPC environment for running ANSYS applications when not renting time on IBM. Each node in a cluster that runs an instance of ANSYS needs to be able to connect to an ANSYS license server and check out a token. The implication of this was that the nodes in IBM's cluster had to be able to access the head node of the ISI cluster. Therefore, we had to maintain a VPN connection between the ISI head node and IBM whenever running ANSYS.

Ideally, the VPN connection should be maintained indefinitely. Each time the head node initiated a new connection to the VPN, it was assigned a new IP address on the VPN, which necessitates altering a configuration file on the IBM nodes to point to the right location for the license server.

IBM provides a range of VPN connectivity options depending on the number of concurrent users required and the maximum throughput

Figure 4. Workflow for dynamic stress analysis



needed for the application. These options included both "hardware VPNs" (physical machines that are connected to the network) and "software VPNs" (software clients that run on a machine with Internet access). In our circumstance, the overhead of configuring a hardware VPN for a network seemed too high, so we chose a software VPN solution.

IBM offered two software VPN solutions: a software VPN client that works over UDP, and an "SSL VPN" that works over an SSL tunnel, which uses TCP. We began with the UDP-based software VPN client, since a UDP-based VPN tunnel should have better performance. The commercial Linux version of the VPN client that we obtained did not function properly on the license server machine. Fortunately, an open-source implementation of the VPN client was available, and worked properly. However, we discovered that the VPN connection would automatically timeout after about 24 hours. This 24-hour timeout was hard-coded into the VPN server implementation.

We opted instead to switch to an SSL VPN solution. This tunnels the VPN over an SSL connection. This had a lower effective throughput than the UDP-based VPN, but it did not have the timeout issues. In addition, because the network bottleneck was not based on the VPN, the lower performance of the SSL VPN was less of an issue.

We did run into several troubles with the SSL VPN. In particular, the supplied VPN clients did not work properly on Linux. Fortunately, an open-source implementation of a Linux client was available that was compatible with the VPN (Knight, 2007).

Network Performance

The goal of using a cloud computing environment is to save compute time. However, there is overhead associated with transferring data to and from the cloud.

Figure 5 shows the network configuration. As mentioned earlier a customer has a choice of different VPN solutions depending on the network throughput and number of concurrent users required. However, in our case, the performance bottleneck of the network was on Company X's network. The computational engineers were located at a branch office of Company X, and all network traffic bound for the Internet had to be routed through the corporate office, over a T1 line that was shared by the entire branch office.

Our network transfer tests revealed an effective throughput of about 130 KB/s from Company X to IBM CoD. Table 2 shows estimates of the file transfer times. Note that a structural analysis consists of 81 independent structural analyses and therefore generates 81 structural results files, each of which is approximately the same size.

Recall that we saved about 90 hours (about four days) by running the transient analysis in the cloud environment. While the time to upload the input data is reasonable (about four hours), the download times for the output data are clearly prohibitive. At these transfer times, this

time saving is swamped by the amount of time required for downloading all of the data (587.9 h), which is roughly three and a half weeks.

The result was that we needed to identify an alternate way for the engineers to be able to access the simulation data for *post-processing*. Typically, when a simulation is complete, the engineers visualize the simulation results through ANSYS, using false color to view the distributions of different types of analysis, such as temperature and various different types of stress measures. This is a highly interactive process, as the engineer manipulates a 3D model, selecting and zooming in on various areas.

We evaluated the feasibility of doing interactive remote post-processing. Since ANSYS on Linux is an X11 application, in principle it can be run remotely. In practice, the bandwidth of the network connection was simply too low to allow the computational engineers to view the simulation results over the network. We evaluated two remote visualization solutions that were designed to provide better performance than X11.

NoMachine's NX accelerates remote X11 applications by using various strategies to compress the X protocol, reducing the amount of round-trip network interactions (Pinzari, 2003). While we did observe better performance using NX, the response time was still much too slow to be usable by the engineers.

One of the drawbacks of NX is that it cannot take advantage of the 3D acceleration provided by modern video cards, and post-processing involves visualization and manipulation of a 3D model. To evaluate whether 3D acceleration would improve performance, we evaluated VirtualGL (Commander, 2009a) to accelerate remote visualization. VirtualGL takes advantage of graphics accelerator hardware located on the remote server. It is designed to be used in conjunction with other remote visualization tools. We evaluated it in conjunction with NX, as well as with TurboVNC (Commander, 2009b), a remote visualization tool that implements the Virtual Network Computing protocol (Richardson, 2009). The compute nodes in our cluster at IBM did not contain any graphics accelerator hardware. Fortunately, IBM also maintains a visualization node that is intended specifically for supporting remote visualization and contains graphics accelerators. To evaluate the VirtualGL solution, we obtained access to the visualization node. The responsiveness of the remote application was too poor for it to be usable. Our final solution was to implement a web-based post-processing solution. Another potential solution would be to simply ship an external hard disk via overnight delivery from IBM to Company X.

WEB-BASED REMOTE POST-PROCESSING

To support the engineers in doing post-processing of the data, we had to provide them with information about the results without requiring them to download hundreds of gigabytes of data. Networking issues aside, we also had to provide them with some way of dealing with this large volume of data. In their traditional workflow, the engineers would work sequentially, running a single simulation, examining the results, and then running another one. In this case, we had given them the results from 81 different simulations, and they had no way of dealing with all of this data other than to examine each file sequentially.

We developed the HPC Remote Simulation Portal (RemoteSimPortal) to support remote, parallel post-processing. RemoteSimPortal is a web-based front-end for ANSYS that allows the engineers to explore the results from many simulations at once. Figure 6 shows a screenshot of the initial view of RemoteSimPortal. This view shows the results of the model for several different types of analyses, at several different viewpoints.

Clicking on one of these images will provide a thumbnail view. Assuming the results contain multiple time points (in the case of a transient analysis) or multiple frequency points (in the case of a harmonic analysis), all of the different points will be visible. The engineer can scan through to identify points of interest, and then click on one for more detail.

Table 1. License negotiation abbreviated timeline

Date	Service	Event
7/15/08	IBM	Initial telecon with IBM to discuss service
7/18/08	IBM	IBM sends USC preliminary service agreement
7/24/08	IBM	IBM sends USC pricing proposal
7/29/08	ANSYS	**ANSYS agrees to donate license**
8/6/08	IBM	**Telecon with IBM to establish service requested**
8/13/08	IBM	Updated pricing info from IBM
9/4/08	IBM	IBM sends USC service agreement
9/9/08	IBM	USC legal requests changes
9/23/08	ANSYS	ANSYS sends license agreement
10/13/08	ANSYS	USC legal requests changes
10/16/08	IBM	IBM responds to USC legal changes
10/21/08	IBM	USC legal requests changes
10/22/08	IBM	IBM legal responds to USC
10/23/08	IBM	USC legal responds to IBM
11/18/08	IBM	USC signs IBM agreement
11/19/08	ANSYS	ANSYS legal responds to USC
11/19/08	IBM	USC requests billing info from IBM
12/1/08	ANSYS	IBM comments on ANSYS letter
12/4/08	IBM	IBM signs agreement, provides VPN access
12/10/08	IBM	IBM sends machine login information
12/17/08	IBM	**IBM nodes become accessible**
12/17/08	ANSYS	ANSYS responds to IBM
12/19/08	ANSYS	USC signs ANSYS agreement
12/26/08	ANSYS	**ANSYS software obtained**

The engineer may also select a new viewing angle, as well as selecting a subset of nodes to focus on a region of interest. The engineer brings up the input model in ANSYS, chooses the desired view with the ANSYS interface, and selects the desired nodes. Next, the engineer opens up the ANSYS session editor, which keeps a log of all of the user interface commands, and copies and pastes them into a field in a web form and clicks the "Generate" button. The result is a visualization of the subset of nodes, as well as a text listing of the numerical values of the selected nodes, which can then be copied and pasted into a text file for later analysis with a different program such as Excel. In this way, RemoteSimPortal provides some interactivity without the need for large downloads.

PERFORMANCE ANALYSIS

In this section, we compare the time to execute these simulations on IBM's Computing on Demand resources, as compared to the time to execute the simulations on Company X's desktop. Table 3 shows a summary of the computing hardware used for the comparisons.

Transient Thermal Structural Response

Recall that the main simulation was a transient analysis that involved a thermal and structural analysis over an interval of time (Table 4). In the first phase, a thermal analysis is performed that models the heat flows through the nozzle over the time interval. This analysis will determine how the temperature distribution across the nozzle changes over the time. In the second phase, several time points on the interval are selected and structural analyses are performed using the temperature distribution. These structural analyses can be executed independently. For this simulation, we ran at 81 different time points.

Company X would not attempt to do such a simulation on the desktop, because of the amount of time it would take (roughly four days of computation).

Note that we did not separate out the time to generate the load files from the time to do the thermal simulation, because we did not have this information separated out in the desktop-based simulations done at Company X. On our cluster, generating the load step files in parallel reduced the load step generation time from 86 minutes to about 9 minutes.

Also note that the structural analysis can potentially scale linearly with the number of cores in an HPC system, because the analyses are all independent. The cluster we used had 56 cores, so in principle we should have been able to reduce the structural execution time even further. Unfortunately, as we only had 10 ANSYS licenses, we could only run 10 concurrent simulations.

Dynamic Stress Analysis

A harmonic analysis is used to determine how the nozzle will respond to vibration. A physical system such as a fuel nozzle has a set of normal modes, or resonant frequencies. It is important to confirm that the operational behavior of the nozzle does not cause it to be driven at one of its resonant frequencies, or structural failure is more likely. Harmonic analysis is a very compute-intensive task, and Company X would not typically attempt such a simulation on such a large model because of the execution time involved. Table 5 shows the execution time results. The engineers at Company X were not able to run the model to completion on a desktop: even after 4 weeks of execution, the simulation had not converged to a solution. By contrast, executing in the IBM CoD environment, the model completed executing in 88 minutes. Including data transfer time (219 minutes, as in the transient analysis) brings it to a total of about five hours.

Advantages of a High-Speed Interconnect

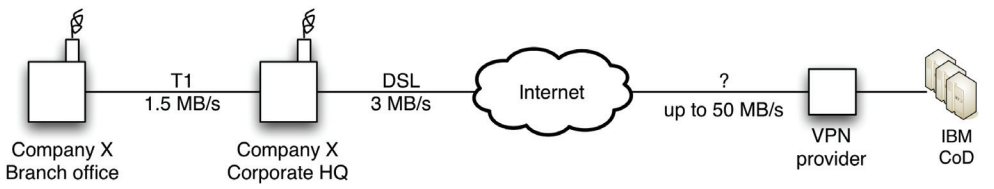
Earlier, we mentioned the advantage of having a high-speed interconnect for the cluster for engineering simulation problems. We ran the thermal analysis on different numbers of cores on the IBM cluster, as well as a test cluster that we had at ISI. On our ISI cluster, we had gigabit Ethernet network interfaces, which have lower bandwidth and higher latency than Infiniband. Figure 7 shows the execution times of the thermal analysis for different scenarios. We can see that we get reductions in execution time when running up to 16 cores on the ISI cluster, after which the performance degrades. By contrast, the IBM performance improves up to the maximum 56 cores, although performance improvements are moderate.

ESTIMATED Economic Benefits

The scope of the study was too small to directly measure economic benefits. Instead, the computational engineers estimated the potential benefits of access to larger computational resources. We considered three classes of benefits, in order of increasing payoff but also increasing uncertainty in estimates:

- Engineering productivity increase due to more efficient workflow
- Risk reduction by reducing probability of failures
- Wider adoption of HPC within the organization (manufacturing & quality)

Figure 5. Network configuration



Engineering Productivity Increase

The first estimate assumed a productivity increase by doing the same amount of work in less time. Engineers at Company X worked with one of their customers, a large aerospace company, to estimate the amount of time they would save through the use of value stream maps (Rother et al., 1999). They estimated the time savings of using HPC for their current finite element analysis, which typically involves 8 iterations, at about 43%. In addition, they estimated the time savings by applying parametric tools to set up multi-node cases and automate the post processing of results to be about 76%.

Risk Reduction

One of the primary goals of the computational engineers is to detect and prevent "design escapes": defects in the design of the nozzle that lead to problems when the nozzle is in operation. In mechanical engineering, as in software engineering, defects become exponentially more expensive to repair the later in the life cycle they are detected (Boehm, 1981). The aim of performing these simulations is to identify design defects earlier in the life cycle so that they are less expensive to fix.

Because the computational engineers have a fixed amount of time that they can devote to analysis, they must make simplifications to their models when running on the desktop so that the analysis will complete in the time available. One of the potential advantages of HPC is the ability to run higher-fidelity simulations in the same amount of time, increasing the likelihood of identifying any design defects.

While we feel that risk reduction is a significant potential economic benefit, because of data confidentiality issues, we cannot present the estimates of savings due to risk reduction in this paper.

Wider Adoption: Manufacturing and Quality

This study focuses on the computational engineers who work on identifying design defects early in the lifecycle. However, many of the development costs associated with Company X's products are due to issues in the manufacturing process, rather than the design process. Thermal analysis of the heat treatment process used in manufacturing could allow Company X to optimize the process time, increasing throughput and decreasing scrap and rework rates. The engineers estimate that productivity gains of 33% could be achievable if the manufacturing process was optimized.

Because of the uncertainties associated with the manufacturing process, it is impossible to completely eliminate manufacturing defects. In the face of manufacturing variations, the engineers must make an assessment about whether a field recall is required. This requires extensive analysis to evaluate risks, and the use of higher fidelity simulations would be directly applicable in this case, with potential effort savings on the order of 80% expected.

LESSONS LEARNED

There Are Significant Potential Benefits From Running Simulations on HPC

Modern engineering software packages are now mature enough to run well in an HPC

Table 2. File transfer times

		Size	Transfer time
	Input (model) file	1.7 GB	3.8 h
Transient thermal & structural analysis	Thermal results file	46.7 GB	104.6 h
	Single structural results file	3.2 GB	7.2 h
	All (81) structural results files	259.2 GB	580.7 h
	Total	262.4 GB	587.9 h
Harmonic analysis	Harmonic results file	61.1 GB	136.9 h

Table 3. Machine characteristics

	Desktop (Company X)	Cluster (IBM CoD)
# of nodes	1	14
Processor	Intel dual-core Xeon 3.8 GHz	Intel dual-processor dual-core Xeon 3.0 Ghz
RAM	12 GB	16 GB
Interconnect	N/A	Infiniband

environment provided a high-speed network interconnect is available. Our study revealed that the ANSYS software package showed performance improvements up to 56 cores, the maximum size of the cluster we had access to, when we ran a harmonic analysis on a real problem. Therefore, access to HPC resources has the potential to provide real value to solving engineering problems.

Preparing to Run on the Cloud Will Take Longer than You Expect

It takes time to get to the point where you can run engineering simulation jobs in a cloud-computing environment. Legal issues related to contracts and software licenses can add substantial delays. The local IT department may need to make network configuration changes to allow access to external resources. Several different VPN solutions may need to be evaluated to determine which one works best with the software and network configuration. Software must be installed and configured for running parallel jobs. Doing an install on an HPC system takes longer than one on a desktop system, because of the extra configuration steps required to run in parallel. In particular, when using a high-speed interconnect like Infiniband, some additional configuration may be necessary to make sure the drivers are working and are compatible with the installed software.

Substantial IT Expertise is Required for Engineers to Leverage the Cloud Today

To assist the computational engineers, we had to install and configure ANSYS for running parallel jobs, install and configure resource management software for running many smaller, independent ANSYS jobs, and we had to write several scripts to generate the ANSYS command files and shell script files to run the simulations. In addition, we had to overcome technical issues related to network configuration for remote access to the cloud and software configuration for optimized performance within an HPC system. All of these require a level of IT knowledge that a typical computational

Figure 6. RemoteSimPortal initial view

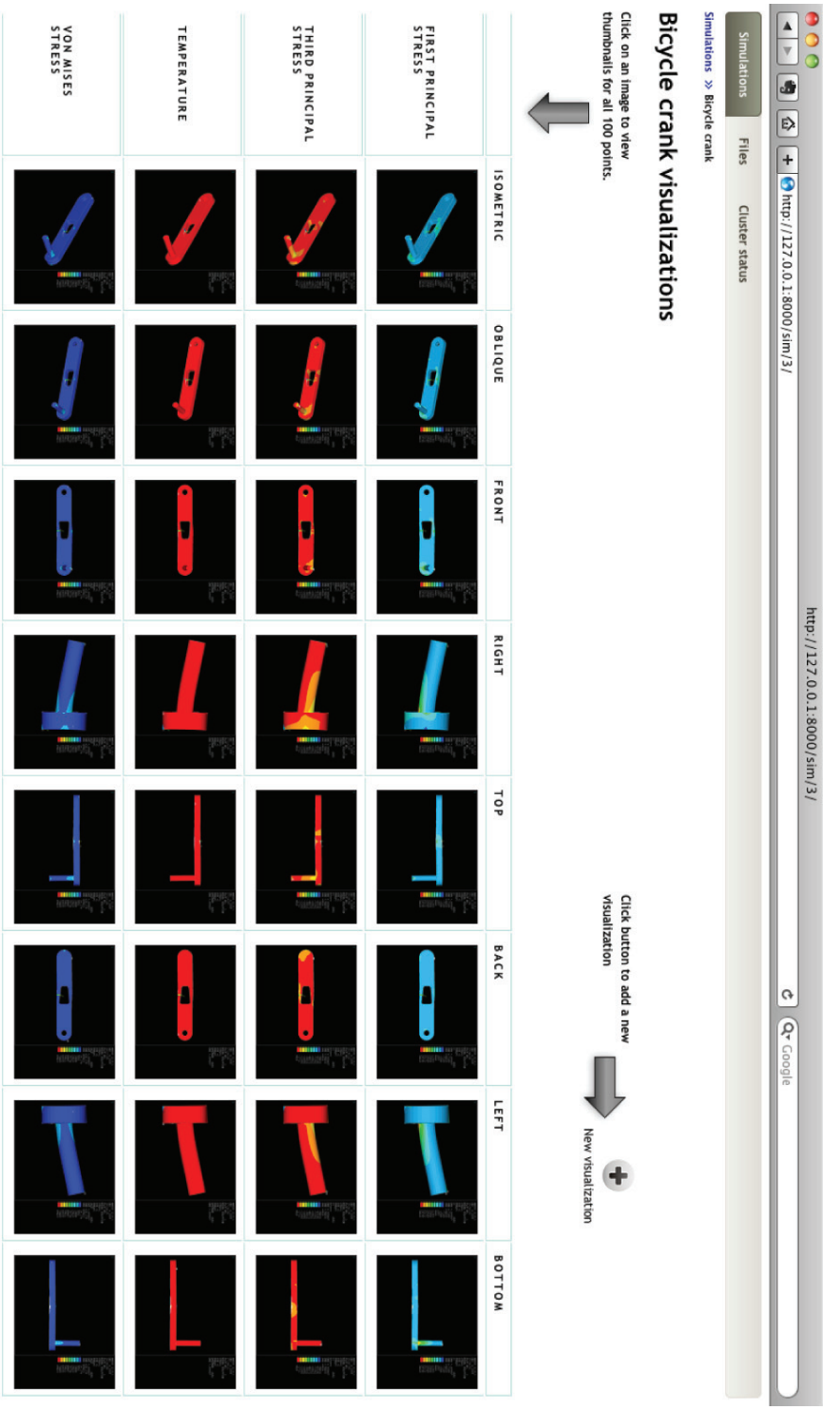


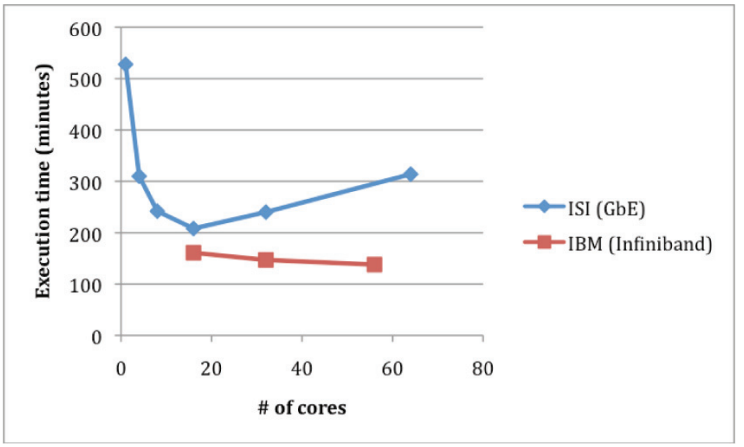
Table 4. Transient analysis simulation time

	Desktop (Company X)	Cluster (IBM CoD)
Thermal analysis	1029m (17h 9m)	148 m (2h 28m)
Structural analysis	4852m (80h 52m)	328 m (5h 28m)
Subtotal (computation time)	5881m (98h 1m)	476 m (7h 56m)
Data transfer	0	219 m (3h 39m)
Total (computation+transfer)	5881m (98h 1m)	695 m (11h 35m)
Time saved		5186 m (86h 26m)

Table 5. Harmonic analysis simulation times

	Desktop (Company X)	Cluster (IBM CoD)
Harmonic analysis	>4 weeks (did not finish)	88 m (1h 28m)
Data transfer	0	219 m (3h 39m)
Total	>4 weeks (did not finish)	307 m (5h 7m)
Time saved		> 4 weeks

Figure 7. Thermal analysis execution time vs. number of cores



engineer would not necessarily possess. It is particularly a problem if the computational engineer has experience primarily with Microsoft Windows-based environments, as all of these tools required substantial familiarity with Linux-based environments.

Poor Network Connectivity is a Major Obstacle for Doing Post-Processing

From a purely technical point of view, the networking "last mile" problem is the most fundamental obstacle to running engineering

simulation jobs on the cloud. On this project, we developed a custom web-based front-end to work around the inadequate network bandwidth required for fully interactive remote post-processing. However, most small-to-medium engineering companies probably do not have the resources to develop this type of custom software to use the cloud.

It would be advantageous for companies if they could temporarily obtain high-speed Internet access for running these simulations, but we are not aware of any such resources currently being offered commercially.

CONCLUSIONS AND FUTURE WORK

The results of our feasibility study suggests that cloud computing environments with HPC-like resources are potentially useful for performing computational engineering tasks, but substantial IT expertise is required to use them effectively today. In addition, if the engineering organization lacks a high-speed Internet connection, then alternative post-processing strategies need to be pursued.

This study focused solely on estimating the benefits of using a cloud computing environment for computational engineering. For future work, we plan to measure the associated costs to provide more complete information for estimating return-on-investment. We are also interested in exploring the execution of multiphysics simulations in cloud environments, where simulations from different software packages are combined (e.g., finite-element analysis with ANSYS Mechanical, computational fluid dynamics with CD-Adapco STAR-CD). In addition, RemoteSimPortal is still under active development, and we plan to conduct usability studies with the engineers to determine how well it can substitute for local post-processing and to tailor the interface to better fit into their workflow.

This study focused on simulations being performed in the context of a single organization. While each individual organization can potentially benefit from higher fidelity simulations, we hypothesize that significant benefits in overall engineering productivity and product quality can be best achieved through simulations that go across organizations in the supply chain. Consider the case of an automobile, where the automobile manufacturer purchases an engine from a supplier, and the engine manufacturer purchases the nozzles for the fuel injection system from a supplier. Currently, the automobile manufacturer is not able to leverage the computer-based models employed by the engine manufacturer and the fuel nozzle manufacturer, because organizations are not necessarily willing to share details of their simulations with customers or suppliers. However, if it were possible to integrate these models, then the automobile manufacturer would be able to run higher-fidelity simulations than is currently possible. Future work will explore the technical, social, and business challenges to identify how to facilitate such integrated simulations across a supply chain.

ACKNOWLEDGMENTS

This research was performed under the DARPA HPC-ISP-PILOTS project under Air Force award #FA8750-08-C-0184 to the University of Southern California. We wish to acknowledge the computational engineers who contributed substantially to the content of this paper, but who must remain anonymous. We also wish to acknowledge ANSYS, Inc., for providing us with the software licenses used in this project. We also would like to thank Jeff Carver for his feedback on an earlier draft of this paper.

REFERENCES

Amazon. (2009). *Amazon elastic compute cloud user guide* (API Version 2009-11-30). Retrieved June 22, 2010, from http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., et al. (2009). *Above the clouds: A Berkeley view of cloud computing* (Tech. Rep. No. UCB/EECS-2009-28). Berkeley, CA: University of California, Berkeley.

Basili, V. R., Cruzes, D., Carver, J. C., Hochstein, L. M., Hollingsworth, J. K., Zelkowitz, M. V., & Shull, F. (2008). Understanding the high-performance-computing community: A software engineer's perspective. *IEEE Software*, *25*(4), 29–36. doi:10.1109/MS.2008.103

Bialecki, A., Cafarella, M., Cutting, D., & O'Malley, O. (2007) *Hadoop: A framework for running applications on large clusters built of commodity hardware*. Retrieved January 14, 2010, from http://hadoop.apache.org

Boehm, B. (1981). *Software engineering economics*. Upper Saddle River, NJ: Prentice Hall.

Callan, J., & Kulkarni, A. (2009). *ClueWeb09 Wiki*. Retrieved January 14, 2010, from http://boston.lti.cs.cmu.edu/clueweb09/wiki

Carver, J. C., Kendall, R. P., Squires, S. E., & Post, D. E. (2007). Software development environments for scientific and engineering software: A series of case studies. In *Proceedings of the 29th International Conference on Software Engineering*.

Cary, A., Sun, Z., Hristidis, V., & Naphtali, R. (2009). Experiences on processing spatial data with mapreduce. In *Proceedings of the 21st International Conference on Scientific and Statistical Database Management* (pp. 302-319).

Cohen, J. (2009). Graph twiddling in a MapReduce world. *Computing in Science & Engineering*, *11*(4), 29–41. doi:10.1109/MCSE.2009.120

Commander, D. (2009a). *User's guide for VirtualGL 2.1.4*. Retrieved January 14, 2010, from http://www.virtualgl.org/vgldoc/2_1/

Commander, D. (2009b). *User's guide for TurboVNC 0.6*. Retrieved January 14, 2010, from http://comments.gmane.org/gmane.comp.video.opengl.virtualgl.user/42

Council on Competitiveness. (2009) *U.S. manufacturing – Global leadership through modeling and simulation*. Retrieved January 15, 2010, from http://www.compete.org/publications/detail/652/us-manufacturingglobal-leadership-through-modeling-and-simulation

Dean, J., & Ghemawat, S. (2004). MapReduce: Simplified data processing on large clusters. In *Proceedings of the 6th Symposium on Operating Systems Design and Implementation* (pp. 107-113).

Dongarra, J. J., Otto, S. W., Snir, M., & Walker, D. (1996). A message passing standard for MPP and workstations. *Communications of the ACM*, *39*(7), 84–90. doi:10.1145/233977.234000

Easterbrook, S. M., & Johns, T. C. (2009). Engineering the software for understanding climate change. *Computing in Science & Engineering*, *11*(6), 65–74. doi:10.1109/MCSE.2009.193

Faculty Connection. (2007). *Software entrepreneurship for students curriculum: A case study.* Retrieved from http://www.microsoft.com/education/facultyconnection/articles/articledetails.aspx?cid=844&c1=en-us&c2=0

Glotzer, S. C., Kim, S., Cummings, P. T., Deshmukh, A., Head-Gordon, M., & Karniadakis, G. (2009). *WTEF panel report on international assessment of research and development in simulation-based engineering and science*. Baltimore, MD: World Technology Evaluation Center.

Heavy, A., Lauret, J., & Keahy, K. (2009, April 8). Clouds make way for STAR to shine. *International Science Grid This Week*.

Helper, S. (2009). The high road for U.S. manufacturing. *Issues in Science and Technology*, *25*(2), 39–45.

Hochstein, L., & Basili, V. R. (2008). The ASC-alliance projects: A case study of large-scale parallel scientific code development. *IEEE Computer*, *41*(3), 50–58.

Hoffa, C., Mehta, G., Freeman, T., Deelman, E., Keahey, K., Berriman, B., & Good, J. (2008). On the use of cloud computing for scientific workflow. In *Proceedings of the 3rd International Workshop on Scientific Workflows and Business Workflow Standards in e-Science* (pp. 640-645).

Joseph, E., Snell, A., & Willard, C. G. (2004). *Council on Competitiveness study of U.S. industrial HPC users*. Washington, DC: Council on Competitiveness.

Keahey, K., Tsugawa, M., Matsunaga, A., & Fortes, J. (2009). Sky computing . *IEEE Internet Computing*, *13*(5). doi:10.1109/MIC.2009.94

Knight, J. Y. (2007). *F5 VPN command-line client*. Retrieved January 12, 2010, from http://fuhm.net/software/f5vpn-login

National Academies. (2007). *Rising above the gathering storm: Energizing and employing America for a brighter economic future*. Washington, DC: National Academies Press.

Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., & Zagorodnov, D. (2009). Eucalyptus: An open-source cloud computing infrastructure. *Journal of Physics: Conference Series, 180*.

Pinzari, G. F. (2003). *NX X protocol compression* (D-309/3-NXP-DOC). Retrieved January 14, 2010, from http://www.nomachine.com/documents/html/NX-XProtocolCompression.html

Post, D. E. (2009). The promise of science-based computational engineering. *Computing in Science & Engineering*, *11*(3), 3–4. doi:10.1109/MCSE.2009.60

Post, D. E., & Votta, L. G. (2005). Computational science demands a new paradigm. *Physics Today*, *58*(1), 35–41. doi:10.1063/1.1881898

Quom, S., & Bhaskaran, R. (2002). *ANSYS short course: Three-dimensional bicycle crank*. Retrieved January 14, 2010, from http://courses.cit.cornell.edu/ansys/crank

Richardson, T. (2009). *The RFB protocol, version 3.8*. Cambridge, UK: RealVNC Ltd.

Segal, J. (2005). When software engineers met research scientists: A case study. *Empirical Software Engineering*, *10*(4), 517–536. doi:10.1007/s10664-005-3865-y

Shainer, G., Sparks, B., Schultz, S., Lantz, E., Liu, W., Liu, T., & Misra, G. (2010, January 26). Cloud computing will usher in a new era of science discover. *HPCwire.*

Sotomayor, B., Montero, R. S., Llorente, I. M., & Foster, I. (2008). Capacity leasing in cloud systems using the OpenNebula engine. In *Proceedings of the Cloud Computing and Applications Conference*.

Wang, L., Tao, J., Kunze, M., Castellanos, A. C., Kramer, D., & Karl, W. (2008) Scientific cloud computing: Early definition and experience. In *Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications* (pp. 825-830).

Zhao, B. Y., Yan, K., Agrawal, D., & El Abbadi, A. (2009). *Massive graphics in clusters (MAGIC) project*. Retrieved January 14, 2010, from http://graphs.cs.ucsb.edu

Lorin Hochstein is a computer scientist at the University of Southern California's Information Sciences Institute (ISI). His current research focuses on improving the productivity of computational science and engineering software developers and end-users, in particular through the application of high-performance computing and cloud computing. Prior to ISI, he was an assistant professor in the Computer Science & Engineering Department at the University of Nebraska-Lincoln. He received a PhD in computer science from the University of Maryland, an M.S. in electrical engineering from Boston University, and a BEng. in computer engineering from McGill University.

Brian Schott is Chief Technology Officer of Nimbis Services. He has over 15 years experience in the development of high performance computing systems, embedded computing systems, and computer networks. Prior to Nimbis Services, he was a Project Leader at the University of Southern California's Information Sciences Institute (ISI). Since starting at ISI in 1997, Mr. Schott has managed DARPA, NASA, and DoD-funded research projects involving large multidisciplinary teams of researchers from leading academic institutions, national laboratories, and industry. Mr. Schott has two patents. He received an MS in computer science from George Washington University and a BS in computer science from the University of Maryland.

Robert B. Graybill is CEO and President of Nimbis Services. He has more than 35 years of HPC-related senior-level experience as a business leader, government program manager and technology researcher. Prior to founding Nimbis, he was Director of National Innovation Initiatives at the University of Southern California's Information Sciences Institute (ISI), where he fostered development of advanced national high performance computing (HPC) collaborative environments to help companies, universities and national laboratories share high performance computing systems and computational science expertise. Before joining ISI, he spent six years at DARPA, where he designed, developed and implemented six new transformational programs in high-end computing architectures and responsive embedded computing hardware, software and network systems. He was a member of the Senior Science Team for government HPC studies conducted by the Defense Science Board task force on DoD Supercomputing Needs and the High-End Computing Revitalization Task Force. He has an MS in computer science from Johns Hopkins University and a BS in electrical engineering from Pennsylvania State University.