Grid computing for energy exploration

Dimitri Bevc[1], Sergio E. Zarantonello[2], Neena Kaushik[3], Iulian Musat[4]

Abstract

3-D seismic imaging is the most computationally intensive task in the oil and gas industry. It is a key technology that has allowed the success ratio of exploratory wells to increase from 20% to 80%. The objective of this paper is to give an overview of a Grid-enabled environment for seismic imaging developed by 3DGeo and comment on its use in a production setting. This system addresses the demand for advanced seismic imaging applications in the oil and gas industry, and the ensuing need of computational and data resources to run these applications flexibly and efficiently.

1. Introduction

A key task in exploration geophysics is the creation of images of the subsurface of the earth to identify deposits of oil and gas. The earth's response to trains of artificially created elastic waves is recorded with arrays of geophones. This data is then processed to ultimately deliver images of the subsurface. The construction of accurate 3-D images of the subsurface from this data is an extremely resource-intensive task. It requires the handling of large data volumes - on the order of 10-15 Terabytes for a single modern marine 3-D survey - and the application of computationally-demanding imaging algorithms. Only large-scale parallel computers can apply these algorithms and deliver the results within a useful turn-around time. Furthermore, the most advanced imaging technologies are resource-demanding and only feasible today on small-sized projects. Harnessing Grid resources flexibly and effectively is thus a fundamentally important development for the oil and gas industry.

In this paper we describe our work developing a Grid-enabled system for seismic imaging.

3DGeo is a leading-edge provider of advanced software products and services to the oil and gas industry[5] . Its commercial offerings include INSP (Bevc and Popovici, 2003), a proprietary Java based Internet infrastructure for remote collaborative seismic processing

[1] 3DGeo Development Inc., 4633 Old Ironsides Drive, Santa Clara, CA 95054 (**dimitri@3dgeo.com**).
[2] 3DGeo Development, Inc., 4633 Old Ironsides Drive, Santa Clara, CA 95054 (**sergio@3dgeo.com**), and Department of Applied Mathematics, Santa Clara University, 500 El Camino Real, Santa Clara, CA 95053 (**szarantonello@scu.edu**).
[3] 3DGeo Development, Inc., 4633 Old Ironsides Drive, Santa Clara, CA 95054 (**neena@3dgeo.com**), and Department of Computer Engineering Santa Clara University, 500 El Camino Real, Santa Clara, CA 95053 (**nrkaushik@scu.edu**).
[4] 3DGeo Development Inc., 17171 Park Row, Houston, TX 77084 (**iulian@3dgeo.com**).
[5] 3DGeo Development, Inc., **http://www.3dgeo.com/**

and seismic imaging, and a suite of advanced imaging applications that can be accessed, executed, and monitored with the INSP system. The conversion of INSP to a Grid-enabled system, providing flexible and secure access to advanced imaging applications and to the resources to run these applications *whenever and wherever needed,* is a strategic step for 3DGeo. We believe such a system will allow 3-D seismic data to be much more effectively used to characterize and delineate oil reservoirs and contribute to opening up exploration venues in extremely complicated geological conditions. A major oil discovery in these areas could decrease United State's dependence on imported oil and have a direct impact on the cost of energy.

The organization of the paper is as follows. In Section 2 we give an overview of 3-D depth imaging, we discuss the computational cost of different algorithms, and give an overview of Kirchhoff depth migration, a typical 3-D imaging application. In Section 3 we discuss design issues associated to PSDM, a particular implementation of Kirchhoff migration, in a Grid environment. In Section 4 we give an overview of the INSP seismic processing system and the specifics of the Grid-enabled extension. In Section 5 we discuss the implications of seismic imaging on the Grid and give our conclusions.

2. Seismic imaging

Seismic imaging methods solve an inverse problem for the classical wave equation of mathematical physics, whereby signals recorded by geophones on the surface of the Earth are converted to a model of the subsurface. 3-D depth imaging methods are the most computationally-intensive tasks in the oil and gas industry. These methods are usually classified into two categories: methods based on the Kirchhoff integral equation, and methods that operate directly with the wave equation. Wave-equation methods are further classified as shot-receiver (*e.g.* common azimuth and narrow azimuth methods) and shot-profile methods. Wave equation shot profile methods are the most computationally expensive methods in use, and are unfeasible except for small projects.

Table 1. The computational challenge: Gulf of Mexico 3-D marine surveys.

Size of data		***Runtime in days***		
Blocks	***Gbytes***	***Kirchhoff***	***Narrow azimuth***	***Shot profile***
10	620	3	31	184
100	6,200	111	1,100	6,640
500	30,700	996	9,960	59,800 (164 yrs !)

To put the computational challenge in perspective, in Table 1 we compare the estimated runtimes of hypothetical imaging projects for Deep Gulf of Mexico 3-D marine surveys on a 128-CPU cluster of 2.4 GHz Pentium® 4 processors delivering a sustained performance of 900 Mflops/CPU. Running shot profile migration on a large 3-D marine

survey would take 164 years to complete on such system ! The INSP system has modules for all the methods represented in Table 1.

The better accuracy offered by advanced methods is illustrated in Figure 1, where we compare the seismic images obtained with PSDM, 3DGeo's implementation of Kirchhoff pre-stack 3-D depth migration, and the image obtained with one of 3DGeo's wave-equation migration methods (Biondi and Palacharla, 1996). The generally computationally more intensive wave-equation method gives better accuracy than Kirchhoff migration, therefore underscoring the need for more compute resources deliverable through the Grid.

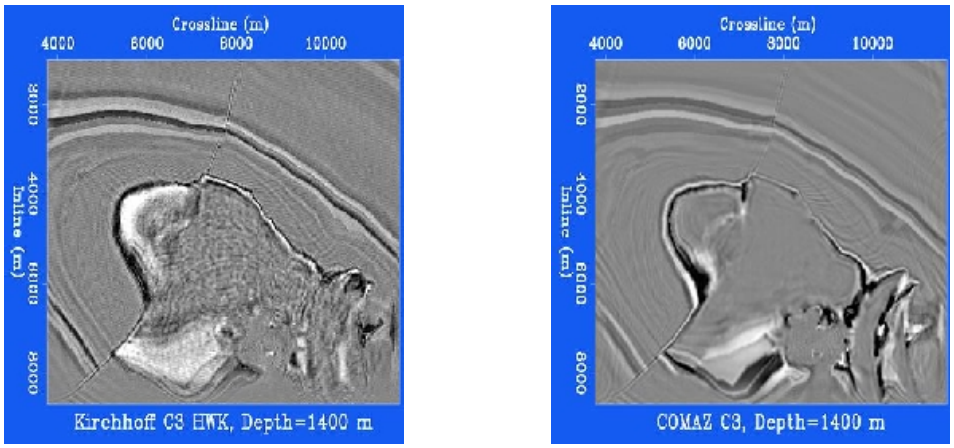


Figure 1. Image on left obtained using PSDM. Image on right obtained using a wave-equation migration method.

3. Overview of PSDM

We use PSDM as an example to illustrate design issues that were addressed for Grid deployment. Pre-Stack Depth Migration (PSDM) is 3DGeo's implementation of the three-dimensional Kirchhoff depth migration, one of the most comprehensive and flexible methods for imaging pre-stack 3-D seismic data. PSDM approximately solves the wave equation with a boundary integral method. The acoustic reflectivity at every point of the Earth's interior is computed by summing the recorded data on multidimensional surfaces; the shape of the summation surfaces and the summation weights are computed from the Green's functions of the single scattering wave propagation experiment.

The essence of 3-D prestack Kirchhoff migration can be expressed in the following integral equation:

$$\mathrm{Image}(\mathbf{x}) = \int \int_{\mathbf{x_s}} \int_{\mathbf{x_r}} \mathrm{G}(\mathbf{x_s}, \mathbf{x}, \omega)\, \mathrm{G}(\mathbf{x}, \mathbf{x_r}, \omega)\, \mathrm{Data}(\mathbf{x_s}, \mathbf{x_r}, \omega)\, \mathrm{d}\mathbf{x_r}\, \mathrm{d}\mathbf{x_s}\, \mathrm{d}\omega$$

If the Green's functions are known this solution is exact. In a computational environment we express the integral as a sum:

$$\mathrm{Image}(\mathbf{x}) = \sum_{\mathbf{x_s}} \sum_{\mathbf{x_r}} \mathbf{A_s} \mathbf{A_r}\, \mathrm{Input}(\mathbf{x_s}, \mathbf{x_r}, \mathbf{t_s} + \mathbf{t_r})$$

where A_r and A_s are determined by the transport equation, and t_r and t_s are either found by ray-tracing or by solving the eikonal equation. We note that each point **x** of the seismic image is calculated by a sum over a travel-time surface, and that the sums for the different points of the image can be calculated independently from each other. Since the wave propagation velocity is not known *a priori*, the process of building an exact image is iterative, with successive improvements made to the velocity field. Each iteration requires running a Kirchhoff migration. The overall procedure, schematized in Figure 2, involves collaboration between multidisciplinary teams and can be extremely demanding in terms of human and computational resources.

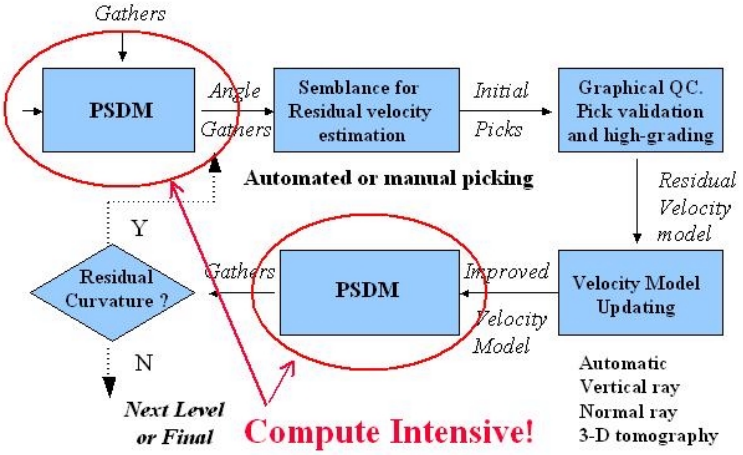


Figure 2. Building a 3-D seismic image requires iterative updating of the velocity model.

Parallelization of PSDM on a cluster. PSDM presents different parallelization issues in cluster and Grid environments. 3DGeo's implementation of Kirchhoff migration was designed to achieve maximum efficiency on a cluster of interconnected multiprocessor computers. To achieve this efficiency the input data is distributed among the computational nodes, while the output image is divided into processing blocks that are distributed over parallel processors on each node. At the end, the results from each node are gathered to build the final image.

Figure 3 illustrates the PSDM MPI architecture. The input data is distributed using a dispatching service which responds to individual requests from each node. This approach ensures a good load balancing over heterogeneous nodes, especially if the computational demands for processing each block of input data are different. This dispatching service keeps track of the execution stage and provides necessary information to restart the entire job in case of a failure, an important feature on a large distributed system where one node may become unavailable anytime during the execution.

Each node has a copy of the output image which is divided into processing blocks. The summation computation for each block is distributed independently over parallel processors. By partitioning the output, the algorithm is scalable with respect to the total amount of memory available, and can run efficiently on workstations where less memory is available or on supercomputers where any remaining memory can be used for staging data. Once the entire input data is processed, the final image is composed by summing the local images from each node.

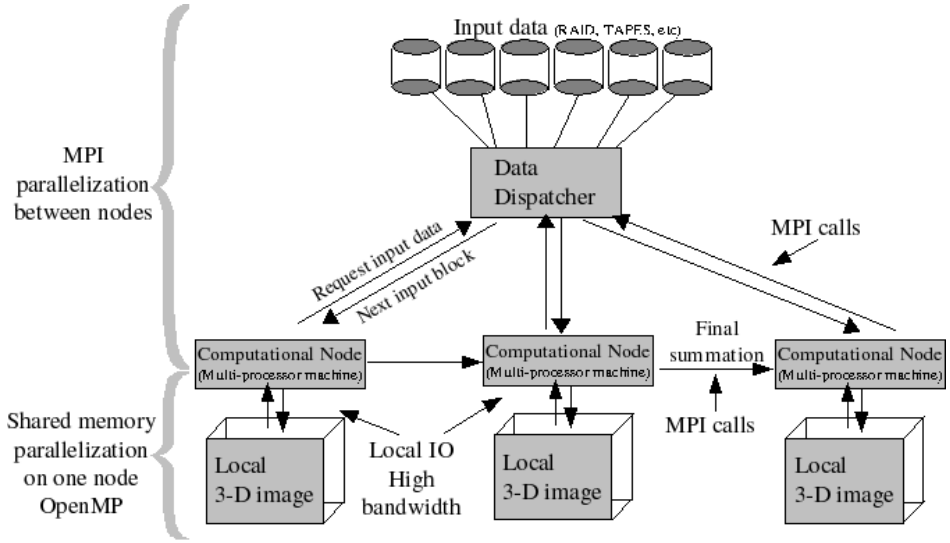


Figure 3. Parallelization of PSDM on a cluster

After comparing several multi-threading libraries, we chose the OpenMP[6] standard for the implementation of the shared-memory parallelization on each multiprocessor node. This standard is jointly defined by a group of major computer hardware and software vendors, and provides a simple and flexible interface for developing parallel applications, portable on a variety of systems.

The solution chosen to implement the input data dispatching mechanism and the collective operations used for the final summation is based on the MPI standard. We used MPICH (Groop, Doss, and Skjellum, 1996), an open source implementation of the MPI standard, optimized and tested on a variety of architectures. PSDM uses about one hundred parameters, usually extracted from a text file. Since many of those parameters refer to files on the local file system, the parameter acquisition library was augmented to accept placeholders for different MPI runtime values such as the process rank. This simplifies the laborious task of setting up a PSDM running job.

PSDM on the Grid. Tests and benchmarking of PSDM on various cluster architectures and configurations shows that in a typical run the I/O operations associated with the input data distribution account for a small fraction of the total processing time. This encouraged us to use - in a first phase - the same architecture for distributing a PSDM job across multiple clusters, interconnected in a computational grid, as shown in Figure 4. Using the Globus Toolkit (Foster and Kesselman, 2003) we set up a Grid, interconnecting two of 3DGeo's processing centers (Santa Clara, California, and Houston, Texas) and a Linux cluster at the San Diego Supercomputing Center.

We had built PSDM using MPICH libraries, and the choice of MPICH-G2 (Karonis, Toonen, and Foster, 2003) for the necessary Grid support seemed a natural decision. The result was a Grid-enabled MPI implementation built on top of MPICH and Globus API. After re-compiling with the MPICH-G2 libraries, we were able to deploy and run a PSDM job on public IP nodes. However, since MPICH-G2 does not support private IP clusters, running a single PSDM job on two or more interconnected clusters was not possible and a modified approach was required.

The main problem was due to the implementation of MPICH-G2 which required direct TCP/IP connection between the nodes of the different clusters. Since most of the clusters were configured without public IP addresses for the internal nodes, direct TCP/IP connection could not be established. For testing purposes we were able to tunnel the TCP traffic through a secure connection, setting up a VPN between two test clusters. This solution was not scalable and the settings are impractical to achieve on a commercial Grid setup where different participants have different security policies. However, the test was fruitful in assessing the limitations of the parallelization model we built for PSDM and to draw future development plans for improving it. Our present plans are to explore and use the functionality of MPICH-GP, a modification of MPICH-G2 by Kum-Rye Park. MPICH-GP includes a Network Address Translation service, and a user-level proxy scheme. MPICH-GP supports public as well as private IP clusters, and mixed combinations of public and private IP clusters (Park *et al.*, 2004).

[6] http://www.openmp.org/

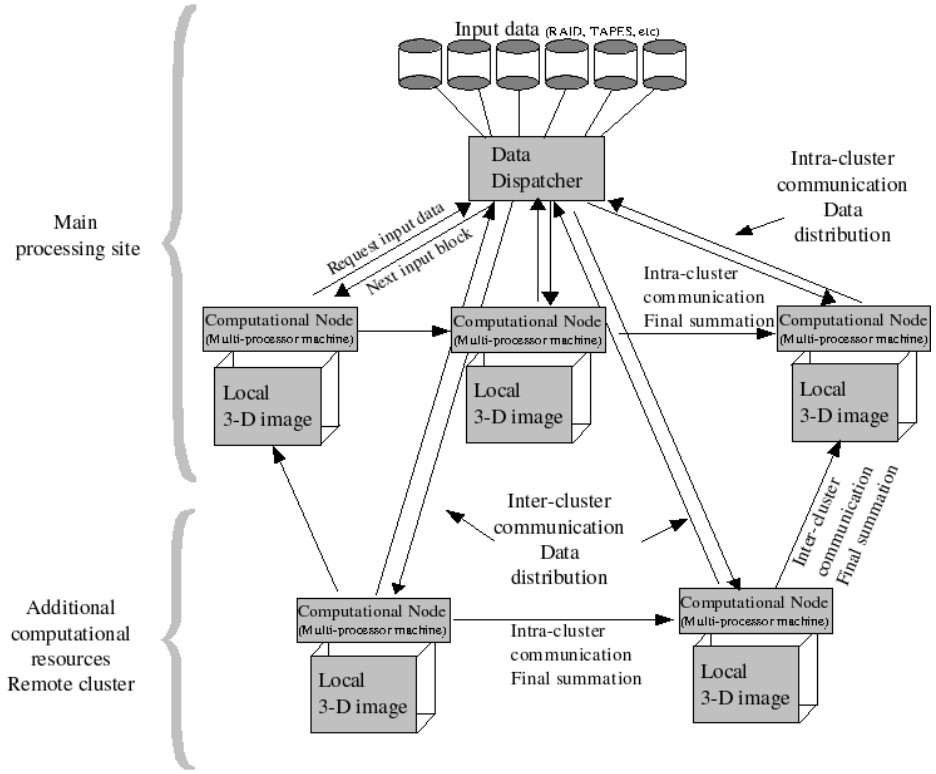


Figure 4. Parallelization of PSDM on multiple clusters.

Demonstration on the Virtual Computer. Once the SDSC cluster was configured to support the seismic imaging software, it was connected to 3DGeo's distributed monitoring Grid. The connection to the monitoring grid was performed by installing the Ganglia Resource Monitoring tool. Ganglia[7] is an Open Source application, originally developed by the Berkeley Millenium project and currently bundled with the NPACI Rock distributions. Ganglia was used to interconnect the 3DGeo Santa Clara processing center, consisting of 32 CPUs and 80 CPUs Linux clusters, with 3DGeo's Houston processing center, 5 clusters hosting another 350 CPUs Linux cluster. The SDSC cluster was the last addition to the monitoring grid. Ganglia gave an overview of resource utilization.

The information is accessible as a web page (Figure 5), and includes graphs showing the evolution in time of metrics such as machine load, memory usage, and number of processes. 3DGeo processing staff were thus able to inspect from a single web page the load of the machines geographically distributed in Santa Clara CA, San Diego CA and Houston TX.

[7] http://ganglia.sourceforge.net/

From the system administration point of view, setting up the monitoring infrastructure consisted of the installation and configuration of the Ganglia monitor daemon on all the cluster nodes, including the master nodes, and the installation of a Ganglia metadaemon, an Apache web server, and the Ganglia web front end on the master nodes. The central metadaemon running in the Santa Clara processing center was configured to interrogate and accept data from the San Diego cluster metadaemon. In addition, Gexec, a component of the Ganglia project, was experimentally installed on 3DGeo's and SDSC's clusters. Gexec is a scalable cluster remote execution system which provides RSA authenticated remote execution of parallel and distributed jobs. Its capabilities include transparent forwarding of stdin, stdout, stderr, and signals to and from remote processes, it provides local environment propagation. Gexec is designed to be robust and to scale to systems of over 1000 nodes. For our experimental grid we installed Globus server software GT 2.4 on the master nodes of all the clusters. The final step was to incorporate the Grid-enabled applications such as PSDM within the INSP framework.

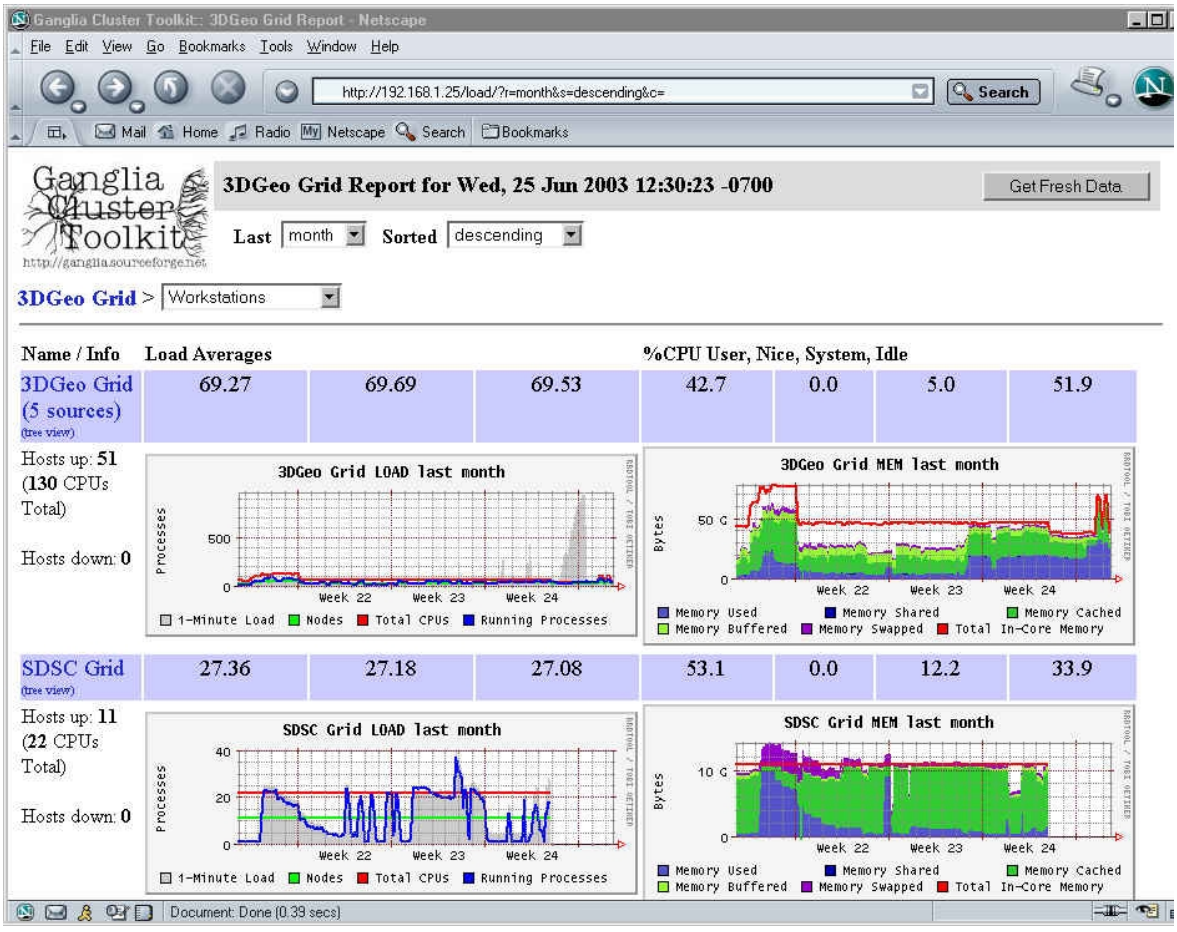


Figure 5. Status of the 3DGeo Grid displayed by Ganglia in Web browser. The windows display CPU and memory load of the available 3DGeo and SDSC resources.

4. Tying Grid resources together with INSP

INSP is a collaborative environment, developed and commercialized by 3DGeo for building and launching workflows of computationally intensive parallel and distributed jobs, visualizing data on client workstations, and monitoring jobs. It is based on a client-server relationship between user interfaces (clients) and computationally intensive workflows (servers), presenting users with options to visualize intermediate results, and monitor, and design workflows for seismic processing (Bevc, Popopvici, and Biondi, 2002). The INSP Data Viewer allows a user to visualize a data cube, to pick locations, and to edit values in the data cube, thereby developing a geological earth model. It allows remote clients to interact with multiple data objects. Figure 6 shows a screen shot of the INSP Explorer interface illustrating some of the INSP functionalities. Visible in the left side is the explorer tree structure of modules (executable application-specific programs), datasets, and workflows for each server. Seismic data, velocity models, and a sample workflow are displayed on superimposed windows.

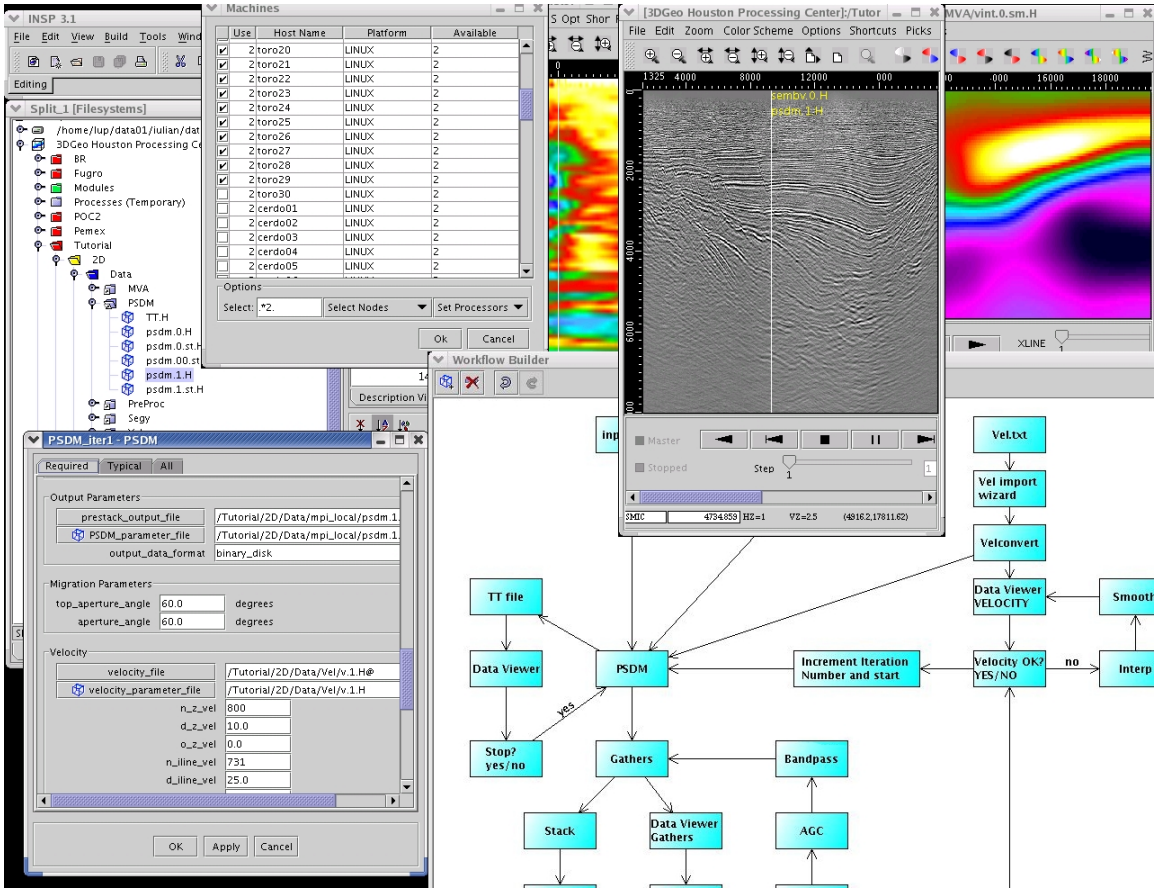


Figure 6. INSP Explorer is the Internet-based GUI for remote processing services.

The INSP graphical user interface presents the users with a complete set of seismic processing and imaging modules and a visualization system. The GUI is written in Java, allowing client portability and access to any type of computer on either a local or wide-area network. This takes advantage that Java can handle security and parallel distributed computing, all key issues in geophysical applications. The Java client-server design of INSP leverages the "write once, run anywhere" capabilities for the GUI and process management, while using highly optimized seismic imaging algorithms running on specialized high performance computers for number-crunching tasks. A key function of INSP is to facilitate the communication between members of a geographically distributed exploration team, providing them with the tools that help them easily share information, regardless of physical distance between members of the team.

Design of current INSP. The current INSP consists of a 3dGeo proprietary client and server. The client part is the GUI which shows the data, flows and the modules. It also allows the user to mount remote file systems, view data, and add new workflows. The modules are populated in the database on the server side using the "inspac" command. They are viewable on the client GUI when connected to the INSP server. The INSP architecture is shown below.

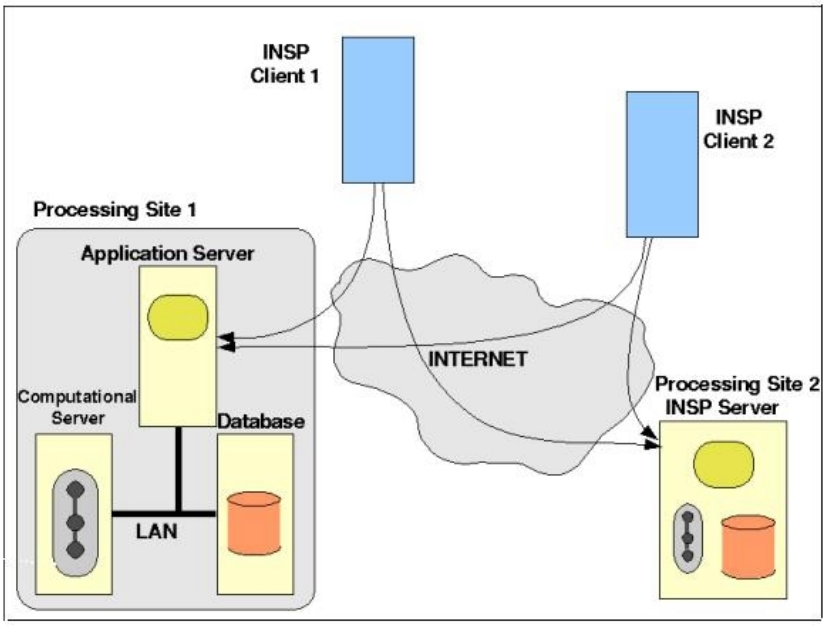
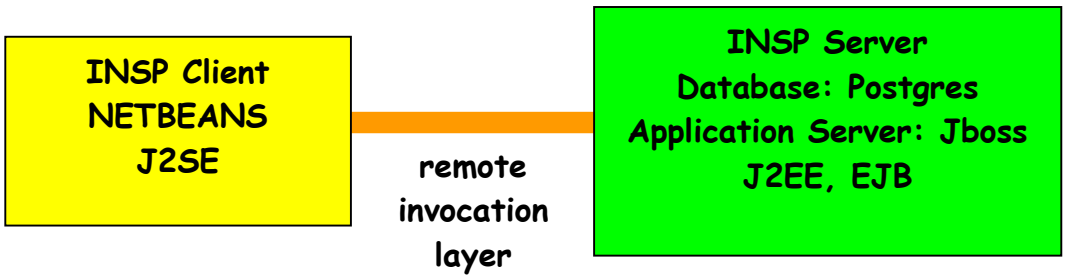
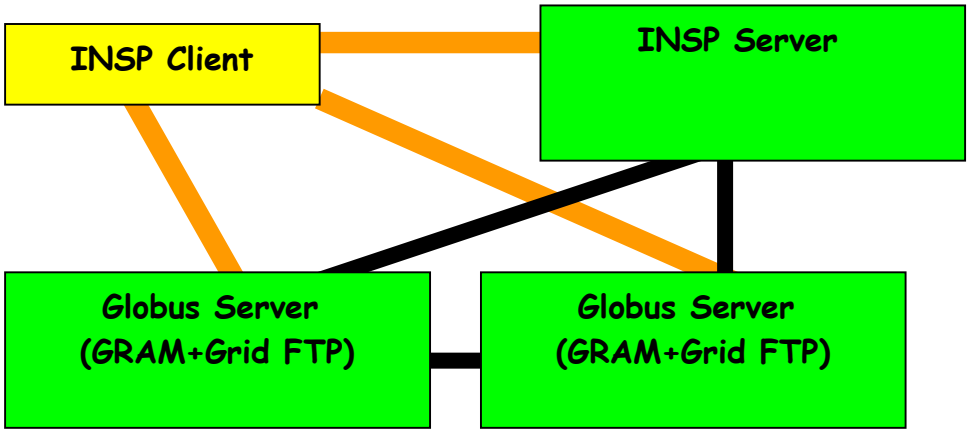


Figure 7. Current INSP architecture.

Grid-enabled INSP. We are currently redesigning INSP to allow the client to use the Globus server to run flows and view/manage data. The objective is to replace the current INSP design:



with a Grid-enabled design shown below:



In this design, security will be handled by the Globus server. The A&A policies will become Grid-specific. While the current INSP server provides an additional layer of authorization (ACL type of permissions for every project), the same can be done at the filesystem level, so any system administrator will be able to handle this task without having to learn a new system. The flows will be simple files which can reside either on the local disk or on the server. So the sharing will be done in the same way as with data files – the user can run his/her own flows or can save them on a server for later use. When connecting from different machines to the same server, the same list of files will be seen, the same as with an FTP client. The flows and data will be logically separated on what we call the project database, which will be an XML file containing all necessary information for a seismic project plus the paths to data and flows. In this framework it is possible to have exactly the same structure as in the present version of INSP - and this is the objective – to enhance it for better usability in a Grid environment. Our plans are to

incorporate Ganglia in INSP by creating a Java API to Ganglia: The Ganglia monitor daemon collect cpu load data, etc. and broadcasts this info.The Ganglia metadaemon runs on a single node and collects this info in an XML database. Our objective is to have an API on Java that can access the monitor daemon.

5. Conclusions

Seismic imaging is an application area where Grid computing holds great promise. Today's operational environment, shown in Figure 8, involves many inefficiencies that are seamlessly resolved in a Grid environment. The data tapes are transported physically (typically by UPS or FedEx) between the raw data acquisition site, data banks, data processing sites (usually a seismic contactor), and quality control and interpretation sites - the customer oil and gas production company. Once the data is interpreted it is often reprocessed with a new set of parameters. The process is repetitious and lengthy, culminating in decision to drill a well which can cost in the order of $30 M. In today's operational environment, a large imaging project can easily require one to two years to complete.

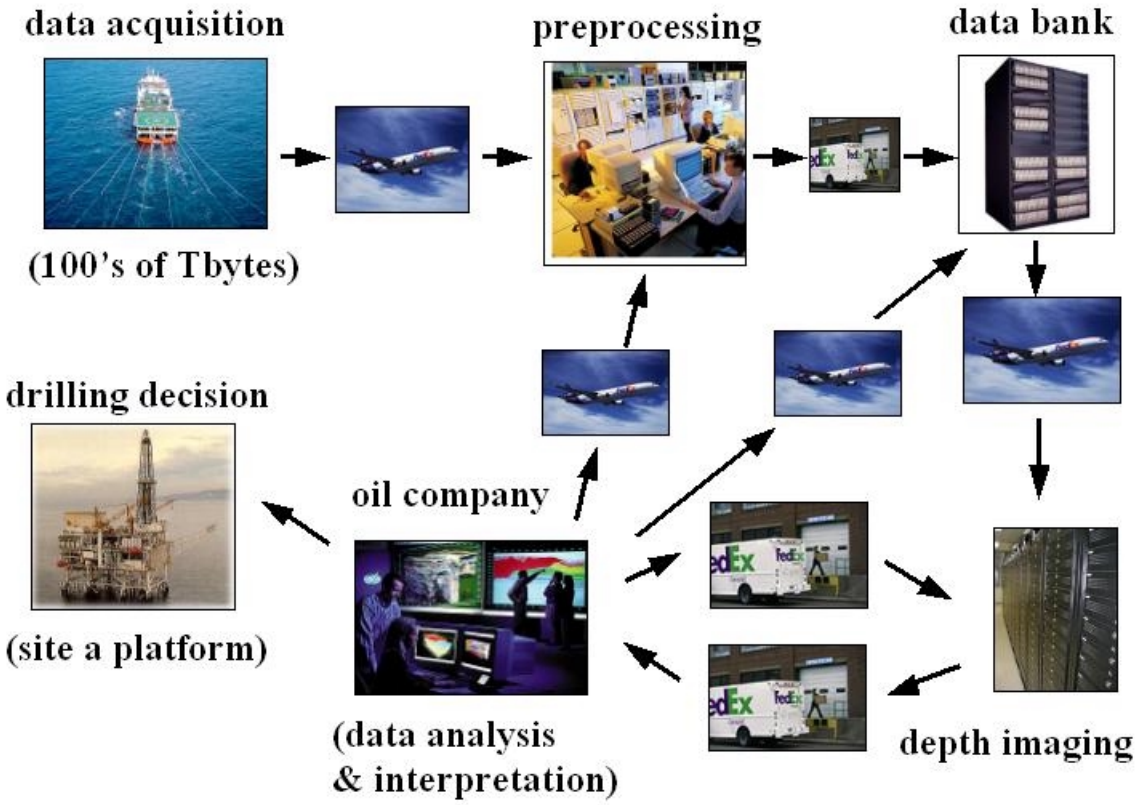


Figure 8. In today's operational scenario data are physically transported between the acquisition, processing, and storage sites, and the end user (oil company).

In a fully-optimized Grid operational environment, shown in Figure 9, we expect the time to complete the same project to be reduced to six months or less. This framework will necessitate a wavelet-based compression utility to allow for electronic transmission of extremely large datasets (Bevc *et al.*, 2004). The data will be monitored as it is being acquired, quality control will be concurrent with data processing, and interpretation and reprocessing will be done with much greater flexibility. These advantages are compelling, and have motivated 3DGeo's efforts to be at the forefront of bringing the Grid to the energy exploration industry.

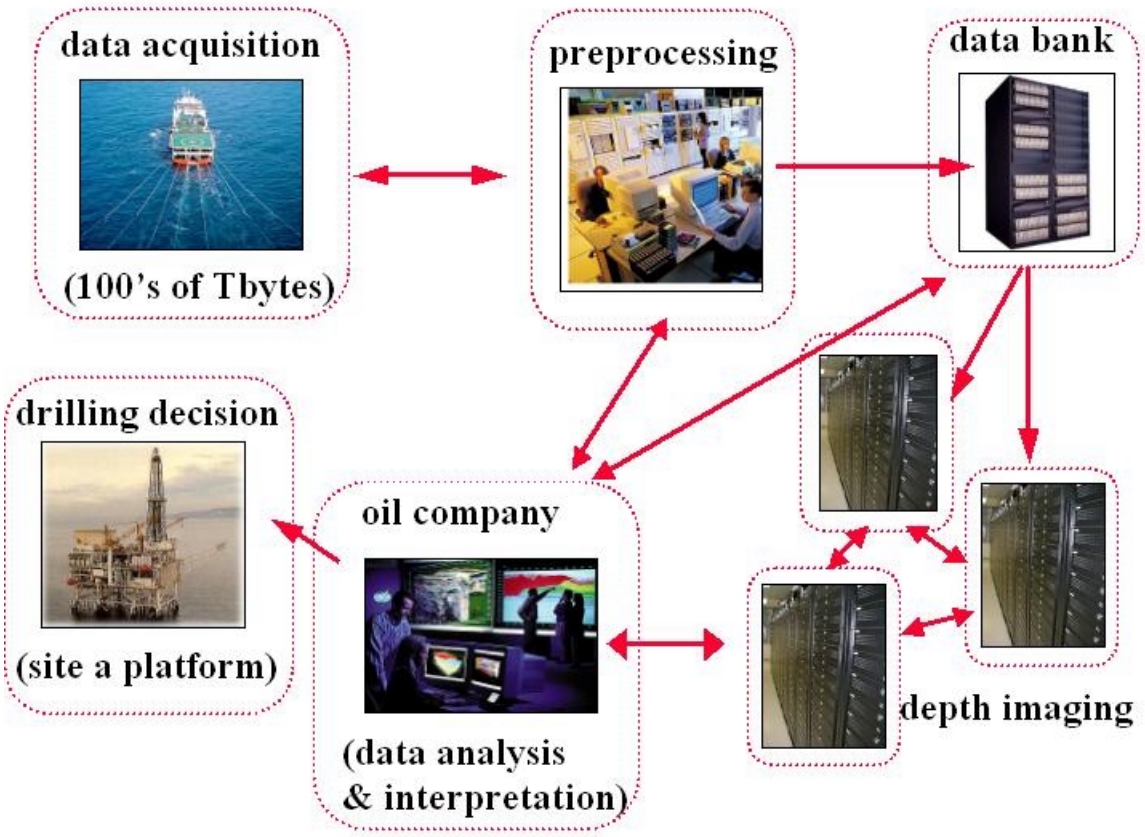


Figure 9. The future : a fully-enabled Grid scenario that by allowing greater and more flexible access to resources (data, computers, personnel), reduces turn around time and dramatically shortens the time to making a drilling decision. In this scenario, all components of the process become Grid nodes accessible through INSP. Data transfers are via GridFTP; resource requests are through Globus Resource Allocation Manager.

A Grid-enabled environment for seismic imaging allows for a new paradigm of commodity computing. In the life of a seismic imaging project the demand for compute cycles will ebb and swell as the project proceeds through its different stages. Buying compute cycles on the Internet whenever needed will free the oil and gas exploration industry to focus on its core competencies, and will result in dramatically increased productivity.

References

Bevc, D., Donoho, D. L., and Zarantonello, S. E., 2004, "Application of 2nd generation wavelets in seismic imaging"*, Proceedings of 74'th Ann. Internat. Mtg: Soc. of Expl. Geophys.*

Bevc, D., and Popovici, A., M., 2003, "Integrated Internet collaboration", *The Leading Edge*, **22**, pp. 54-57.

Bevc, D., Popovici, M., and Biondi, B., 2002, "Will Internet processing be the new paradigm for depth migration interpretation and visualization ?", *First Break*, **20**, no. 3.

Biondi, B., and Palacharla, G., 1996, "3-D prestack migration of common-azimuth data", *Geophysics,* **61**, pp.1822-1832.

Foster, I, and Kesselman, C., 1997, "Globus: A Metacomputing Infrastructure Toolkit", *The International Journal of Supercomputing Applications and High Performance Computing*, **11**, issue 2, pp. 115-128.

Groop, W., Doss, N., and Skjellum, A., 1996, "A high-performance, portable implementation of the MPI message passing interface standard", *Parallel Computing*, **22**, no. 6, pp. 789-828.

Karonis, N., Toonen, B., and Foster, I., 2003, "MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface", *JPDC*, **63**, no. 5, pp. 551-563.

Park, K-R, *et al*., 2004, "MPICH GP: A Private-IP-enabled MPI over Grid Environments", *Proceedings of the 2nd International Symposium on Parallel and Distributed Processing Applications*.

Acknowledgement

This work was supported in part by NASA. We take this opportunity to thank Tom Hinke, of NASA and Kevin Walsh of SDSC for their insight and support, and Moritz Fleidner and Mihai Popovici of 3DGeo Development Inc for their contributions.