A Computational Environment for the Management, Processing, and Analysis of Geological Data

Matthew O. Ward
Computer Science Department
Worcester Polytechnic Institute
Worcester, MA 01609 USA
William L. Power and Peter Ketelaar
Fractal Graphics
CSIRO Division of Exploration and Mining
P.O. Box 437
Nedlands, Western Australia, 6009 Australia

January 7, 1999

Abstract

The complexity and diversity of data and processing now being used in geoscientific data analysis are increasing at such a pace that it is difficult to maintain a single software package capable of performing all required tasks. A typical solution is to use a number of commercial software packages loosely linked via file format conversion routines or other customized communication paths. Creating and maintaining these paths is made difficult by both the number of distinct packages being used and by the evolutionary changes taking place within each package. In this paper we attempt to create a generic model for a geological data processing task, identify some of the difficulties involved in performing these tasks using current technology, and outline a computational architecture that may correct many of the deficiencies of existing technology. Our goal is not to provide a detailed design but to present some key issues and possible solutions as a foundation for discussion and investigation.

Key Words: Computational environments, database management, distributed processing, meta-data

Introduction

Geoscientists are becoming increasingly dependent on computational support for both the analysis of data and the simulation of geological processes. Accordingly, a plethora of computer programs for use in geoscientific data collection, storage, and analysis have been developed. Throughout this paper the term *Application Software Package (ASP)* refers to any software system designed to solve computational problems in one or more domains. Most ASPs are comprised of a set of smaller modules (functions or subroutines) that the user invokes in sequence on his or her data sets. We refer to these modules as *Data Manipulation Operators*, or simply *operators*. Additional terms frequently used include 'experiment' or 'procedure', a sequence of operators applied to one or more data sets to produce a clearly defined result. An example of an experiment might be to perform ore grade estimation on a particular data set using a specified sequence of operators.

The computational environment for an individual user can be defined as all ASPs and data sets that may be applied to solving a given set of problems. Our examination of the typical computational environment in use today by geoscientists has resulted in the following observations.

- Geology and geophysics are information intensive; that is, many types of information are needed to solve problems. Much data is file-based, creating difficulty in keeping track of specific data as the volume of data increases. Another significant problem is that the required data may be distributed over a number of computers.
- Geological and geophysical data analysis is growing in complexity. Effectively using all the ASPs available to accomplish a task (or even keeping track of those available for a particular task within a single organization) is very difficult.
- Frequent changes in data and analysis methods create a significant problem in maintaining a computational environment that is sufficiently flexible to accommodate frequent modifications.
- Developing and maintaining a single software package to meet these rapidly evolving needs has become labor intensive. Frequently, users rely on a range of software products.
- More users rely on distributed, networked, non-homogeneous computation facilities to perform routine tasks, because it is often difficult to find the required ASPs on a single computer.

- Though many ASPs have been designed specifically for geological investigations, others have been adapted from other disciplines and do not consider the special needs of the geosciences.

Computational support in geology consists predominantly of an assortment of stand-alone ASPs for data analysis, collection, storage, visualization, and report generation, each with its own database and/or data file formats (Bellotti and Dershowitz, 1991). Creation of a single ASP to support the changing needs of users has become impossible, and most users now realize the importance of *interoperability* among ASPs to accomplish distinct components of larger tasks. This need for interoperability is often managed through customized data transfer protocols, where special code is written to link each pair of application programs. The problems with this approach include the following:

Extensibility: The number of linkages needed may approach N^2, where N is the number of ASPs being interfaced. This becomes a serious problem as N increases. While standardization of data formats reduces this problem, software developers have shown reluctance to adopt these standards. A simple method is needed to incrementally enhance the functionality of a system by integrating new ASPs and data sources.

Maintainability: Each time one of the application programs is modified, N-1 modifications of the interfaces to the other applications may be required.

Usability: Effective use of each ASP generally requires user expertise. As N increases it becomes difficult for users to achieve the required level of knowledge for each ASP.

Consistency: Typically, each ASP works in different ways, providing different levels of sophistication in its user interface, different degrees of user assistance, and different methods of task invocation. Optimal selection and use of an algorithm within an ASP may require assistance from an expert.

There is a research project under way at the CSIRO Divisions of Information Technology and Exploration and Mining (Ward and others, 1994) to design a computational environment for geological data analysis that alleviates some of the problems mentioned above. Our goals in this paper are the following:

1. To generalize the process of geological data processing using detailed scenarios.

2. To present a framework for a cohesive, extensible computation and information management environment. This would consist of a central scheme for the description and representation of data and operators; an information management system (covering data, operators, data manipulation procedures, and auxiliary information); linkages to existing analysis, display, and report generation ASPs; and a visual interface for browsing available data, operators, procedures, and experiments, specifying and invoking queries and data analysis tasks, and interactively editing data and generating reports.

3. To illustrate the usefulness of this framework by replaying the geological data analysis scenarios mentioned earlier assuming the existence of this framework.

Examples of Geological Data Analysis Tasks

Most geological data processing tasks can be characterized by one or more data sets passing through a series of data manipulation steps, often culminating in visual presentation of the results. The data itself may reside in multiple forms (files, databases) on distributed and possibly heterogeneous hardware platforms. Similarly, data manipulation operators may reside in different ASPs, each with their own format for invoking operations, and potentially on distributed and heterogeneous platforms (Baumerwerd-Ahlmann and others, 1991). The components of this process can be seen in the framework outlined in Figure 1, and a typical task may require multiple instances of each component. A brief description of each module of the framework is given below. Some detailed examples of tasks currently performed within this framework are presented in the following subsections.

Distributed Analysis Tools refers to all ASPs in current use, which may reside on multiple, diverse hardware platforms. These include not only analysis and modeling ASPs, but also visualization, report generation, and data conversion facilities. **Distributed Shared Data** consists of files, remote archives, or databases generally accessible to multiple ASPs. **Custom Access Tools** are the functions used to retrieve data from databases and remote archives and convert data formats to allow interoperability. Many ASPs maintain their own non-sharable, application-specific data, which we refer to as **Private Data**.

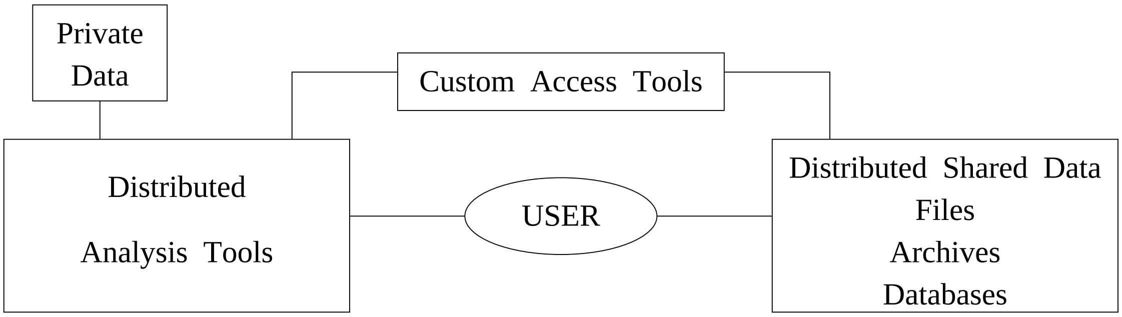

Figure 1: Components of current computational environments for geological data management, processing, and analysis and their interrelationships.

Geological Model Construction and Editing

Figure 2 shows the typical stages involved in the creation of a 3-D solid geological block model for use in such tasks as mine planning and design, stress and strain analysis, and geophysical forward modeling. The process consists of entering information from a variety of sources (computer files, plans, sections, and maps) and converting them into a 3-D polyhedral model representing the spatial distribution of rock units, faults, fractures, and geologic contacts. The result must be compatible with the 3-D geometry manipulation packages to be used. The final result must also be converted into formats compatible with the type of visualization or other processing desired. The results of visualization may also be directed to a hard-copy output device, which again usually requires some form of data transformation.

Finite Element and Finite Difference Modeling

Finite element and finite difference modeling techniques can be used in geology and geophysics to simulate the migration of fluids and heat and the deformation of rock materials (including stresses and strains in rock). A common requirement of all finite element or finite difference techniques is discretization of the problem domain, usually into tetrahedral or hexahedral (brick-like) finite elements or finite difference zones. In fluid flow and geotechnical applications, it is also common to utilize surface discretizations of fractures or frictional interfaces. This discretization, however, is unlike voxels or mine block models, because the orientations and positions of the interfaces between the different rock units can be of critical importance to fluid flow, mechanical deformation, or frictional sliding. The normal procedure involves using customized software to create the mesh, usually with significant user input via mouse or keyboard. The resulting mesh

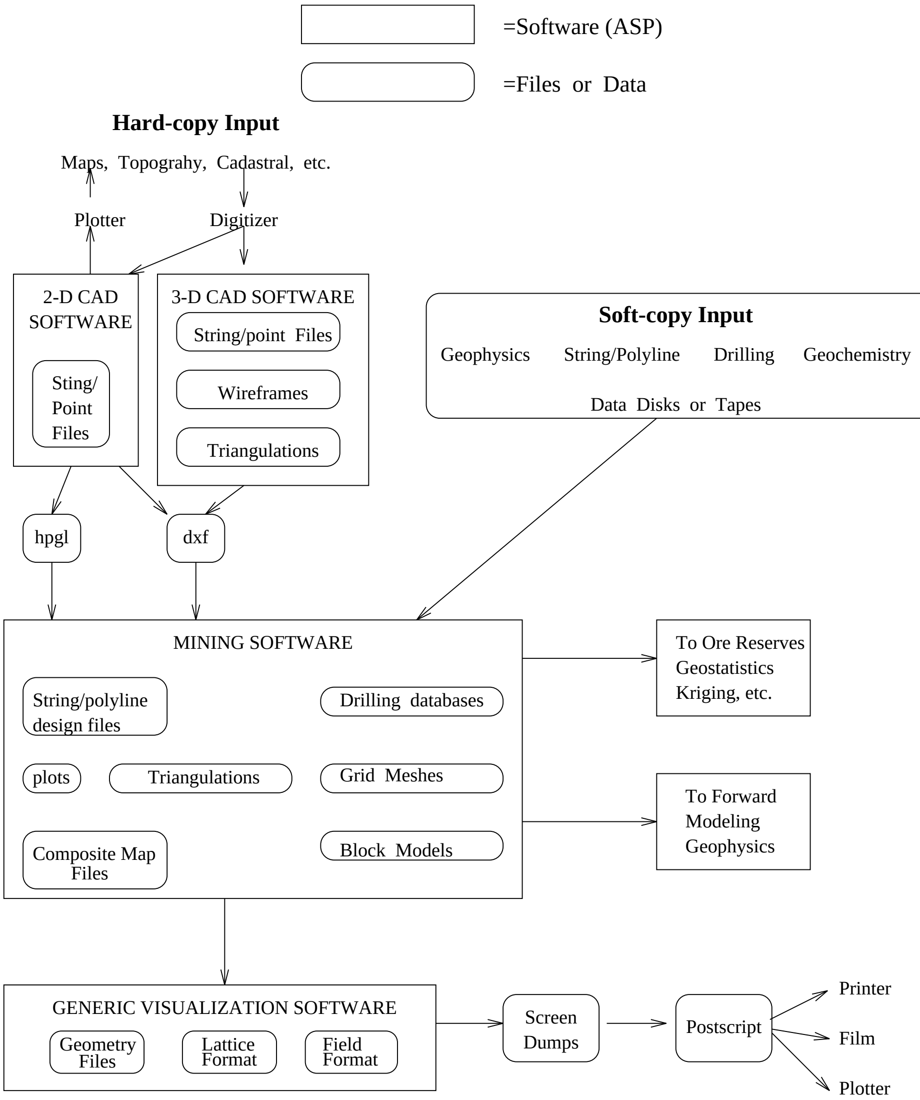

Figure 2: Geological model building flowchart.

is usually an equivalent, although in some cases simplified, version of the initial 3-D solid geology. Simulations of transport or deformation are performed, using the discretization. Finally, the results must be displayed in an easily readable format, such as color coded displays, contouring, or other symbols. In some cases hard-copy results are required. In most cases it is necessary to repeat the entire process more than once to update either the rock properties or the 3-D solid geology, based on results obtained in earlier simulations (see Figure 3).

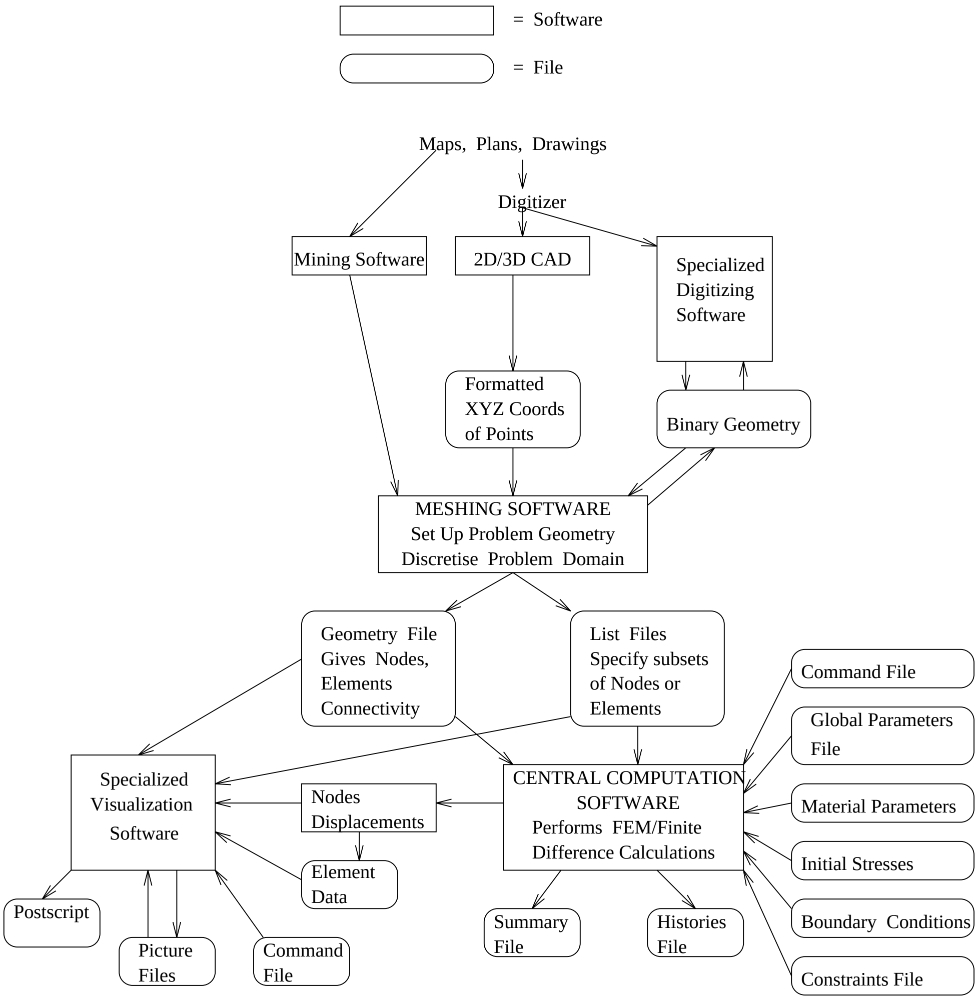

Figure 3: Finite element/difference flowchart.

Computer Knowledge Needed to Perform Tasks

The tasks described above require extensive computer expertise on the part of the geologist or geophysicist. In general, the user must become acquainted with five aspects of an ASP. The first relates to overall functionality, which can be summarized by the data analysis techniques available to the user. Second, the types and formats of information that can be processed by the ASP are described by the geological data structures used and the geological meaning of the data within these structures. We refer to this component as the *Data Structure and Semantics* (DSS). Third, the *Operator Structure and Semantics* (OSS) specifies the protocol (e.g. input and output parameters) for invoking a Data Manipulation Operator and the contextual meaning of the operation. Fourth, the Human-Computer Interface specifies the communication possible between the user and the ASP, and permits the user to invoke operators, interact with executing operators, and examine results (generally in a graphical fashion). Finally, some form of information management is required, as the data to be used by the ASP may reside in many forms, including files, remote archives, and databases.

If we consider the ramifications of interoperability between ASPs, the characteristics of data being applied to solving problems (distributed, diverse formats), and the growing complexity of data analysis tasks, the demands on the user become formidable:

Data formats, meaning, and access methods: Current geoscientific data analysis is performed predominantly in a file-driven mode; users are responsible for tracking the content, format, and location of potentially hundreds of data files. A single experiment may involve the creation and naming of several files for communication between operators. Some of these files are transient and need to be removed, others are useful as input to other tasks. Maintaining knowledge of what processes a given file has been subjected to is usually left to the user. Data may also be stored in remote archives or databases, which requires additional knowledge to retrieve data and store the results.

Data conversion techniques: Users must be familiar with programs that convert data between different formats. Each format transformation may result in the loss or modification of information, a fact that is often lost in the life cycle of a data set.

Functionality of data manipulation operators: Users must know what task each operator within an ASP can perform. Unfortunately, these operators are often identified solely by some cryptic, abbreviated name, and it is the responsibility of the user to match this name with the task the operator performs via on-line or (more commonly) hard-copy manual pages.

Format for operator input and output parameters: Each operator accepts input and generates output of a certain format. The user must verify that input has been specified appropriately and that output is interpreted correctly for each operator invoked.

Procedures for using each operator: The structure of the command to invoke a given operator varies widely, both between ASPs and within a single ASP. Knowledge of parameter order, valid ranges for numeric parameters, and interdependencies between parameters is required of the user.

Procedure for using the ASP that contains each operator: Each ASP has its own user interface, including command-line task specification, simple menu structures, and elaborate graphical front-ends. Each interface assumes a certain level of knowledge and provides varying degrees of assistance/error handling.

Procedure for using systems containing data and ASPs involved: Currently, more geoscientists employ multiple computer hardware (PCs, workstations, mainframes) and operating system (DOS, UNIX, Mac-OS, VMS) platforms in performing data analysis, requiring not only some level of proficiency in using these systems, but also knowledge about transferring data between machines.

Reproducing complex sequences of operations: Often the user must remember a potentially complicated series of operations, creating and naming intermediate data repositories along the way and hoping that no step has been forgotten. A common problem with complex tasks is reproducing exactly the steps taken to generate a result. Simple note taking is ofthen the only documentation of a procedure.

The term used to describe much of the information or knowledge above is *metadata*, or data about data or tasks(Bretherton and Singley, 1994). Metadata may be as simple as a file header indicating characteristics of the data within the file, such as spatial extents, time of acquisition, method of acquisition, and so on. Other more complex metadata may include format information, instructions on using data or operators, or the sequence of processing stages that produced a result.

Most metadata in scientific data processing currently resides either in the minds of the users or as user-generated text files. This substantial burden on the user could be eased by introducing a computational framework such as the one outlined in the next section.

Proposed Structure for a Computational Environment

The critical aspects of a software architecture that could potentially meet the needs of the various components of geological and geophysical data analysis and modeling include the following:

Extensible Data Structure and Semantics (DSS): The type, structure, semantics, and interrelationships of data being used to solve problems is constantly evolving, thus necessitating extensibility.

Extensible Operator Structure and Semantics (OSS): New ASPs and operators are appearing in the literature and on the market daily. A mechanism is needed to facilitate incorporation of these into a user's work environment.

Context Sensitivity: Tasks and information should be accessible in a manner natural to the geoscientist. The user should not be interacting with bits and bytes, but functional modules and geologically labeled information. The data and ASPs available to a particular user can also be limited based on the task at hand.

Intuitive User Interface: The complexity and diversity of software (and hardware) indicate a need for an interface that frees users from low-level details. Visual point-and-click interfaces and graphical information presentation have been shown to provide effective front-ends to complex analysis.

Experiment Management: A mechanism is needed to provide high-level creation, execution, modification, and documentation of scientific procedures and the data to which they are being applied.

Metadata Management: Metadata associated with data, operators, and experiments must be managed.

Module Descriptions

Figure 4 shows an initial attempt to define an architecture containing the previously described capabilities. The shaded area depicts our view of the computational environment commonly found today; the unshaded area shows proposed extensions to this environment. This structure is based loosely on Gaea (Hachem, Gennert, and Ward, 1993), a spatio-temporal data analysis and management system under development at Worcester Polytechnic Institute for use in geographical information processing. The descriptions below outline the purpose of each of the system modules.

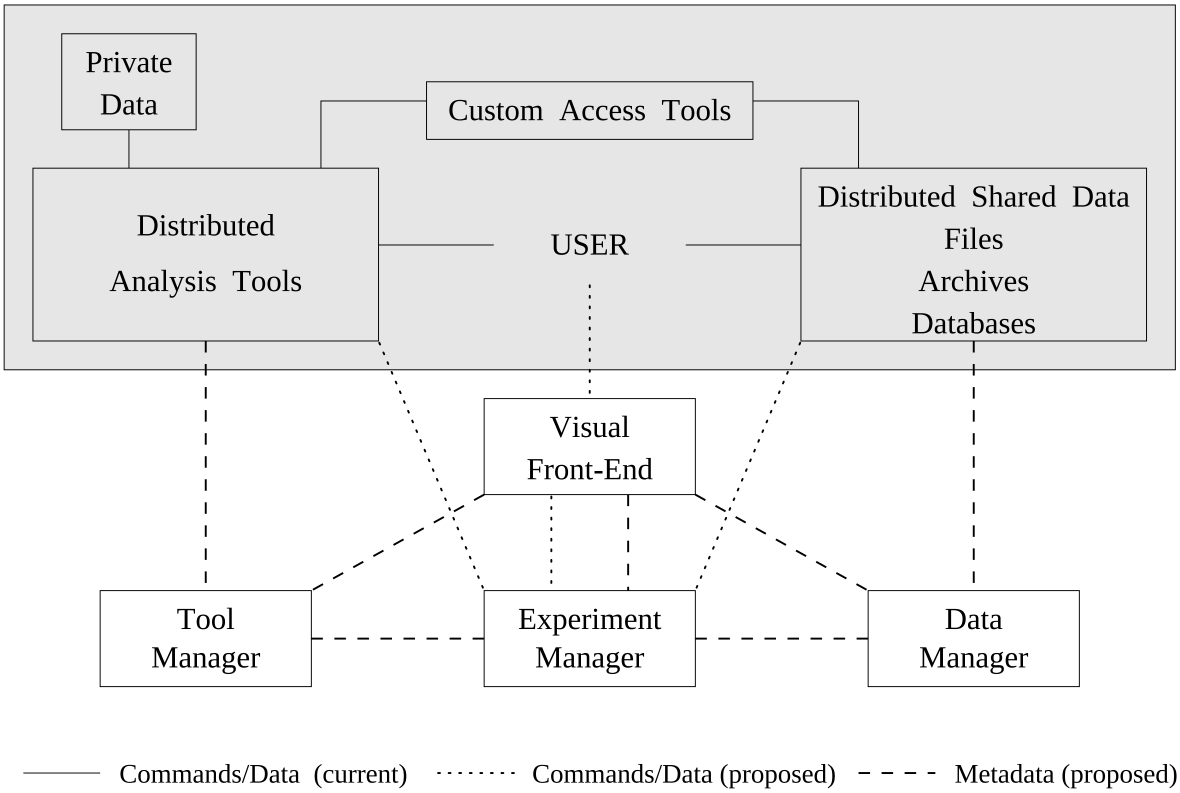

Figure 4: Components of the proposed computational environment and their interrelationships. The shaded area indicates the components of current computational environments.

The **Visual Front-End**, or Graphical User Interface (GUI), is used for browsing existing data, operators, metadata, and experiments, for creating, editing, and invoking experiments through the Experiment Manager, and for displaying all forms of information. It is separate from the Experiment Manager to facilitate portability and the distribution of the modules over multiple platforms. The **Experiment Manager** performs data/operator compatibility checks and implicit type conversions, invokes completely specified experiments, captures static and dynamic information for experiments (data, operators, parameters, interactions, annotations), and manages and assists in browsing of experiment metadata. The **Tool Manager** performs maintenance on the OSS, keeps track of communication protocol for various packages and platforms and supplies them to the Experiment Manager as needed, provides facilities to allow simple addition of new operators and ASPs, and manages and assists in browsing of operator metadata. The **Data Manager** performs maintenance on the DSS, keeps track of communication protocol for various data sources and supplies them to the Experiment Manager as needed, provides facilities to allow simple addition of new data sources and types, and manages and assists in browsing of metadata.

Intercommunications between Modules

The communication between modules can be categorized as Commands/Data and Metadata (see Figure 4). The description of each is presented below.

User to Visual Front-End: User performs tasks using a mouse and keyboard.

Visual Front-End to Experiment Manager: Information flowing between the Visual Front-End and Experiment Manager is generally in the form of a procedural description, composed visually and then converted into a language interpreted by the Experiment Manager. The Experiment Manager also provides metadata to the Visual Front-End for display as needed, as well as requests for input for running experiments.

Experiment Manager to Tool Manager: The Experiment Manager acquires operator metadata from the Tool Manager and checks for consistency between the selected operators and data. It also acquires the necessary communication protocol for operator execution.

Experiment Manager to Data Manager: The Experiment Manager acquires metadata about data from the Data Manager and checks for consistency between the selected operators and data. It also acquires the necessary communication protocol for data retrieval and formatting.

Experiment Manager to Distributed Shared Data: The Experiment Manager retrieves and formats data as required by the specified experiment based on the protocol passed from the

Data Manager.

Experiment Manager to Distributed Multi-platform Analysis Tools: The selected operators are invoked by the Experiment Manager with the protocol retrieved from the Tool Manager, passing requests for interaction from the operator to the Visual Front-End. Information from the user is passed to the operator requesting it.

Visual Front-End to Tool and Data Managers: It might be useful for the Visual Front-End to be able to access the Tool and Data Managers directly when performing browsing tasks, as the Experiment Manager is not needed for this.

Tool Manager to Distributed Multi-platform Analysis Tools: The Tool Manager must track the availability of operators.

Data Manager to Distributed Shared Data: The Data Manager must track the availability of data.

Performing Geological Data Analysis Tasks with the Proposed Computational Environment

Returning to our examples of geological data analysis, and now assuming the existence of the proposed architecture, the first major difference the user would notice is the elimination of boundaries between ASPs. The appearance would be as if a single ASP had enveloped the functionality of all ASPs linked to the Tool Manager, even though each ASP could in fact be executing on a different computer. A procedure such as 3-D solid geology model creation might already exist as an experiment (which the user could execute with different data sets), or the task may be interactively specified by functional components (vectorization, visualization, interactive specification of surfaces and volumes) and flow of data between operators.

Metadata associated with each operator would be presented to the user to verify the appropriateness of the operator for the desired task. Alternative operators might be queried and selected to replace any operator deemed suboptimal by the user. For example, in Finite Element Modeling different mesh generation operators could be substituted until the best results were obtained.

Input data for an experiment would be specified, not by a file name, but by attributes of the desired information. The user could specify the geographical region of interest (e.g. a section of a particular mine), a temporal constraint, if applicable, and the data type (e.g. radiometrics, drill hole data, or geophysical rock properties). The user could be totally unaware of the location or format of the data.

Any data format conversion necessary to make the retrieved data compatible with the selected operator would be inserted into the procedure automatically by the Experiment Manager (if conversion operators didn't exist, the user would be prompted to modify the data or operator selected). Inputing a mine block model into a forward modeling operator might automatically invoke a voxelation operator. Intermediate data and final results would be stored along with metadata describing the process that created it. Users would have the option of discarding them or storing them permanently. Stored data would be classified by the user to facilitate future retrieval.

Metadata associated with a data set could also be browsed by the user to better understand what data is available as well as the origins and modification history of the data. Understanding how a data set was created and processed is important both for validation and for the generation of additional data with similar characteristics.

With the proposed architecture much of the computer knowledge previously required becomes encapsulated into the computational environment in the following fashion (referring to the list of the section entitled "Computer Knowledge Needed to Perform Tasks"):

Data formats, meaning, and accessing methods: Data to be used or created is specified by its semantics (meaning), not by its format or location. The Data Manager holds the metadata necessary to acquire data. The user would request data by specifying its spatial and temporal extents and data type.

Data conversion techniques: When the format of a data set is incompatible with the selected operator, the Experiment Manager will insert conversion operators, and the metadata associated with the resulting data object will be updated to reflect this stage. Parameters to this conversion will be selected by the Experiment Manager, although the user may wish to override some or all of the selections.

Functionality of data manipulation operators: The Tool Manager presents the user with classes of operators based on their abstract functionality and the types of data they routinely operate upon. Operators from ASPs with similar functionality will appear in the same class.

For example, a class of functionally related operators would be 3-D interpolation algorithms from one or more distinct ASPs. Implementation details for specific algorithms are available as metadata through the Tool Manager to help users select appropriate operators.

Format for operator input and output parameters: Users are prompted for appropriate numbers of input data, which are further specified using the data browser. Low level details of data formats are hidden from users. Output data sets are created by the operator and the metadata associated with them are automatically inserted.

Procedure for using each operator: Commands for invoking each operator are contained in the Tool Manager, so that the location of each operator and what ASP it is part of is hidden from the user. Additional user input is requested through context-sensitive prompts in a consistent fashion.

Procedure for using ASP containing each operator: The user doesn't require knowledge of the ASP in use, though he/she can specify a particular operator in a particular ASP if desired.

Procedure for using systems containing data and ASPs involved: The user has a single interface to the system, and thus does not require knowledge of operating system calls, file system formats, and so on.

Reproducing complex sequences of operations: Details of the data and operators used in an experiment are captured automatically as metadata, permitting reproduction of the results as well as reuse of an experiment with modification in the data or operators used.

Related Work

Besides the Gaea Project (Hachem, Gennert, and Ward, 1993), there are a number of efforts under way to improve the management of large volumes of distributed scientific data and integrate this data with existing ASPs. Some of the projects most pertinent to our work are identified below.

- A design for an information management system for the mining industry is described in (Baumerwerd-Ahlmann and others, 1991), that integrates diverse forms of mining data into a single database and provides conversion routines to allow the processing and display of data using a variety of existing ASPs.

- Alexandria is a project under way at the University of California, Santa Barbara with the goal of developing a geographically dispersed virtual library that attempts to eliminate the distinction between different types of stored data, allowing access to maps, images, and text with equal ease. The focus is mainly on data storage and access, and not on the processes that generate and manipulate the data (Dozier and others, 1994).

- The QUEST project (Mesrobian and others, 1994) is focused on providing content-based access to very large Grand Challenge data sets. The main issues are permitting user queries at an appropriate level of abstraction, natural user interaction, and the application of massively parallel processors. The approach employs statistical techniques to extract relevant features defined over a spatio-temporal extent and uses these features as database indices.

- Sequoia 2000 (Stonebraker, 1994) has a goal of furthering climate research by making very large data sets more accessible. A large part of this work involves high-performance storage systems and network design to improve the management and access speed for distributed data sets.

- TSIMMIS (Chawathe and others, 1994) is a research project at Stanford University that is primarily aimed at developing language support for heterogeneous systems, focusing on declarative and semi-declarative languages and query optimization techniques for these languages. Another related research area is in data sets that describe themselves.

Summary

We have analyzed the computational and information management needs of geoscientists and compared these needs to available technology. We have identified key shortcomings in current geoscientific data analysis practices and have proposed an abstract design for a computational architecture which, we believe, is well suited to addressing these missing features. The key concepts of the proposed architecture include 1) a consistent and intuitive user interface; 2) management of data, data analysis operators, and experiments; 3) maintenance of metadata for each of these components; and 4) interoperability among diverse data sources and Application Software Packages.

The trend is to rely on several application packages, data types, and storage mechanisms to solve increasingly complex problems. By encapsulating the details of interoperability within the computational system, the user is relieved of the need to maintain this knowledge. By capturing the essence of complicated experiments for later validation and modification, data analysis procedures are facilitated. By incorporating within the system as much information as possible about the data and ASPs available to users, we hope to create a more powerful architecture for solving geoscientific problems.

Acknowledgements

The authors wish to thank the large number of people within the CSIRO Divisions of Information Technology and Exploration and Mining who have contributed to the evolution of this document.

References

Baumewerd-Ahlmann, A., Cremers, A.B., Kruger, G., Leonhardt, J., Plumer, L., and Waschkowski, R., 1991, An information system for the mining industry: Proc. Database and Expert Systems Applications (DEXA '91), D. Karagiannis (ed.), Springer-Verlag, p. 86-91.

Bellotti, M.J., and Dershowitz, W., 1991, Hydrogeological investigations: data and information management: Computers & Geosciences, v. 17, no. 8, p. 1119-1136.

Bretherton, F.P., and Singley, P.T., 1994, Metadata: a user's view: Proc. 7th Intern. Working Conf. on Scientific and Statistical Database Management, p. 166-174.

Chawathe, S., Garcia-Molina, H., Ireland, K., Papakonstantinou, Y., Ullman, J., and Widom, J., 1994, The TSIMMIS Project: Integration of Heterogeneous Information Sources: Proc. IPSJ Conference, Tokyo, Japan, p. 7-18.

Dozier, J., Goodchild, M., Ibarra, O., Mitra, S., Smith, T., Agrawal, D., El Abbadi, A., and Frew, J., 1994, Towards a distributed digital library with comprehensive services for images and spatially-referenced information: Proposal to NSF Digital Libraries Initiative, available as `http://alexandria.sdc.ucsb.edu/public-documents/proposal/0main.html`.

Hachem, N., Gennert, M., and Ward, M., 1993, The Gaea system: a spatio-temporal database system for global change studies: Proc. of American Association for the Advancement of Science Annual Meeting, p. 84-89.

Mesrobian, E., Muntz, R.R., Shek, E.C., Mechoso, C.R., Farrara, J.D., and Stolorz, P., 1994, QUEST: content-based access to geophysical databases: Proc. AAAI Wkshp. AI Techniques for Environmental Applications, Seattle, WA, 9 p.

Stonebraker, M., 1994, Sequoia 2000—a reflection on the first three years: 7th Int. Working Conf. on Scientific and Statistical Database Management, p. 108-116.

Ward, M., Lin, T., Kravis, S., and Lamb, P., 1994, A conceptual design for a computational environment for geological data analysis and management: Technical Report TR-HJ-94-06, CSIRO Division of Information Technology, 24 p.