UNIFORM ACCESS TO SIGNAL DATA IN A DISTRIBUTED HETEROGENEOUS COMPUTING ENVIRONMENT

Steven Jeffreys
BBN Systems and Technologies
A Division of Bolt Beranek and Newman Inc.
10 Moulton Street, Cambridge, MA 02138

KEYWORDS

Computer Networks, Data Analysis, Distributed Computing

ABSTRACT

One of the problems in analyzing data is getting the data to the analysis system. The data can be stored in a variety of ways, from simple disk and tape files to a sophisticated relational database system. The variety of storage techniques requires the data analysis system to be aware of the details of how the data may be accessed (e.g., file formats, SQL statements, BBN/Probe commands, etc.). The problem is much worse in a network of heterogeneous machines; besides the details of each storage method, the analysis system must handle the details of network access, and may have to translate data from one vendor format to another as it moves from machine to machine.

This paper describes a simple and powerful software interface to telemetry data in a distributed heterogeneous networking environment, and how that interface is being used in a diagnostic expert system. In this case, the interface connects the expert system, running on a Sun UNIX machine, with the data on a VAX/VMS machine. The interface exists as a small subroutine library that can be linked into a variety of data analysis systems. The interface insulates the expert system from all details of data access, providing transparent access to data across the network. A further benefit of this approach is that the data source itself can be a sophisticated data analysis system that may perform some processing of the data, again transparently to the user of the interface. The interface subroutine library can be readily applied to a wide variety of data analysis applications.

INTRODUCTION

In the past, data analysis was done entirely on a single machine. The data would arrive on an analog tape, and be digitized and placed on a disk or other storage device on the machine using specialized hardware and software, resulting in a file of raw data. It was not unusual for the data to undergo further filtering or reformatting (resulting in yet another copy of the data) before being analyzed using a tool built just for that purpose. Given enough time, a whole suite of formatting and analysis tools would grow up around a particular telemetry stream. This situation has some advantages: the data is easy to get to, and the tools work. Unfortunately, as the processing demands increase, things go downhill rapidly. Upgrading to a faster machine is not always practical, as the specialized hardware and software that evolved over the years may not be portable to a new platform. Likewise, home-grown analysis tools may not be able to respond to new demands (new data formats, new report formats, new types of analysis) without extensive (expensive) modification. The recent proliferation of fast, cheap hardware coupled with the arrival of many commercially available data analysis tools seems to offer a way out, but first some new problems must be faced. Figure 1 shows a typical situation.

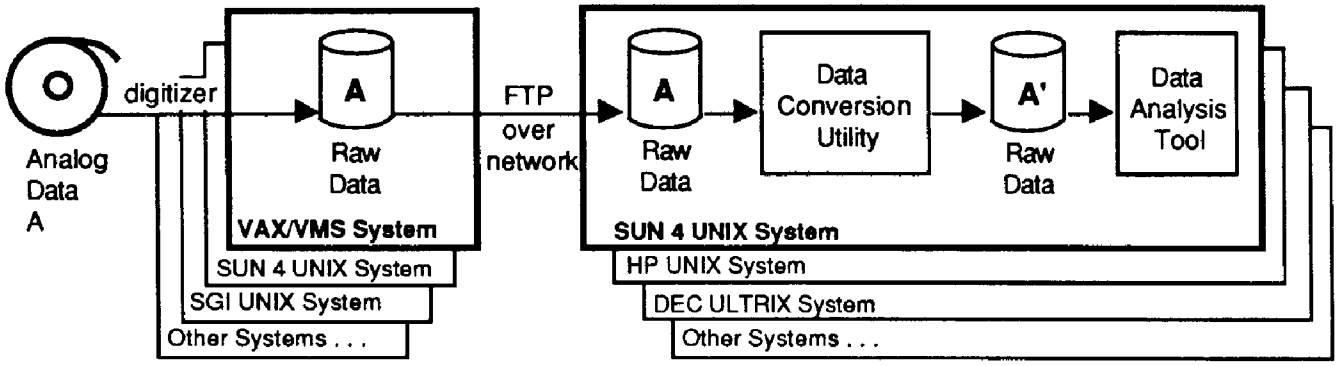

Figure 1. Accessing Data In A Heterogeneous Network. Digitized telemetry data sets are typically huge files of a unique and complex format, requiring specialized conversion routines and data analysis tools. To take advantage of additional processing power or analysis tools elsewhere in the network, the data must be copied to the remote system and converted to the appropriate format. However, it is time-consuming and wasteful to make redundant copies of huge files, especially since the data required for a particular analysis may represent only a small fraction of the available data. Likewise, it is difficult to track and manage the data, because many redundant but slightly different copies of the data are created as it moves from system to system.

Before the flexible, powerful new analysis tool running on the fast, cheap hardware can be used, the data must be brought to the right place and massaged into the right

format. Transferring the files is a lot easier than it used to be, thanks to high-speed networks and (more-or-less) standard file-transfer utilities. Even so, telemetry data files tend to be huge, and transferring them to another system takes time and consumes disk space. Once on the other system the file format almost always needs conversion because data and file formats differ considerably across operating systems and hardware platforms. Also, the new analysis tool must either be adapted to read the raw data, or the raw data must be converted to a format acceptable to the tool, creating yet another copy of the data. This process must be repeated for each different system and tool to which each different data file is transferred. Copying and translating the data more than once is undesirable. Redundant copies of data are difficult to manage, and most analysis procedures require only a small fraction of the available data.

THE DATA ACCESS UNIT

What is needed is some mechanism to bring data and analysis tools together, regardless of the format of the data or the location of the data and analysis tool in the network. The mechanism should have a simple, programmatic interface to make it easy to incorporate into new and existing data analysis tools. We call this mechanism the Data Access Unit and Figure 2 presents a simplified overview.

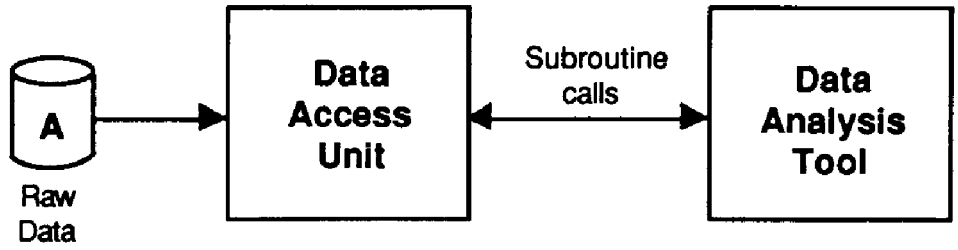

Figure 2. Data Access Unit Concept. The Data Access Unit (DAU) provides transparent access to data regardless of the data format or its location in the network. The DAU handles all details of locating, accessing, and transferring data across the network. The Data Analysis Tool (DAT) interfaces to the DAU through a small subroutine library. The DAT refers to the data set (usually a data file) and signals (telemetry variables in a given time period) within the data set by operating system independent names. The result is that only one copy of the data need exist in the system, and consumers of the data located anywhere in the network get only the data they need delivered to them in a form convenient for processing.

The principal benefit of the Data Access Unit (DAU) is that it allows the user to take advantage of all the available resources of the network to process the data. By hiding the details of locating, accessing, and translating the data behind a uniform programming interface, the data and the analysis tools can be deployed in the most advantageous fashion. Notice that Figure 2 shows no network or machine boundaries. Data entering the network on older systems (with the specialized hardware and

software) is now available to the analysis tools running on fast hardware elsewhere in the network. The existing investment in hardware and software is preserved, while at the same time new resources can be integrated into the data analysis environment. The architecture of the DAU is shown below in Figure 3.

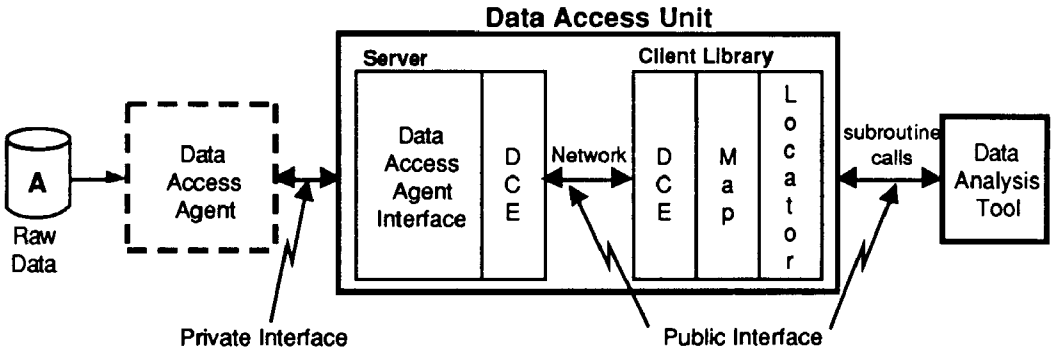

Figure 3. Data Access Unit Architecture. The Data Access Unit (DAU) is actually two components, a Client and a Server. The Client is a subroutine library linked into the Data Analysis Tool (DAT). The Server is a separate process that may be located elsewhere in the network. The Client and Server communicate using a commercial Distributed Computing Environment (DCE) package that handles all the details of network access and data translation. The Locator of the Client is responsible for finding resources (data sets and map files) requested by the Data Analysis Tool. The Map translates names of signals (variables, functions, and events) into the actual variable names within the data set. In response to requests for data, the Server may access the data directly or it may use a Data Access Agent (DAA) to perform the task. The public interface between the Client and Server and between the Client and DAT make it possible to replace both Client and Server components without affecting the DAT.

To hide the details of accessing the data from the data analysis tool (DAT), the DAU uses the concepts of data sets and signals. All communication between the DAT and DAU is through a subroutine interface. To the DAT, it appears as if the entire DAU is merely a subroutine library linked into DAT. In reality, the DAU has two principal components, a Client (the subroutine library) and a Server. The server exists as a separate process elsewhere in the network. The various components of the client and server are discussed below.

Data Sets

The key data structure shared by the Data Analysis Tool and the Data Access Unit is the data set. A data set is simply a named collection of signal data. The DAT must first open the data set by name, and receives back a handle to the data set. All subsequent requests for signal data require a valid data set handle be presented by the

DAT. So that data sets may be protected from unauthorized access, a username and password must be supplied by the DAT when it first attempts to access a data set. Finally, the data set must be closed by the DAT at some appropriate point.

Signals

Within a data set, the Data Analysis Tool (DAT) makes requests for data on one or more signals. Signals correspond to variables encoded in the telemetry stream. Signals have names that are simply alphanumeric strings. We currently support three types of signals:

- Variable. An array of timestamp-value pairs defined by the signal name.
- Function. An array of timestamp-value pairs defined by a function of zero or more signal names.
- Event. An array of timestamps defined by a boolean search condition; the timestamps represent points where the condition is true.

The DAT may request the signal by specifying a signal name, start and end times, and a data set. For functions, a function expression must also be specified, and for events a boolean search condition must also be specified. The response from the Data Access Unit is the requested data plus some descriptive information about the signal, or an indication that the request failed. Other signal types can be added later as needed.

Map Component

The Data Analysis Tool requests variables by name, where a name is simply an alphanumeric string, such as “ALTITUDE”. The variable name usually comes from a data dictionary, which defines the format of the telemetry data stream to the Data Access Agent (DAA). Each telemetry stream has its own data dictionary, and there is typically no way to enforce consistent naming of variables across similar data dictionaries. Thus, the variable that corresponds to fuel pressure may be “Altitude” in one data dictionary and “ALT” in another.

To insulate the Data Analysis Tool from having to know many names for what is actually the same signal, the map component of the Data Access Unit (DAU) automatically translates the signal name specified by the DAT into the actual variable name. A map entry indicates that occurrences of some name A must be translated to another name B. A map consists of zero or more map entries. The map component translates a variable name if a map entry applies, otherwise the name passes through

unchanged. The simple text substitution provided by mapping is not limited to mapping one variable name to another. For event and function expressions, it is possible to map a variable name to an expression, representing a form of macro expansion.

Maps can be selected and loaded by the Data Analysis Tool depending on the known characteristics of the current data set. We provide the DAT with the capability to associate maps with a data set as part of the public interface. The interface hides the details of the Map component implementation from the DAT, allowing us to enhance the implementation of this component at a later time.

Locator Component

Locating data sets becomes particularly tricky in a networked environment. Host names and addresses change as the network is reconfigured, and these changes may not be readily apparent to the user. Also, it is possible that the network could contain more than one Data Access Unit Server. The locator component is responsible for finding data sets and servers in a network environment. The location process is applied to all requests for data sets and maps, and the operation of the locator is completely transparent to the Data Analysis Tool. In practice, the DCE component provides much of the underlying mechanism used to find the data sets and maps.

DCE Component

The Distributed Computing Environment (DCE) component is the core technology around which the Data Access Unit is built The DCE hides the fact that the Client and Server may in fact be running on different machines in a large, far-flung network. The DCE transparently locates the Server, and translates data passed between Client and Server as necessary. The interface the DCE presents to the Client and Server is a Remote Procedure Call (RPC) [1] subroutine library. Several commercial DCE products exist that provide the necessary capabilities, and the DAU architecture is not tied to a particular product.

The locator component takes advantage of the DCE's ability to find Servers in the network to implement the process of locating a named data set. When the DAT opens a data set, the locator contacts all Servers in the network and asks if they manage the data set. In the case where more than one instance of the data set is available, the locator will pick one.

Data Access Agent Component

The Data Access Unit (DAU) Server may either access the raw data directly, or it may work through an intermediate Data Access Agent (DAA). If the format of the data is simple (e.g., an ASCII tab-delimited file) then it may be expedient for the Server to use whatever local operating system interface is available to read the file. However, most telemetry streams have complex structures that require specialized input routines to extract selected variables, and data analysis tools often exist that can read the data. In other cases, the data may not reside in a file at all, but is instead stored in a relational database. When the DAU Server cannot effectively access the data, it instead uses the DAA to broker the data. This approach allows the DAU to leverage the capabilities of existing tools while making the data they manage available to a wider audience. The interface between the DAA and DAU Server is private, and is tailored to make the transfer of control information and data to be as efficient as possible.

A SAMPLE DAU APPLICATION

At the time of this writing, an initial implementation of the Data Access Unit is underway. The work was undertaken as a component of the Failure Analysis Expert System (FAES) [2] project. A basic requirement is that FAES be written in portable Common Lisp and be deployed on a Sun SPARCStation-II (UNIX) platform, while the data resides on a DEC MICROVAX-II (VAX/VMS) system. The expert system requires data on several signals, and each signal has between 10,000 and 60,000 samples. (Recall that a sample is a timestamp-value pair, or 8 bytes of information; 60,000 samples is roughly half a megabyte.)

Figure 4 illustrates how the abstract components of the DAU (Figure 3) can be realized in this instance. The Data Access Agent (DAA) shown here is BBN/Probe, a commercially available data analysis package. (The fielded version of this system will use Navy DataProbe, an ancestor of BBN/Probe.) BBN/Probe uses a data dictionary to name the variables in the telemetry stream and to describe the format of the data. Two different telemetry streams are analyzed, each with its own data dictionary. The Map component hides the differences between the names of similar variables in the data dictionaries.

The interface between BBN/Probe and the DAU Server uses one VAX/VMS mailbox (a basic interprocess communication service) to send commands to BBN/Probe and a second mailbox to capture the command output. The output is parsed by the Server to determine the command result. Potentially large amounts of data are transferred as the result of each command. To make the transfer of data as fast as possible, we take

advantage of BBN/Probe's ability to call an external function (a user-written subroutine) to move the data into shared memory buffers. The interface between BBN/Probe and DAU Server requires no modification of BBN/Probe.

The Distributed Computing Environment (DCE) component is Cronus [3], which accomplishes all network access and data translation services transparently to both Client and Server. Cronus also handles a large portion of the Locator component in that it automatically locates the DAU Server in the network. Cronus presents a Remote

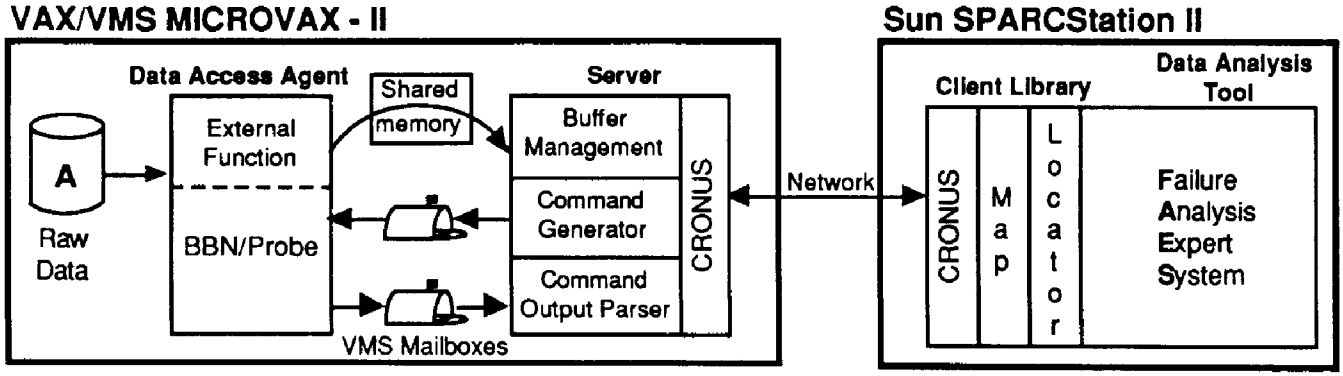

Figure 4. A Data Access Unit Implementation. This diagram shows a sample implementation of the DAU Architecture shown in Figure 3. The Data Analysis Tool in this case is a failure analysis expert system written in Common Lisp and running on a Sun SPARCstation-II. The DAU Client library is written in C, and uses the Cronus distributed computing environment to communicate with the DAU Server. The Data Access Agent is (DAA) BBN/Probe, a commercially available data analysis package. The DAU Server uses VAX/VMS mailboxes to pass control and status information, while data is transferred as quickly as possible using BBN/Probes's external function capability to write directly to shared memory.

Procedure Call (RPC) subroutine library interface to the Client and Server. Both Client and Server are designed so that other DCE products could be used in place of Cronus without requiring extensive modifications to either.

The DAU Client Library is is written in portable C, and can run on a variety of hardware and software platforms. The interface to the DAU Client by FAES (written in Allegro CL) is through Allegro's foreign-function package. Data Analysis Tools written in C and FORTRAN may call the DAU library routines directly.

CONCLUSION

The Data Access Unit (DAU) bridges the gap between data and analysis tools in a distributed network of heterogeneous machines. Data analysts no longer have to worry

about data format or location to get their work done. Time-to-solution should decrease, as the analyst does not have to wait for support staff to transfer and translate the data. Managing the data also becomes easier, as only a single copy is necessary for all users in the network. The DAU architecture is flexible enough to accommodate a wide range of applications. Figure 5 illustrates a range of possibilities.

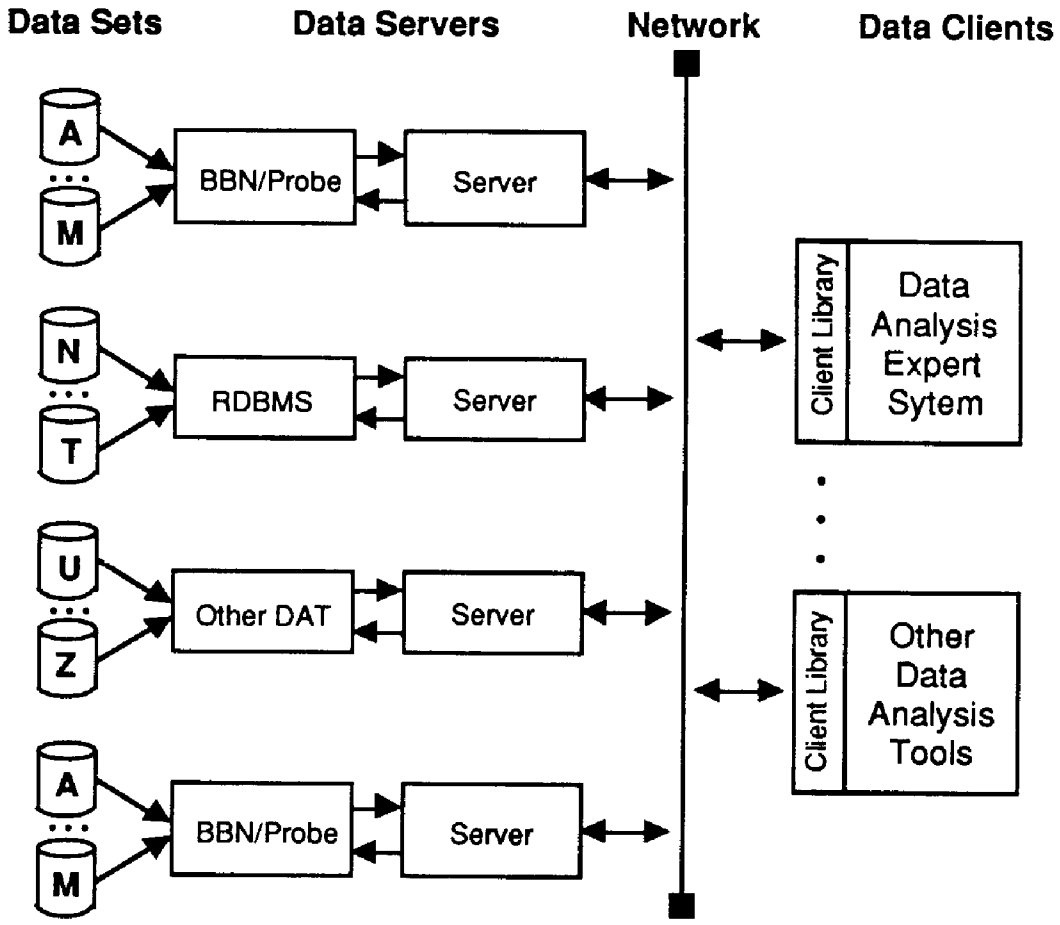

Figure 5. Distributed Data Clients And Servers. With the Data Access Unit (DAU), machine and network boundaries disappear, as clients of the DAU get transparent access to data anywhere in the network. The clean separation of data server from data client allows a wide variety of client and server implementations. Data and the data-processing capabilities provided by existing data analysis tools (such as BBN/Probe) and data repositories (like a relational database) can be exported to new applications by interfacing them to a DAU Server.

Notice that Figure 5 shows a second instance of data sets A through M being served by a second DAU server. With redundant servers and data, users enjoy more reliable and faster access, since if one server becomes inaccessible or overloaded the other can take over. (The choice of which Server to use is handled transparently by the DAU Client when the data set is initially opened.)

It is becoming more common for users to store their time-series data in commercial relational database systems. These and other data repositories can export their data and services across the network by being interfaced to a Data Access Unit Server. Since any DAU Client can work with any DAU Server, each new Client or Server that is developed only increases the value of all the data and analysis tools.

ACKNOWLEDGMENTS

Portions of this work were supported by the Naval Undersea Warfare Center under contract N66604-91-D-4033.

REFERENCES

[1] Birrell, A. D., Nelson, B. J., "Implementing Remote Procedure Calls", Transactions on Computer Systems, ACM, February 1984

[2] Delatizky, J., Morrill, J., Lynch, T. J. III, Haberl, K., "Expert Analysis Of Telemetry Data", Proceedings of the International Telemetering Conference, Volume XXVII, 1991

[3] Vinter, Dr. S. T., "Integrated Distributed Computing Using Heterogeneous Systems", Signal, June 1989