INTEGRATED COMPONENT-BASED DATA ACQUISITION SYSTEMS FOR AEROSPACE TEST FACILITIES

Richard W. Ross
NASA Langley Research Center
Hampton, VA 23681-2199

Abstract

The Multi-Instrument Integrated Data Acquisition System (MIIDAS), developed by the NASA Langley Research Center, uses commercial off the shelf (COTS) products, integrated with custom software, to provide a broad range of capabilities at a low cost throughout the system's entire life cycle. MIIDAS combines data acquisition capabilities with online and post-test data reduction computations. COTS products lower purchase and maintenance costs by reducing the level of effort required to meet system requirements. Object-oriented methods are used to enhance modularity, encourage reusability, and to promote adaptability, reducing software development costs. Using only COTS products and custom software supported on multiple platforms reduces the cost of porting the system to other platforms.

The post-test data reduction capabilities of MIIDAS have been installed at four aerospace testing facilities at NASA Langley Research Center. The systems installed at these facilities provide a common user interface, reducing the training time required for personnel that work across multiple facilities.

The techniques employed by MIIDAS enable NASA to build a system with a lower initial purchase price and reduced sustaining maintenance costs. With MIIDAS, NASA has built a highly flexible next generation data acquisition and reduction system for aerospace test facilities that meets customer expectations.

Introduction – Building Better Data Systems

Developing real-time data acquisition and reduction systems is an inherently difficult and complex job. Building a *good* system is even more difficult.[1] So what is a good data acquisition system? Most users would associate a good system with one that possesses these attributes: fully functional, flexible, easy to use, reliable, expandable to meet future requirements, and, of course, low cost. The NASA Langley Research Center in Hampton, Virginia, U.S.A., has developed a next generation data acquisition and reduction system called MIIDAS (Multi-Instrument Integrated Data Acquisition System) that meets these criteria.

MIIDAS provides integrated data acquisition and data reduction support, eliminating the need for purchasing or developing two separate systems that perform distinct, but closely interrelated, functions. Integrating the data acquisition and reduction systems allows the same features to be used for both products through the use of a common user interface. This integrated approach ensures compatibility between these two products and allows the user the flexibility to decide what is to be performed in real-time versus post-test, based on the operational requirements of the aerospace test facility, not the limitations of the various systems. For example, MIIDAS provides the capability of generating several types of plots, reports, and export files; the user has the flexibility to choose whether these are generated in real-time, offline after the test is completed, or both.

For future expandability, MIIDAS provides a generic instrument framework that allows future data acquisition hardware instruments to be easily added without requiring significant modifications to existing portions of the software. This framework provides consistent, common interfaces (such as data file formats, instrument configuration interfaces, and user interfaces) and standardizes the overall architecture of the system across all types of data acquisition hardware.

Standardization of the data acquisition and data reduction software reduces the training effort required by personnel that must support multiple test facilities. Users find the software is easy to use because no additional effort is required to learn a new product, and all of the developers' efforts can be devoted to enhancing the usability of a single product, rather than being diverted to support multiple products.

Software developers' support efforts are further reduced by the use of software components, or "building blocks," enabling systems with new capabilities to be easily developed from existing functional components. As a result, a significant cost savings can be realized.

Using commercial off the shelf (COTS) products reduces both the initial development and the continuing maintenance costs by purchasing, rather than developing, software. The costs incurred by the manufacturers of COTS software can be amortized over their entire user base, not absorbed by a single user (or limited number of users) as is required for custom-developed software.

Designing MIIDAS to function on multiple platforms greatly reduces the need to rewrite the software in the future. As market fluctuations and public opinion determine or influence what hardware and software platforms are the systems of choice, the software can easily adapt to these changes without extensive software redesign or modification.

MIIDAS takes advantage of the object-oriented design and programming capabilities offered by computer programming languages such as C++ and Java to isolate software from inadvertent "side effects" caused by seemingly unrelated functions as can be observed in traditional procedural programming languages such as FORTRAN.[2]

The combination of these techniques enables MIIDAS to achieve the design goals of being a low-cost data acquisition system supporting future expansion capabilities. Each of these techniques is described below in further detail.

Integrated Architecture Ensures Compatibility

There are currently many commercially available off-the shelf data acquisition systems that provide the ability to acquire and log data and view these data in near real-time. Frequently, these DAS software products are designed around the specific characteristics of a particular vendor's line of data acquisition hardware. These commercially available DAS products provide very good support for selected data acquisition hardware, but computations and data analysis are left to the end user.

There are also many off-the-shelf computational and data analysis tools to allow the users to perform these operations. By combining a data acquisition system from one supplier with a data reduction and analysis system from another supplier, users can acquire, reduce, and analyze test data, provided that these two systems are compatible with each other.

Using two separate products, one for data acquisition and another for computations and analysis, requires that the user learn how to operate two independent systems. The output of the DAS must be compatible with the data reduction system, often requiring additional effort to export data from the DAS into a format that can be imported into the data reduction software. Frequently, some capabilities may be present in one product, but absent or limited in the other product. End users are often frustrated when an operation can easily be performed online during test execution, but not offline during post-test processing (or vice versa).

One of the two primary design goals for the MIIDAS software is to combine the capabilities of an online data acquisition system with the offline computational and analysis capabilities of a data reduction system. MIIDAS integrates both the data acquisition and data reduction functionality into a single system using a common user interface and common file formats. Combining these two capabilities into a single product increases the flexibility of the users by allowing them to perform complex data analysis operations during the execution of a test, or to play back previously recorded data to simulate the execution of a test.

Figure 1 depicts the MIIDAS architecture, which consists of two major subsystems: the Real-Time Subsystem and the Offline Subsystem. Each of these subsystems consists of components, or building blocks. The Acquisition Component consists of the data acquisition software and a data server to collect the raw data, convert it to engineering units (EUs), and record the acquired data to a raw data file. The Computation Component performs additional complex computations on the acquired data, serves the data to display workstations and external computer systems, and stores the data in a relational database. The User Interface retrieves the data from the server and from the database, displays the data on the workstations, and manages user input requests and transmits them to the server for processing. All of these activities are supported by a set of Core Libraries that allow the components to communicate with each other.

The Offline Subsystem consists of a Playback Component that reprocesses previously acquired data from the raw data file. The data can be displayed as if they were being acquired in real-time, or can be reprocessed using different computations and corrections. The Legacy Data Interface component provides backward-compatibility with raw data files recorded by other NASA Langley data systems. The Offline Subsystem also uses the Computation Component, User Interface, and Core Libraries used by the Real-Time Subsystem. Providing a common user interface, computations, and support library that is shared between the online and offline software increases the users' flexibility for acquiring and processing test data.

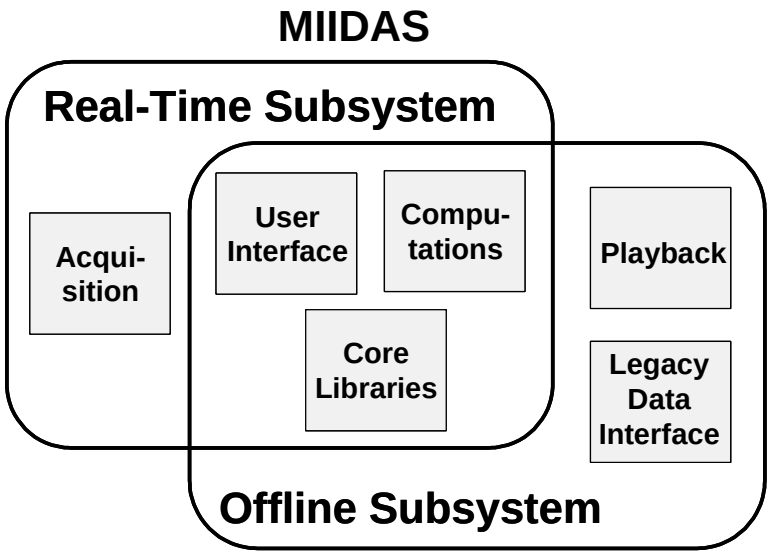

Figure 1. MIIDAS Integrated Architecture Ensures Compatibility and Increases Flexibility.

Instrument Framework Increases Expandability

As previously described, many data acquisition systems are designed to support specific data acquisition hardware. Such a design makes it difficult and expensive to expand the capabilities of the DAS to support other types of instruments, especially if the new instruments are significantly different than the types of devices currently supported.

There is great diversity among families of data acquisition hardware. Some instruments are highly sophisticated and can be programmed to automatically perform engineering unit conversions and perform several types of corrections and compensations. These types of devices can be configured to specify parameters for each channel, such as amplifier gain settings, filter types and cutoff frequencies, and scan rates. Some instruments can automatically perform self-tests, initiate calibrations, and can be interrogated to obtain card types and other parameters.

By contrast, other devices (especially older instruments) are less configurable and support fewer features. These types of devices may not support data buffering, automatic calibrations, or setting channel parameters. Frequently, one manufacturer will provide similar hardware interfaces among all instruments within their product line, but another manufacturer will require a very different interface.

Clearly, the software that communicates with the hardware needs to be different for each type of device to support the capabilities and communications protocols used by new hardware products. In a well-designed system, the fundamental structure of the software needs to be flexible enough to support these radically different types of hardware, across multiple vendors and with different levels of programmability.

However, if a data acquisition system was not designed to be expandable to accommodate future types of instruments, or to interface with hardware from multiple vendors, then all the supporting software must also be changed if these new instruments are to be supported. Implementing these changes can be very time-consuming and expensive.

MIIDAS currently supports hardware interfaces for several data acquisition units manufactured by Neff Instrument Corporation, including the Neff System 620 Series 500 Measurement and Control I/O System, Neff System 620 Series 600 Amplifier/Multiplexer, and the System 470 Data Acquisition System. MIIDAS also supports Electronically Scanned Pressure (ESP) measurement systems from Pressure Systems, Inc. MIIDAS is designed to support the addition of future instrument types by using a generic instrument framework. This framework provides a hierarchical structure that defines a generic “device,” which is simply an instrument that can be connected to a computer. Next, a device interface is specified, which

identifies the physical hardware interface to be used (TCP/IP, IEEE-488, RS-232-C, etc.).

Each instrument is classified according to its type and usage: controller unit, input unit, or output unit. Controller units support communication between the computer and instrument. Controller units are configurable and programmable; these are the units that control the operation configuration of the input and output units.

Input units are used to acquire data from channels on the instrument. Signals from transducers are digitized and transmitted from the instrument to the computer. Input units may be buffered to allow high data acquisition rates without requiring constant interaction with the host computer, or may be simple, unbuffered devices that are directly scanned and controlled by the computer.

Output units are similar to input units, except that digitized data are transmitted from the host to the instrument. These digitized data are converted to analog voltages and are used for controlling devices connected to the instrument.

The format of the data received from input units varies from instrument to instrument, which requires that all software that processes the received data support each type of instrument. The supporting software that would be affected includes all the internal data structures, the format of the recorded data, all programs that process the data in real-time, the post-test processing software that reads the recorded data file, plus utility and testing software used to interrogate and validate the integrity of the recorded data files. This approach is not practical, since it requires extensive changes whenever a new data acquisition unit is added to the DAS.

One solution is to convert the data from each manufacturer's format into a new format that is used internally by the DAS for all processing. Earlier versions of the MIIDAS software used this approach, but it had very high processing overhead due to the amount of copying and reformatting of the data that was required. As a result, this approach is not practical at high acquisition rates. What was needed was a solution that allowed the data to be interpreted and processed in its native format, without recopying or reformatting.

Despite the variations in protocols, MIIDAS accommodates these disparate data formats by adding a generic header to the acquired data. The header allows the data to be treated as a matrix, and the header specifies the number of rows and columns in the matrix, the data format (such as floating point, signed integer, unsigned integer, or text), and offsets that define the beginning of the matrix. The header also contains the information needed to convert the raw data values to millivolts by defining the full range of the instrument. Using this header, all of the supporting software can process data from various sources, in diverse formats, in a generic and consistent manner.

Standardization Promotes Ease of Use

As of early 2001, there were three major data reduction systems in use at the NASA Langley Research Center, and each of these systems had several variants used at different aerospace facilities. Langley users were frequently frustrated by the lack of standardization and its associated problems. Different programs produced varying results, and users didn't know which was the "right" answer. Maintenance costs were extremely high, since many different systems and versions needed to be maintained.

Worst of all, data reduction system users that were transferred to another aerospace facility, either temporarily or permanently, needed to be retrained in the operation of the new systems. These users had already invested a significant effort to become familiar with the operation of one system, and then had to learn another system at a different facility. As a result, the users often expressed feelings of frustration and being overworked.

By contrast, the data acquisition systems were much more standardized. Only a single product existed at the major LaRC aerospace test facilities, although there was still a different version of the DAS software in use at each of the facilities.[3] A similar solution was needed for the data reduction systems that would replace the existing three disparate data reduction systems with a single, standardized system common to all facilities. By increasing the degree of standardization, the ease of use is elevated due to the elimination of the need for retraining.

The data reduction capabilities of MIIDAS were developed to overcome these difficulties by employing three different strategies. The first strategy, using an integrated architecture to provide a common interface between the data acquisition and data reduction software, has already been described above. By integrating the data acquisition and the data reduction systems, the users can operate more efficiently due to the common interface between the two products.

The second strategy employed is to replace the different data reduction products currently in use at the various Langley facilities with a single, standardized system.

By reducing the number of diverse data reduction systems, maintenance costs have been significantly lowered, and the maintenance personnel can concentrate on a single product, instead of the workforce being diluted by supporting several products concurrently.

The third strategy is to develop a single, standardized version of the software that meets the needs of all users of the system. This strategy is an extension of the previous strategy: focus the efforts into developing and maintaining a single version. By maintaining only one version of a single product, the support staff can provide a higher level of service, including both greater in-depth knowledge and reduced response times, due to the increased number of available and knowledgeable personnel.

Components Lower Software Complexity

Data acquisition and reduction systems are inherently complex, due to the need for extensive functionality and the high degree of reliability that is required. Generally, the complexity of a software system increases rapidly with the degree of functionality. Since development and maintenance costs are directly related to the complexity of the system, reducing complexity is an effective method for lowering costs.[4]

While the growth in complexity due to the increase of functionality is unavoidable, it can be mitigated through the use of software components. A software component is a collection of software modules that perform a specific set of functions and have a well-defined interface. In this sense, a software component is comparable to an integrated circuit chip: it performs a particular function, or set of functions, and has a clearly defined interface defined by its pin-out specifications.

Similarly, a software component can be defined by identifying the requirements that a component will fulfill, and by delineating the communication protocol between the component and other portions of the software system. To effectively lower the overall complexity of a software system, the components must be capable of enforcing the interface requirements by disallowing any access to data or functions other than through the defined interface.

MIIDAS uses several software components that have already been previously described. Each of these components consists of an interface manager, or server, that controls the communication between the component and its clients. In some cases, MIIDAS uses multiple interface managers to control access to data by different types of clients.

One example of the use of multiple interfaces is with the Computation Component, which provides one set of services to the Display Component, an integral part of the MIIDAS system. The Computation Component also provides another interface to external computer systems. This interface supplies a much more limited set of capabilities to these external systems to prevent inadvertent and inappropriate actions from being initiated by these systems.

Off the Shelf Products Decrease Cost

In addition to reducing the complexity of the software as an effective method for lowering costs, the volume of the software can be reduced by purchasing third-party, commercial off the shelf (COTS) software products.[5] COTS products usually provide functionality that equals or exceeds that of custom-developed software, since a larger staff of skilled developers is available. Purchasing COTS software is usually less expensive than maintaining custom-developed software because the development and maintenance expenses for COTS products can be shared among users of these products.

MIIDAS incorporates several COTS products as part of the standard system, and provides support for other optional COTS products. Microsoft® Office Professional includes two products that MIIDAS uses extensively: Microsoft Access and Microsoft Excel. Access is a low-cost relational database that supports database standards such as Structured Query Language (SQL) and Open Data Base Connectivity (ODBC) protocols. MIIDAS currently uses Access to provide database services, although any SQL- and ODBC-compliant database could be substituted. The database is used to store reduced test data for later retrieval and presentation. By using a relational database, MIIDAS can perform queries based on user-specified conditions, generate reports, and export data to other programs for further analysis.

One of these analysis programs that MIIDAS uses is Microsoft Excel. Access can directly transfer data from a database to an Excel spreadsheet without the need to export data from one program and import data into the other. Once the data are in the spreadsheet, the user can perform additional computations on the data, display the data using plots or reports, or compare the data to historical or theoretical data.

MIIDAS also uses MATLAB®, by The MathWorks, Inc., as an optional tool for more complex analysis and plotting capabilities. MATLAB is a high-performance

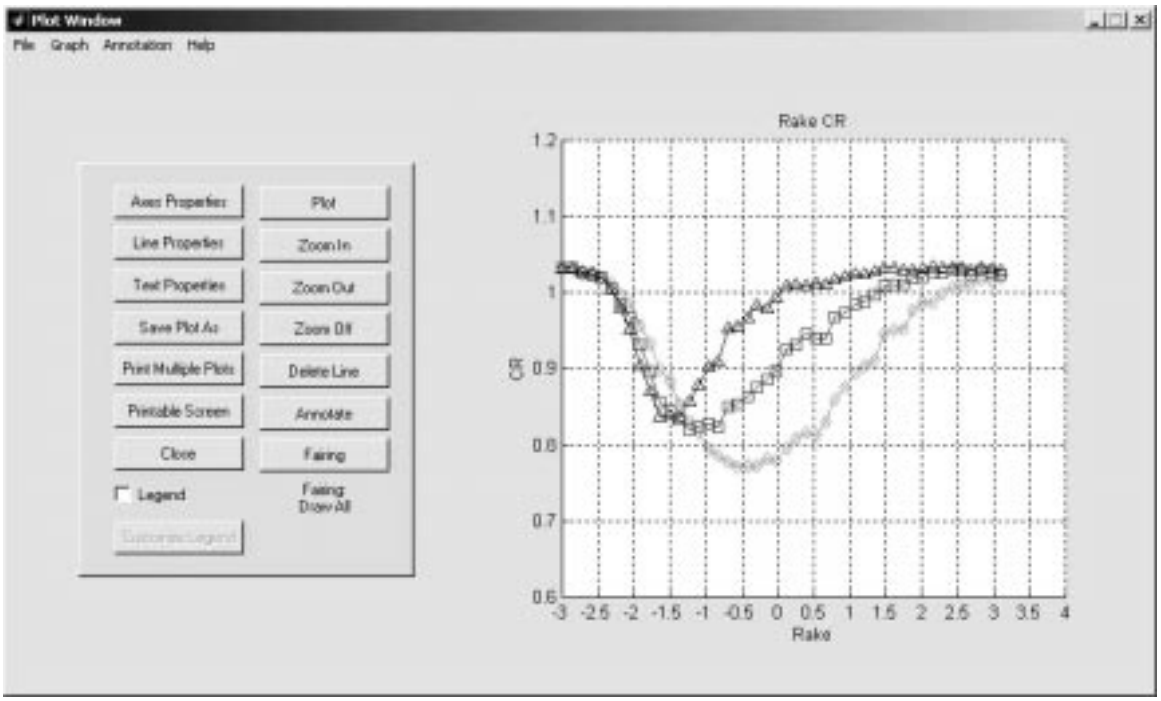

Figure 2. Commercial Products Increase Functionality and Lower Costs.

language for technical computing. It provides an extensive library of advanced mathematical functions and can be operated interactively or through a scripting language. The execution of these scripts can be performed under program control as well, allowing the DAS software to transmit current data to a MATLAB script and retrieve the results up to ten times per second. Used in this manner, MATLAB provides the capability for a user to develop custom software, using a high-level scripting language that can be tightly integrated with other real-time and offline computations.

MATLAB also provides the capability to plot data, as well as the ability to develop user interfaces. MIIDAS uses these features to present acquired data using two-dimensional plots, with user-configurable parameters such as plot scales and colors. MATLAB can also generate and manipulate three-dimensional graphical representations of objects, allowing test articles to be depicted in 3-D based on data acquired by MIIDAS. Figure 2 shows a user-defined plot generated by the MATLAB software.

Platform-Independence Promotes Expandability

One of the design goals of MIIDAS was to provide full functionality with a low-cost data acquisition system. To keep the cost of these systems low, MIIDAS was originally designed to run on Intel® Pentium-based (x86) personal computers (PCs) running Windows® 2000. However, some applications require a system with the greater horsepower. NASA Langley currently has many data acquisition systems using Compaq (formerly Digital Equipment Corporation) Alpha-Servers® running Tru64® UNIX. Although these systems are much more expensive, they offer high levels of performance and have already been purchased. Current Alpha users would like to be able to upgrade their data acquisition software without the need to replace their existing hardware.

MIIDAS is currently being redesigned to provide support for multiple platforms, including Windows 2000/NT on x86-based hardware as well as several variants of UNIX, including Tru64, Linux, and Solaris. To provide this capability, MIIDAS uses COTS products that are readily available on all of these platforms. For software developed in-house, all MIIDAS software is either designed to be platform-independent, using only industry-standard program-

ming language capabilities and constructs, or is designed to isolate any platform-specific code from the platform-independent software.

Object-Oriented Design Reduces Maintenance

While the design and development of new software projects is expensive, the maintenance of software products is even more costly. Software maintenance typically consumes about 60% or more of the total expenditure over the total life cycle of a software system, while only 40% of the total cost is consumed by the initial phases of a software project.

These initial phases include the definition of the requirements, top-level design, detailed design, implementation, unit testing, system testing, integrated testing, installation, acceptance testing, user training, and documentation. These costs are incurred up front and are highly visible, yet on some projects the initial design and development costs may represent as little as 20% of the total cost of developing a data acquisition system.[6] The remaining 80% is spread out over time and is often unanticipated, yet the maintenance costs can far exceed the original development costs, especially for products with long life-cycles.

MIIDAS, as previously described, is designed to allow the software to be easily adaptable to both new types of data acquisition instruments and to new computer hardware and operating system software platforms. Thus, MIIDAS is expected to exhibit a long software life cycle, and the design of MIIDAS is based on this assumption and on its long-term goals. Frequently, software products are designed to minimize development costs, yet both developers and customers discover later that maintenance costs vastly outrun the initial development costs.

Even users who are not familiar with software development and its associated life-cycle process can easily relate to the same relationship between initial cost and maintenance costs with commercial products. Many top-end commercial software products charge an annual maintenance fee of approximately 20% of the original purchase price to provide telephone support, patches to solve immediate problems, and full product releases with new capabilities. Many "low-cost" ink-jet printers sell for low purchase prices, but replacement ink cartridges are very expensive and must be replaced frequently. As a result, the cost of the ink for a heavily used printer can easily exceed the initial cost of the printer, even in the first year of ownership!

Software maintenance efforts, and their associated costs, can be classified as corrective (fixing software bugs), perfective (adding new capabilities to software), adaptive (changing fully functional software to work in a new environment), or emergency (modifying software to allow the continued operation of the system).[7] Modern programming languages such as C++ and Java use object-oriented technology to reduce these maintenance efforts and their associated costs.

MIIDAS uses C++ to provide the core functionality of the data acquisition and reduction systems, and uses Java applications (stand-alone, not web browser hosted) for the user interface. Java is used to provide a platform-independence environment for user interface development, resulting in lower development costs.

Both Java and C++ support encapsulation, which isolates portions of the code, including both data and functions, from other parts of the system that do not need to interact with these functions. Encapsulation allows portions of a complex system to be developed without concerns for the interaction with other portions of the system.[8] Developing a system using encapsulation may increase the initial development costs by a small amount, but result in greatly reduced maintenance costs, depending on the expected lifetime of the product.

MIIDAS also utilizes other object-oriented techniques such as inheritance and polymorphism. Inheritance allows one class and its objects to acquire, and even change, some or all of the properties of the original class without duplicating the source code. This technique minimizes the development effort by allowing useful code to be reused, rather than copied. Copying code, then making some slight variation to that code, means that there are now two versions that must be maintained; if a bug is found in the logic of the original code, it must now be fixed in two places! Using inheritance, the correction is made once, and the change is automatically propagated to the inheriting classes without additional code modifications. Polymorphism, meaning "many forms," enables software functions with the same name to perform different operations, depending on how the software is used.

Future Enhancements Will Expand Capabilities

MIIDAS, as it exists today, provides many immediate benefits: decreased complexity, increased standardization, greater flexibility, lower life-cycle costs, and greater expandability. Planned future enhancements promise to build on these strengths while providing greater functionality.

One major enhancement is the addition of an algorithm to automatically assess the quality of acquired data using statistical methods. This planned approach would analyze and evaluate the data using a combination of data validation criteria specified by the user and preprogrammed constraints determined by the limitations and characteristics of the data acquisition hardware and transducers.

Other enhancements include upgrading to higher-performance databases, such as Oracle® or Informix®, and optimizing the user interface software to maximize ease of use.

Summary – Simpler is Better

Building a complex data acquisition system is a challenging task, but developing a simpler system takes even more effort. At first, this statement may seem paradoxical, but when the full life-cycle costs are considered, it becomes apparent that simplicity, and hence low total cost of ownership, doesn't just happen by accident. Building a high quality system that is easily maintained at a low cost requires extensive up-front planning and design.

MIIDAS is based on the assumption that simpler is better (meaning greater functionality at a lower cost). MIIDAS consists of a data acquisition system integrated with a data reduction system to provide commonality and increase the flexibility of the total system. The system is expandable by providing a generic instrument framework that can be easily customized to support the addition of new data acquisition hardware.

By developing a system that can be applied to a wide range of testing needs, users are not required to learn the operation of new software programs when they are temporarily or permanently assigned to other facilities. The result is of benefit to the user, since the user does not need to learn another software product, and the cost to the organization is reduced.

The use of software components reduces the complexity of the system, resulting in lower initial development and purchase costs as well as reduced continuing maintenance costs. Commercial off the shelf products further lower these costs by reducing the total volume of custom software development required.

Developing software that is not targeted to a specific hardware or operating system platform reduces long-term adaptive maintenance costs by enabling the useful life of the software product to exceed the hardware lifespan. Object-oriented design and programming techniques further reduce the maintenance costs by maximizing reuse of the software and isolating programs and data.

Expanding the future capabilities of MIIDAS will allow even greater enhancements, such as the ability to support high-performance databases, to expand support to other hardware, and to assess data quality in real-time. These future capabilities can be easily incorporated into the existing software because of use of software components and the generic instrument framework.

Because of the expandability that comes from platform independence and the generic instrument framework, MIIDAS has broad applicability to a wide variety of applications. MIIDAS promotes flexibility in the workforce by enabling personnel to perform a wide variety of operations with a single, integrated product. This product can accommodate current as well as future needs for a test facility, reducing the maintenance and upgrade costs that would otherwise be required.

MIIDAS was designed to minimize the total life-cycle cost using the techniques described. This effort resulted in a program that is less complex, promotes reusability of software components, and is easily maintainable. In other words, building a simpler data acquisition system results in a better system overall.

References

1). Douglass, B. P., 2000, *Real-Time UML: Developing Efficient Objects for Embedded Systems*, Addison-Wesley, pp. 2-12.

2). Glass, R., L., 1992, *Building Quality Software*, Prentice-Hall, pp. 311-312.

3). Batts, F. E., 1998, "The Evolution of Wind Tunnel Data Acquisition Systems at NASA Langley Research Center," *Proceedings of the 44th International Instrumentation Symposium*, Reno, NV, pp. 1-6.

4). Weinberg, G., 1992, *Quality Software Management: Systems Thinking*, Dorsett House, pp. 135-139.

5). Boehm, B. W., 1981, *Software Engineering Economics*, Prentice-Hall, pp. 311-312.

6). Martin, J., McClure, C., 1983, *Software Maintenance: The Problems and Its Solutions*, Prentice-Hall, pp. 84-87.

7). Software Engineering Standards Committee of the IEEE Computer Society, 1998, "IEEE Standard for Software Maintenance, IEEE Std 1219-1998," Institute of Electrical and Electronics Engineers, Inc., p. 5.

8). Lippman, S. B., Lajoie, J., 1998, *C++ Primer*, Addison-Wesley, p. 33.