The Distributed Data System of the Automated Manufacturing Research Facility of the National Bureau of Standards

Cita M. Furlani
Don Libes
Edward J. Barkmeyer
Mary J. Mitchell

Factory Automation Systems Division
Center for Manufacturing Engineering
National Bureau of Standards

Presented by:

Mohammad Khatib
COMPCON SPRING '88
Thirty-third IEEE Computer Society International Conference
San Francisco, California
February 29 - March 4, 1988

The Distributed Data System of the Automated Manufacturing Research Facility of the National Bureau of Standards

Cita M. Furlani
Don Libes
Edward J. Barkmeyer
Mary J. Mitchell

ABSTRACT: A major facility for manufacturing research exists at the National Bureau of Standards (NBS), the Automated Manufacturing Research Facility (AMRF). The AMRF has been designed as a "data driven" control system. This permits it to handle a broad range of parts for automated manufacturing but requires an effective interface between the data generated in a manufacturing system and the control modules that use the data. Data resources are physically distributed across a network of heterogeneous hardware and software systems acquired from numerous vendors. To meet these requirements a distributed data system, the Integrated Manufacturing Data Administration System (IMDAS), has been developed.

IMDAS is characterized by 1) a common interface to user programs, 2) a common interface to underlying databases, and 3) a hierarchical architecture, providing both centralized and distributed services. It is designed to provide the control systems of the AMRF access to the data necessary to support the design, planning, manufacturing, and inspection of parts. Research has focused on the identification and modeling of factory data and relationships. Control processes specify requests for data services in a common data manipulation language.

KEY WORDS: distributed, data administration, automated manufacturing, CIM, IMDAS, AMRF.

future components of small-batch manufacturing systems. Another is to provide a laboratory for the development of factory-floor metrology in an automated environment, developing new ways of making precisely machined parts. Commercially available products are used in the facility wherever possible, in order to expedite transfer of research results into the private sector.

To provide a real testbed for interface standards, the AMRF is intentionally composed of manufacturing and computing equipment from many vendors, thereby making its construction a major integration effort [1][2]. Shop floor equipment types include Computer Numerical Control (CNC) machines, a coordinate measuring machine, robots, a vision system, robot carts, automated storage & retrieval systems, cleaning and deburring devices, and part fixturing and robot gripper systems. The configuration is structured around several self-contained workstations, each capable of executing a well-defined set of manufacturing functions. Each workstation is able to operate either as an independent manufacturing unit under control of a local operator, or as an element of a multi-workstation manufacturing system under control of a higher-level process. A typical machining workstation consists of a CNC machine tool, a robot, a materials transfer station, and local buffer areas for tools and workpieces.

The intelligence structure of each workstation includes the Robot control system, the Machine Tool control system, sophisticated sensor systems and a Workstation control system to coordinate the activities. Above the workstation level, batch manufacturing coordinators, or Cells, and a floor manager, or Shop controller, provide higher levels of control. The highest level of control, the Facility controller, implements the "front office" functions that are typically found in small manufacturing facilities. All of these control and sensory processes are software systems, which reside on interconnected computer systems, making the AMRF a distributed computing network [3]. Many different computer languages and types of computer systems make up the computing environment of the AMRF. In addition to the shop floor activities, manufacturing data preparation activities, including part design, geometry modeling, group technology classification, process planning, and offline control programming, are performed on "engineering" computer systems linked into the factory floor network. These types of data together form the global shared database of the manufacturing facility (Figure 1).

In the AMRF, as well as in most automated factories, data resources are physically distributed across a network of heterogeneous hardware and software systems acquired from numerous vendors. Such a distributed system requires a method of transferring information which is fast, accurate,

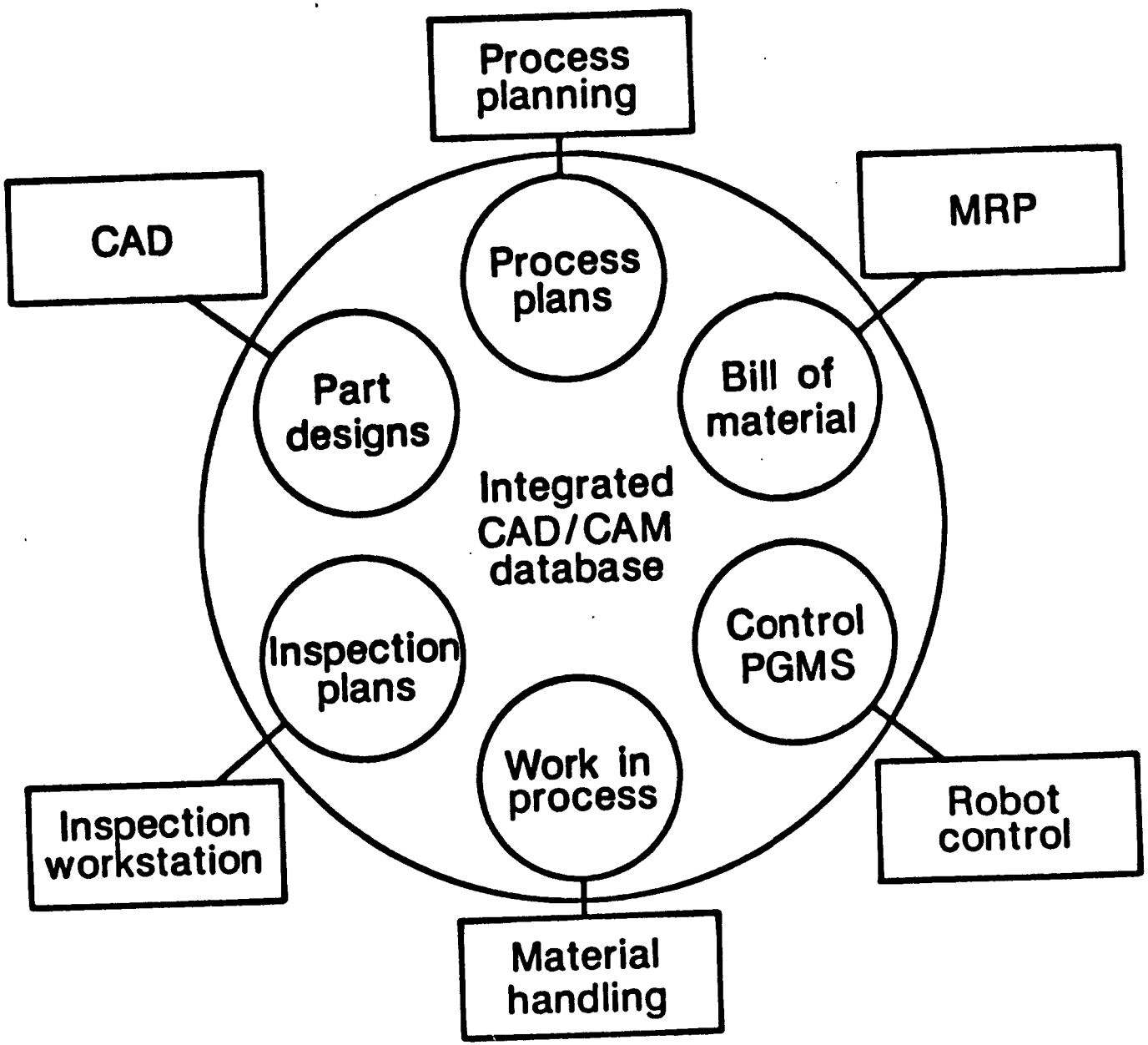

Figure 1. Global Shared Database

reliable, with consistent representation, and independent of the actual physical location of the machines.

The AMRF uses the concept of computer "mailboxes," areas of memory on various computers to which all of the machines in a particular group have access through the network communications system, subject to strict rules of protocol. Control processes can leave "messages" for each other and stop to read their own "mail" at opportune times without interrupting each other.

Currently, the AMRF communications network, the backbone of the distributed data system, uses an Applitek[1] broadband token bus and a combination of older computer communications protocols, including RS232 and Ethernet systems. Work is underway to upgrade the AMRF network to one based on the principles of the Manufacturing Automation Protocol (MAP) network proposed by General Motors and others.

Computer integrated manufacturing (CIM) refers to the integration of diverse systems into an automated production complex closely coupled to the engineering and administrative systems that support and drive it. Rarely does the same kind of computer system perform engineering support, real-time control and administrative applications. These component systems range in data management capabilities from simple shared memory managers to special purpose software which optimize data access for a particular function to general purpose database management systems with a full range of supporting tools. The software system which integrates the diverse and distributed data resources of the factory environment must mask the differences inherent to these conditions. The critical element in such a complex is the ability of all the associated programs and users to share data. To quote: "Data--its generation, processing, storage and use in the implementation and control of the manufacturing enterprise--is the essence of CIM, the NBS AMRF, and the fully automated factory of the future." [4]

DATA IN A DISTRIBUTED ENVIRONMENT

A significant part of the AMRF distributed data system research has focused on the identification and modeling of factory data and relationships. A critical step in accomplishing integration in an environment of diverse applications is the definition of a common logical model of the shared information and meanings. In the CIM environment, almost all data is significant to more than one application, but the way in which it is best organized for each application area may be different. For example, Production wants to keep track of component inventories by PART-TYPE and LOCATION while Purchasing wants to organize the components by SUPPLIER, SHIPMENT and ORDER and Accounting wants to track them by PART-TYPE, INVOICE and PRODUCT. Naturally, having been organized for different applications they have different

schemas and often use entirely different database management systems. Because there are interactions of the data, it becomes necessary to integrate databases. This almost always requires an external mechanism to keep them consistent, to define inter-relationships and to describe their interaction.

Controllers retrieve and modify through views of data structured to meet the needs of the particular application. These logical views are relations whose objects and attributes may or may not coincide directly with the fields of a physical record. A distributed architecture demands that a data dictionary and directory system exist at each communications node to index and define the data sets that reside at that node. Data include the translation of logical names associated with data structures and elements and the definition of actual schemas or physical structures in terms of the local data management system. The information in the dictionary and directory system is active, in that data sets are created and deleted while the test bed is operating. In addition, a common conceptual model of the entire database complex must be maintained at some generally available location. This central model contains information on the distribution and the logical structure of the data sets, as well as the relationships between records in the data sets that span multiple nodes or are replicated at them.

The researchers within the AMRF have used existing data modeling methodologies for discovery, concept formation, and validation of information shared by the shop floor and data preparation activities. Individual components of the AMRF are modeled and then merged to develop an integrated data model. The techniques used, Information Analysis [5] and IDEF1X [6], emphasize the discovery of the meaning and relationships between information units, in addition to recording representation forms and deriving convenient storage structures.

Commercially available software tools have been used to support these modeling activities. Analysts use these products to describe discovered relationships and constraints in English sentences, such as: "a storage_device has one or more storage_areas associated with it", "a storage_device is uniquely identified by an equipment_id and unit_id", and "a tool is a type of reusable_resource_item". Then the control system engineers validate and refine the evolving integrated data model by deciding whether they agree or disagree with these statements.

The integration data model being used in the AMRF is SAM*, a semantic data model [7][8]. This approach was accepted over others because of the model's expressive power to represent arbitrarily complex objects and constraints. This model is the heart of the AMRF dictionary system which makes use of the available semantics to control the distribution of the data and its complex interrelationships and constraints.

MANAGING DATA IN A DISTRIBUTED ENVIRONMENT

As a "data driven" control system, the AMRF is capable of handling a broad range of parts for automated manufacturing. But an effective interface is required between the data generated in a manufacturing system and the control modules that use the data. Control processes and factory personnel must be able to specify requests for data services in an environmentally neutral form, a common data manipulation language (DML). The integrated data service provider must translate requests in this language into commands that can execute at the site or sites which manage the data resources. In addition, individual data units may have to be translated to resolve the representation differences across hardware boundaries. The integrated data services system may be required to assemble results from several sites and to perform user specified formatting of resultant data. Finally, a problem which is characteristic of, but not unique to, the manufacturing floor is real-time access to data.

So we see that a manufacturing facility comprises a large number of dissimilar computer systems, data systems and databases. The need for sharing data among these systems results in overlapping databases with representational inconsistencies. We have argued that we need to build a "common data system" to solve the integration problem. This system would make knowledge about shared data resources available for all applications to use. In addition, we assert that such a system must support real-time use. How does one go about constructing such a system? The simplest approach to development of a common data system is the centralized system: a single common database on a central computer system with communication paths to all client systems. The inherent simplicity of this architecture, particularly with regard to management and maintenance, makes it highly desirable if it can be made practical. With current computer technology, it is possible to create a central system with sufficient redundancy that a total failure is extremely unlikely. However, the ability of such a system to handle all of the database transactions for the whole manufacturing enterprise without bottlenecks or unacceptable delays is very questionable. In all but the smallest organizations, performance and cost considerations will dictate some distribution of function and data.

Another possibility is maintaining distributed overlapping databases with conversion of data between them. This is viable if the number of databases is small and the interactions between them are carefully timed and controlled. The number of conversion programs will grow proportionally to the square of the number of databases. Any two databases must be idle for a certain period of time in order for information to be transferred between them. Moreover, such databases will

normally be inconsistent and be brought into alignment only at the times of transfer between them.

The alternative of modifying programs to access data from multiple databases can be overwhelming. It avoids the consistency problems inherent in the database conversion approach, but it is simply impractical in an environment of numerous systems and regular changes.

By using a common interface between programs and databases, this complexity can be avoided. Each new program has only one interface to all data. A change to the architecture of one database requires modification of only its interface to the common service, not to any of the applications. It is our belief that most enterprises will benefit from a system of distributed databases with common interfaces. This will enable integration of new and existing data systems and relieve programs and programmers of network, access and conversion problems.

The remaining choice is whether to provide the common interface from a single server with interfaces to each of the distributed databases, or to distribute the common interface service over several systems with interfaces to each other. Again, the centrally controlled system is simpler and currently feasible, but its ability to handle all of the transactions, even when it can distribute the actual data manipulations, must be in doubt. It also acts as a single point of failure. On the other hand, the protocol problems that result from trying to do distributed data management by committee has been the subject of much academic discourse and few, if any, sound solutions.

INTEGRATED MANUFACTURING DATA ADMINISTRATION SYSTEM

Our approach to providing a common interface to distributed data is the Integrated Manufacturing Data Administration System (IMDAS)[9]. IMDAS is characterized by 1) a common interface to user programs, 2) a common interface to underlying databases, and 3) a hierarchical architecture, providing both centralized and distributed services.

Application, or control, programs communicate with IMDAS using a standard language, referencing data names from a common dictionary. The IMDAS Data Manipulation Language (DML) was closely modeled after ANSI standard SQL [10]. Extensions to SQL were made to allow the projection and selection of elements in complex objects. The DML also allows programs to specify files or memory buffers as the sources and destinations of data.

On the other side, IMDAS has a common interface to underlying data repositories, such as commercial database systems. This minimizes the work needed to incorporate new databases into the common data system while allowing existing systems to continue operating without change.

This set of common interfaces affords users (AMRF control processes) a generalized view of data access. They see data manipulation as operations on information units, not databases, and are not concerned with what system or machine has the data. The result is conceptually simple. The user sees a single common database managed by the IMDAS (Figure 2).

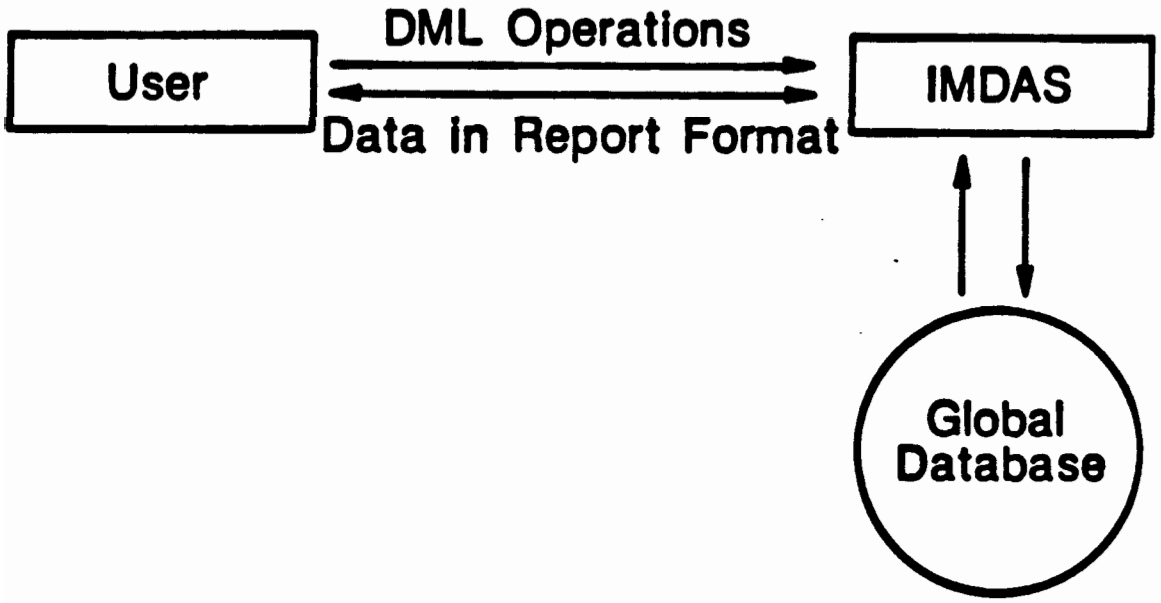

Figure 2. IMDAS Concept

The internal architecture of IMDAS is a 4-level hierarchy (Figure 3). The levels are distinguished primarily by scope of responsibility for data management. The higher the level, the more data is administered. At the bottom level of the IMDAS are the data repositories, some of which are commercial DataBase Management Systems (DBMSs). Also included are other repositories of sharable information, such as file systems, common memory, and locally developed, application specific, data managers.

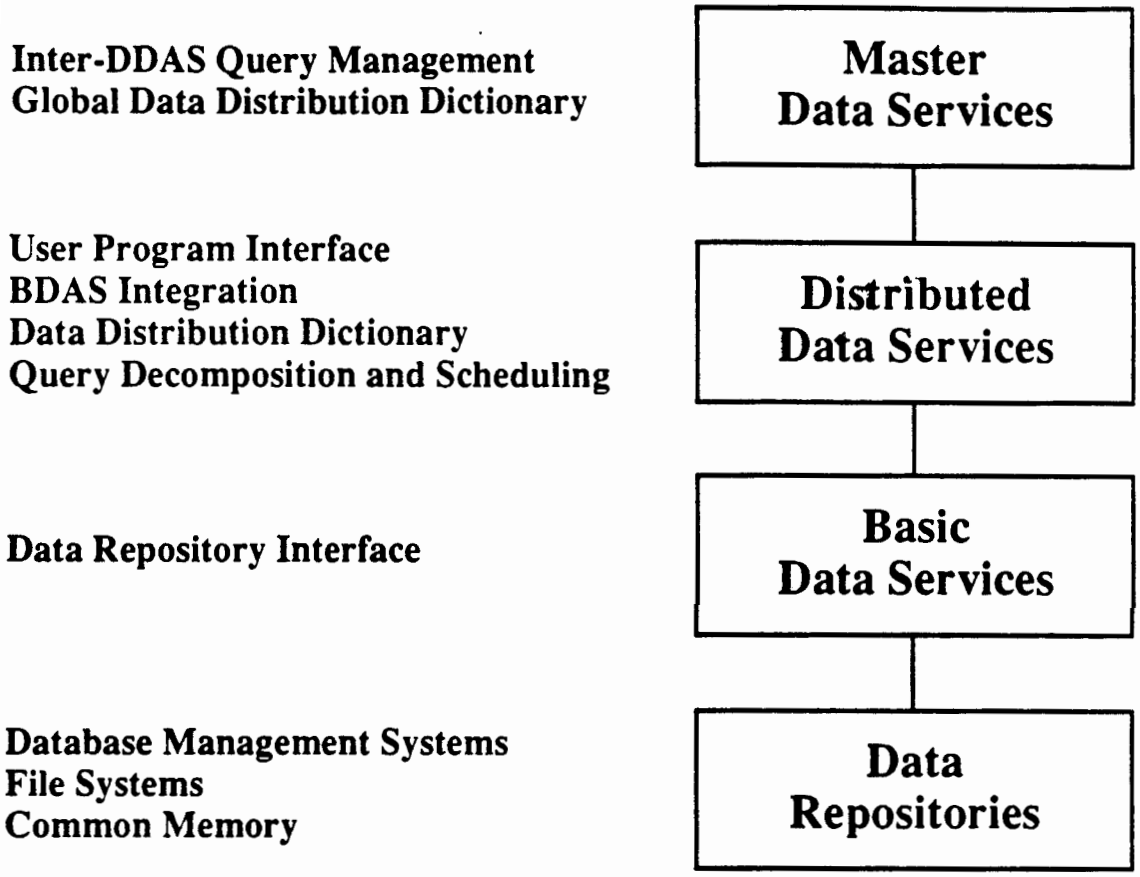

Figure 3. The IMDAS Hierarchy

BDAS - Basic Data Administration System

The Basic Data Administration System (BDAS) resides on each computer system in the AMRF. Each BDAS integrates its local data repositories into the IMDAS, and with its associated DBMSs, executes all manipulations on that data. The BDAS must convert the IMDAS internal form of a transaction to that accepted by the DBMS which has to execute it, pass it to the DBMS and interpret the status which comes back.

The BDAS must also convert any data involved between the DBMS dependent form and the IMDAS interchange form. Because capabilities and access techniques differ dramatically from DBMS to DBMS, and can be further complicated by local operating system conventions, the interface to a particular DBMS is encapsulated in a separate process called the Command Translator/Data Translator (CT/DT) for that DBMS (Figure 4). In addition, the BDAS must access local user data areas and convert between the declared user representation and the IMDAS interchange form.

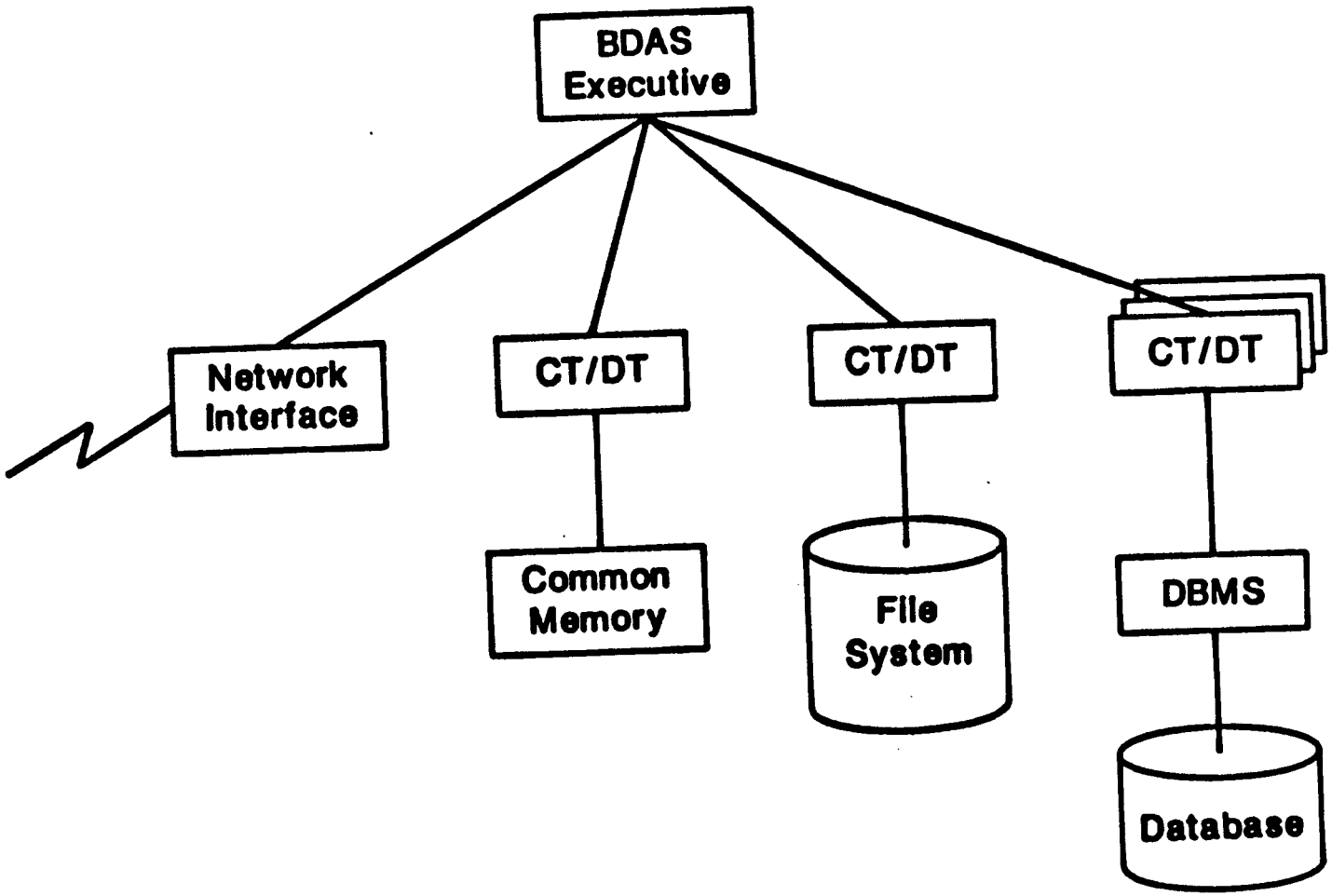

Figure 4. Typical BDAS

The BDAS receives commands from and returns status to the DDAS which supervises it, but it also deals with other BDASs as network peers for the purpose of delivering data. In this way, data moves directly between the user and the data repositories rather than following the IMDAS hierarchy. This feature is mandatory for achieving the performance required in a real-time environment.

DDAS - Distributed Data Administration System

At the Distributed Data Administration System (DDAS), the collection of data repositories managed by a group of BDASs is logically integrated into a segment of the global database, using a dictionary describing the distribution of the data. The DDAS becomes the data manager for that segment and supervises all manipulations on it. In addition, each DDAS provides the IMDAS interface to some set of the user programs, as a DDAS is available on each computer system that is capable of supporting its activities.

User programs issue transactions to the IMDAS, represented by the DDAS, which accepts them, oversees their execution and returns status to the user. User transactions are stated in the DML. These transactions are converted into an IMDAS standard internal form and then modified to reflect the differences between the user's external view and the IMDAS global conceptual view.

The DDAS attempts to map the transaction into a set of operations on elements of the global database which are managed by individual DBMSs. In order to do this, it consults a dictionary that describes how data is distributed over the integrated databases. The result is a set of tasks to be executed by specific DBMSs. If any of the data is outside the segment managed by this DDAS, the whole transaction is sent to the Master Data Administration System (MDAS).

MDAS - Master Data Administration System

In order to integrate segments managed by separate DDASs and to execute user transactions that require this level of integration, a single system is designated the Master Data Administration System (MDAS). The MDAS is an optional component of the IMDAS which is made necessary by having more than one DDAS. So, from our point of view, the issue of centralized versus distributed control of the distributed databases does not have to be resolved at the outset. If a single system can manage all data activity in an enterprise, one installs the sole DDAS there and makes its controlled segment the whole global database. But when more than one DDAS becomes necessary (as we expect must inevitably occur), rather than trying to solve the crossover problems by committee, we appoint a Master DAS. The MDAS supervises, in