516-60
150486
p. 10

N93-25577

THE DATABASE QUERY SUPPORT PROCESSOR (QSP)

Patrick K McCabe
Rome Laboratory
Rome, NY 13441

ABSTRACT

The number and diversity of databases available to users continues to increase dramatically. Currently, the trend is towards decentralized, client server architectures that (on the surface) are less expensive to acquire, operate and maintain than information architectures based on centralized, monolithic mainframes.

The database query support processor (QSP) effort evaluates the performance of a network level, heterogeneous database access capability. Air Force Material Command's Rome Laboratory has developed an approach, based on ANSI standard X3.138 - 1988, "The Information Resource Dictionary System (IRDS)" to seamless access to heterogeneous databases based on extensions to data dictionary technology.

To successfully query a decentralized information system users must know what data are available from which source, or have the knowledge and system privileges necessary to find out. Privacy and security considerations prohibit free and open access to every information system in every network. Even in completely open systems, time required to locate relevant data (in systems of any appreciable size) would be better spent analyzing the data, assuming the original question was not forgotten.

Extensions to data dictionary technology have the potential to more fully automate the search and retrieval for relevant data in a decentralized environment. Substantial amounts of time and money could be saved by not having to teach users what data resides in which systems and how to access each of those systems. Information describing data and how to get it could be removed from the application and placed in a dedicated repository where it belongs. The result: simplified applications that are less brittle and less expensive to build and maintain. Software technology providing the required functionality is off the shelf. The key difficulty is in defining the metadata required to support the process.

The database query support processor effort will provide quantitative data on the amount of effort required to implement an extended data dictionary at the network level, add new systems, adapt to changing user needs, and provide sound estimates on operations and maintenance costs and savings.

THE DATABASE QUERY SUPPORT PROCESSOR (QSP)

INTRODUCTION

The Database Query Support Processor (QSP) is the culmination of research and development that began with a particularly complex database conversion effort. In the early 1980's, Strategic Air Command (SAC) decided to migrate their entire intelligence support database to a completely different environment. Originally, SAC/IN was supported by a unique, home grown database management system developed specifically for SAC in the mid 1970's. In terms of maintainability this was intolerably expensive. To decrease maintenance costs, it was decided to migrate to a commercial product.

The database management system (DBMS) for the new system was the Cullinet DBMS. The Cullinet DBMS (called the Integrated Data Management System or IDMS) was considered by many to be the best DBMS at the time. IDMS was based on the network data model[1], which was consistent with SAC's existing data architecture.

Although the network data model was common to both databases, the hardware platforms and DBMS internals were completely different. The hardware platform in use was a Honeywell 6080; 4 CPU's, 1 MByte main memory (36 bit), and 3.8 GBytes (36 bit) disk storage. The target architecture was an IBM 3081; 4 CPU's, 32 MBytes (32 bit) main memory, 8.8 GBytes (32 bit) disk storage.

The conversion process was intensely manual. Software tools to assist this process were not available and had to be developed from scratch and on the fly. Change control procedures were lengthy and complicated. There were four distinct partitions constituting the development system at HQ SAC; one for development, one for integration, one for final testing, and a fourth for operational use. Physically moving the applications and data from one partition to the next was tedious. Many test errors were traced to missing pieces of software or incorrect versions of software modules being ported from one partition to the next.

Another requirement of the transition process was to provide simultaneous access to both systems. The sheer magnitude of the transition, with its inherently high technical risk, made a "knife switch" cutover approach an unacceptably high operational risk. The databases on both old and new systems had to be synchronized, and both systems required cognizance of what portions of the "operational configuration" were on which system. The existing user interface had to be maintained to the maximum extent possible. Users had to be insulated from the idiosyncrasies of each individual system[2].

NETWORK RESIDENT TRANSITION SUPPORT

The transition could have been orders of magnitude more difficult but for a unique element of SAC's architecture, the Micro-Programmable Controller (MPC). The MPC was an array of asynchronously operating microprocessors that shared a common backplane bus[3]. Developed originally to normalize the physical interfaces between quasi-intelligent workstations of various vendors and the Honeywell mainframe, the MPC evolved into a sophisticated distributed computing environment that was well ahead of its time.

Software was developed within the MPC to support simultaneous system access, minimizing changes to the user interface. Host resident software on the Honeywell system did not require modification and there was no need to develop throw away code on the IBM system. Software implemented on the MPC was essentially an

---

1 Most aspects of data models are extremely well covered in [MAR77]. Cullinet was absorbed by Computer Associates in the late eighties.

2 Additional information on the transition effort is provided in [RAD85].

3 The MPC predated general acceptance of local area networks. It still provides some network services, but has mostly been supplanted by a local area network. The local area network consists of clusters of IEEE 802.3 LANS connected by an FDDI backbone. Additional details pertaining to the MPC may be found in [RAD86].

extension of the network support functions already provided. Unfortunately, this software was essentially throw-away since it would have no purpose once the transition phase was complete.

The difficulties encountered during the transition effort made it clear that automated tools were required for future database transitions. It was also clear that simultaneous access to multiple databases would be a required capability for future systems. The network itself was the logical provider of these capabilities. What exactly these services should be and how the network should provide them was the primary question. Some sort of dictionary/directory would be required that provided database access support services, but what was required beyond that wasn't clear.

DESIGN CONCEPTS FOR DATABASE UTILITIES

As a result, a study effort entitled "Design Concepts for Database Utilities" was initiated to better define the characteristics of network level database access utilities. An architecture for an "Integrated Data Network (IDN)" was developed[4]. The architecture consisted of a three level hierarchy of six types of processors, four of which were specific to the IDN (see figure 1.)

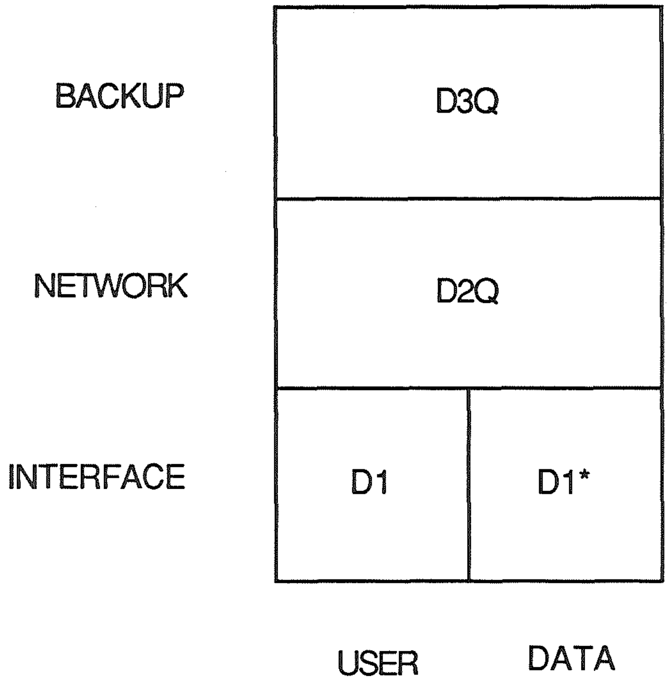

Figure 1
Hierarchy of Processors

The user node corresponds to the processor at which the application or user requesting data resides. Data nodes are the physical repositories of the requested data. User nodes and data nodes are considered outside the scope of the IDN.

At the interface level of the IDN architecture are D1 nodes and D1* nodes. D1 type nodes interface user nodes to the network, accept queries, perform first order validation of the queries, and assemble query responses. D1* type nodes interface data nodes to the network, receive subqueries directed to specific data nodes, accept responses from the data nodes, and compose aggregate responses for transmission to D1 nodes.

At the network level of the IDN architecture are the D2Q nodes. The D2Q nodes complete query validation, dispatch subqueries, and control query execution. These nodes are core to the IDN architectural concept, providing the actual dictionary, directory, and query support services required.

The D3Q node is at the backup level and serves to provide backup facilities for all other types of node, except the user node. Additionally, contents of data nodes can be replicated on D3Q nodes. Replicating data (in the long haul network environment) can improve performance by balancing communication load and supporting fault

---

4 See [RAD86.1] for more details. It is also important to realize that the context of this effort was a wide area (if not global) information network. Performance and fault tolerance were critical design considerations.

tolerant operations. Data node failures won't halt query activity. The resultant network architecture is depicted in figure 2, below[5].

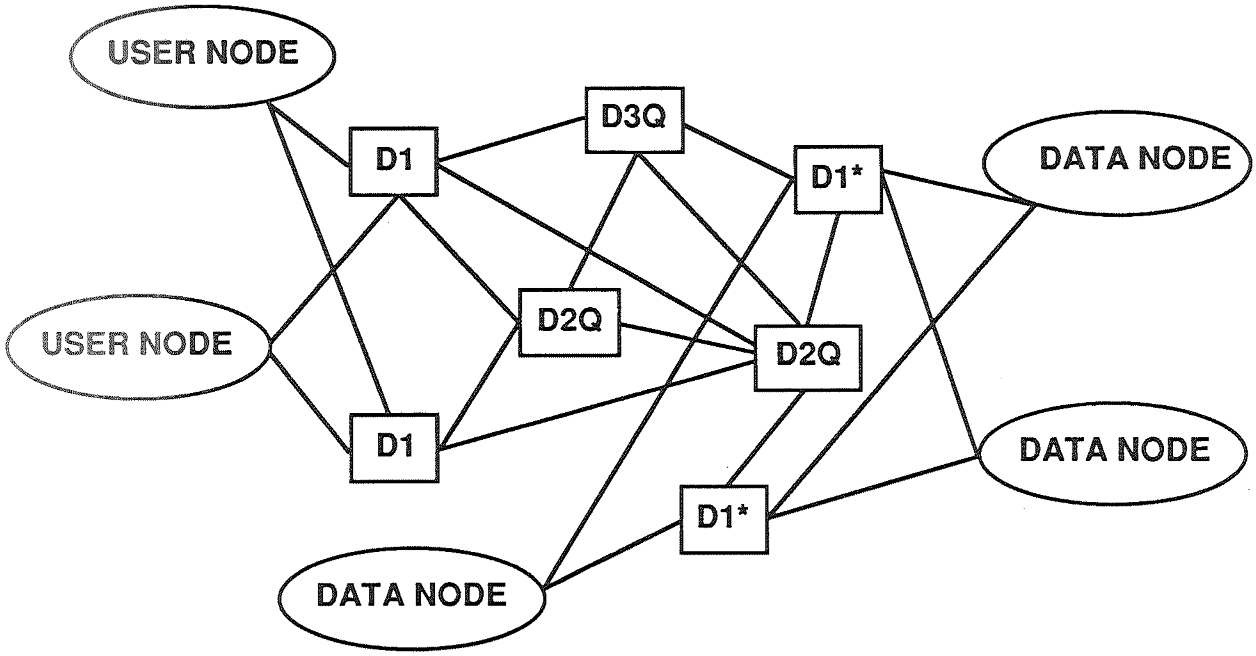

Figure 2
IDN Architecture

5 The architecture was developed deliberately to maximize functional redundancy. The figure illustrates this concept by showing multiple paths between each node. At least on of these redundant paths connects to a shadow node, a node capable of acting as a hot backup for a similar node.

DATABASE QUERY SUPPORT PROCESSOR

During the effort it was realized that the same technology applied to local area networks as well. Implementation details would differ due to differing bandwidth, topological, and fault recovery characteristics of wide area networks versus local area networks. Within the local area network environment, the functionality of the D1 node would be absorbed by the user's workstation, the functionality of the D1* node would be absorbed by the data node, and the D3Q node would constitute the QSP. Since the D3Q provides all the functionality of the D2Q, with the addition of replicated data from selected data nodes, the D2Q can be eliminated as a separate device (see figure 3, Notional QSP Architecture).

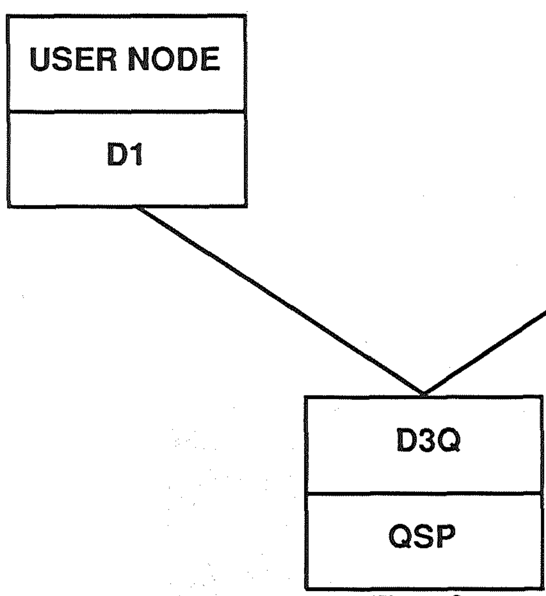

Figure 3
Notional QSP Architecture

For the proposed solution to be effective, it had to have the characteristics of an active, in-line data dictionary at the network level[6]. This meant that all activity against the databases in the network, including application development, database modification and maintenance, and routine database access had to utilize services provided by the utility. The methodology for operation of the IDN and subsequently the QSP, was based on the emerging Information Resource Dictionary System (IRDS) standard[7]. In other words, the functionality of the D2Q or D3Q nodes discussed above, was based on the IRDS standard.

THE INFORMATION RESOURCE DICTIONARY STANDARD

The motivation for the development of the IRDS standard, ANSI X3.138 - 1988, was the proliferation of redundant and inconsistent data. The data dictionary system was seen as a key tool for the effective management of information resources and reduction of inconsistent, redundant data. A number of incompatible, stand alone data

6 Detailed discussion of the philosophy behind data dictionaries and their characteristics is provided in [ROS81].

7 There were two efforts initiated about the same time to develop standards in this area. The American National Standards Committee for Information Systems (X3) began work on a standard for an "Information Resource Dictionary System." The National Institute of Standards and Technology (NIST, formerly the National Bureau of Standards) effort focused on the development of a Federal Information Processing Standard for Data Dictionary Systems. Both groups had identical goals and similar approaches [QED85]. Both efforts were merged in 1983 and the result was the IRDS [ANS88].

dictionary systems were on the market, and each database management system had closed, internal implementations of data dictionaries (if they had any). It was perceived as necessary to develop a standard for data dictionary software[8].

The IRDS standard describes a four level information architecture, level 2 and 3 of which constitute Federal Information Processing Standard (FIPS) 156 (see figure 4, IRDS Architecture). Each level describes and controls the lower level. The first level, Information Resources, is the data in your database. The standard does not apply at this level, although it must accommodate it. The second level, the Information Resource Dictionary (IRD), is the data in the data dictionary, which describe the data in the database. One likely extension of the IRDS approach is to extend the control function from level 2 to level 1. As you might expect, the data dictionary is itself a database that consists of data elements and relationships. Definitions of the data elements and relationships that constitute the data dictionary must be managed. The third level of the IRDS Standard, the Information Resource Dictionary Schema, consists of the definitions of the data elements and relationships contained by the data dictionary. The fourth layer is called the Information Resource Dictionary Schema Description, and consists of data that describes the IRD Schema (level 3).

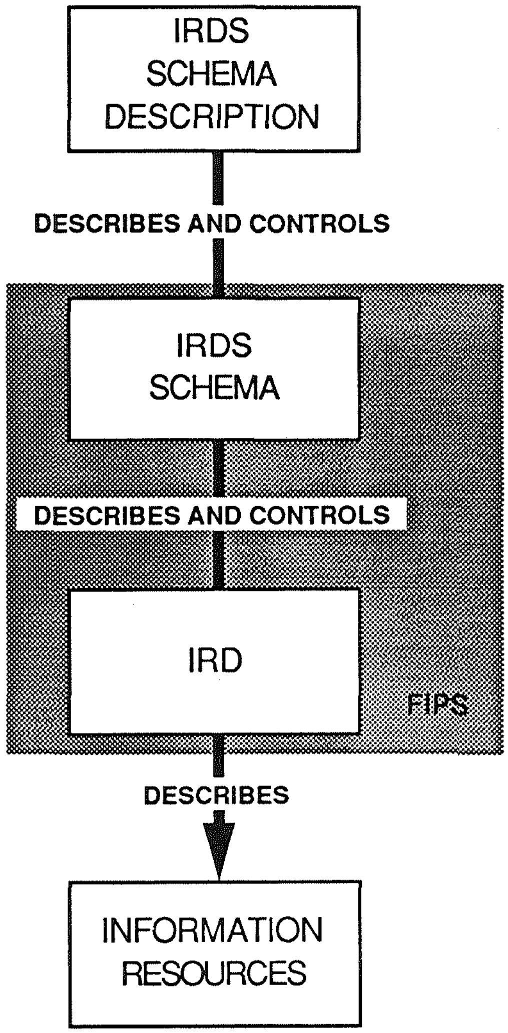

Figure 4
IRDS Architecture

8 See [QED85]. The standards committee took the approach that the standard should specify the characteristics of an interface to a data dictionary and the functionality that a data dictionary should provide. They wisely avoided the mistake of trying to dictate how to implement the dictionary itself.

Key to the concept of levels of description is the corollary that the higher the level, the simpler the model required to describe it. What is left is a mechanism that anyone can use to retrieve data relevant to a specific query. Services provided at each level take care of details such as how to determine what data is available, how to locate it, how to request it, how to navigate the database to get it, and how to put it together into a usable product.

The results of the Design Concepts for Database Utilities work were used by the performing contractor to develop a commercial product in this area. They were successful in obtaining SBIR phase I and phase II funding, and did build a prototype[9]. Rome Laboratory became aware at this time that several organizations were working on similar capabilities.

By 1990 it was apparent that the technology required to support network level database support utilities was mature. The last set of questions requiring answers prior to operational implementation of the technology pertained to performance and policy. More specifically, how much overhead would be introduced into operational systems to achieve what degree of benefit (in terms of flexibility, operations and maintenance savings, etc.). Additionally, simultaneous access to multiple databases adds a new dimension to security policies and procedures, which must be fully understood before implementation.

QSP STATUS

In 1991, the Database Query Support Processor (QSP) effort was initiated to answer these questions. The effort presupposes the availability of network level database support systems with the following capabilities;

a. To retrieve data from multiple databases irregardless of data location, database architecture, or database navigation constraints.

b. To support the definition, modification, administration, and maintenance of:

(1) A network level schema describing the totality of information available from all databases in the network.

(2) Network level subschemas, which are logical subsets of the network level schema and assigned to specific classes of operational users.

c. Provide tools to assist database administrators in defining specific database views for inclusion in the network level schema.

d. Manage and control the definitions of, inter-relationships among, and definitions of inter-relationships of: data elements, data structures, applications, products, user descriptions, and information requirements.

During 1992 and 1993, the QSP effort will focus on collecting quantitative data such as:

a. Volume, patterns, and types of network traffic generated by the QSP.

b. Volume, patterns, and types of network accesses to the QSP.

c. Elapsed time from issuance of a query at a workstation to its receipt by the QSP.

d. Elapsed time from receipt of a query by the QSP to generation of all subqueries.

e. Elapsed time from subquery generation to subquery issuance by the QSP.

f. Elapsed time from issuance of a subquery to receipt by host resident QSP interface software.

g. Elapsed time from issuance of data request by the host resident QSP interface software to that software's receipt of the host's response.

h. Elapsed time from issuance of subquery response by the host resident QSP interface software to receipt of the response by the QSP.

i. Elapsed time from receipt of all subquery responses to the issuance of a query response by the QSP.

j. Elapsed time from issuance of query response by the QSP to receipt of the response by the workstation.

---

9 The Small Business Innovative Research (SBIR) program provides up to $50,000 for phase I efforts and results in a specification for phase II implementation. Phase II provides up to $500,000 for implementation of the idea. Phase III is usually contractor funded and results in a commercial product (with some limited Government rights). See [RAD90] and [RAD90.1] for more information on the SBIR efforts.

The effort will wrap up in 1993 with a comprehensive analysis of collected data in the context of an operational environment. Implications to security policy and accreditation, hardware and software short comings, and operations and maintenance costs will be assessed. Flexibility of the QSP approach will be assessed with respect to the amount of work required to accommodate new databases, changes to old databases, and to initially implement the QSP in an operational network. This data will be used to build a specification for a production version of the QSP.

CONCLUSION

The benefit of the QSP is in the network level support services made possible by the active, in-line repository at the heart of the device. Knowing the relationships among data elements and applications across system boundaries allows better control over change. The ripple effect induced by modifying data elements or applications can be identified in advance and more effectively priced. Additionally, data elements may already exist somewhere in the network that meet the needs of a proposed development, minimizing new development.

Additional benefits could result from adding system documentation to the information available in the network. From the QSP's perspective, documentation can be treated as just another database. Network level information pertaining to relationships among documentation, data elements, applications and other elements of the information environment could be maintained in the QSP. This capability makes update of relevant system documentation an integral part of application or database development, rather than an afterthought.

Data element and application standardization are also supported by the information contained in the QSP repository. The information necessary is already available, all that would remain is to define the rules. Triggers or other mechanisms provide the vehicles for implementation.

The QSP effort will provide hard data on which to base future implementation decisions. Specifically, which services to implement and to what extent to implement those services in operational IDHS systems. Start up costs and the operations and maintenance tail required will also be determined. In the long run, the QSP should provide real benefits in terms of more flexible and robust information systems, with lower operations and maintenance costs.

BIBLIOGRAPHY

[ANS88] American National Standard Information Resource Dictionary System, ANSI X3.138-1988 (Federal Information Processing Standard 156), ANSI, New York, 1988.

[MAR77] Martin, James, Principles of Data-Base Organization, 2nd ed.;Prentice Hall, 1977.

[QED85] AOG Systems Corporation, The Draft Proposed American National Standard Information Resource Dictionary System, QED Information Sciences Inc, Wellesley, MA, 1985.

[RAD85] Planning Research Corporation, Anderson, Richard D and Gerald H. Paes, SAC IDHS Improvement Project, RADC-TR-85-187, October 1985, Secret.

[RAD86] McCabe, Patrick K., Michael J. Wessing, and Ernst K. Walge, SACINTNET Future Study, RADC-TR-86-144, August 1986.

[RAD86.1] AOG Systems Corporation, Design Concepts for Database Utilities, RADC-TR-86-48, April 1986.

[RAD90] AOG Systems Corporation, Local Area Network Schema Server, RADC-TR-90-376, December 1990.

[RAD90.1] AOG Systems Corporation, Local Area Network Schema Server, RADC-TR-90-375, December 1990.

[ROS81] Ross, R.G., Data Dictionaries and Data Administration, New York, AMACOM, 1981.