A Federated Information Management System for the Deep Space Network

E. Dobinson
DSN Data Systems Section

This paper examines some general requirements for an information management system for the Deep Space Network (DSN). It also presents a concise review of available database management system technology. It then recommends that a federation of logically decentralized databases be implemented for the Network Information Management System of the DSN. Overall characteristics of the federation are specified, as well as reasons for adopting this approach.

I. Introduction

There have been many advances in database management systems over the last decade. Faced with the task of modelling a particular application environment, an organization today must make important choices. Off-the-shelf products are commercially available for a wide selection of computers. Nonprocedural query languages, report-writers, forms-based interfaces, programming languages, and graphics are but some of the tools offered for today's applications. However, before selecting a particular product, certain fundamental issues of database organization must be considered. The functional requirements of the application environment must be analyzed and then carefully matched to an information system architecture best suited to meet those requirements.

The purpose of this paper is to study the information management needs of the Deep Space Network of NASA, and to recommend a database management system architecture which will meet those needs most effectively. We begin with an overview of the Deep Space Network, describing the way in which the organization is administered, and the ways in which various elements of this administration interact with each other and with the rest of the Jet Propulsion Laboratory. From this discussion we formulate some general requirements for an information management system [Section II]. We then turn to an examination of information management systems available today [Section III]. By matching the characteristics of these systems with our requirements, we recommend an approach for the DSN, give a top-level design of the system using a small, but representative, subset of data, and indicate how this system can be expanded to serve adequately the whole of the DSN and/or JPL [Section IV]. We then analyze the benefits and costs of our choice in comparison with the benefits and costs of an alternative proposal [Section V]. Lastly, we conclude with a brief summary of the paper [Section VI].

This paper is of particular importance because of its timely coincidence with plans for the Network Information Management System for the DSN already underway. As no major software decisions for this system have as yet been made, the recommendations contained in this paper may be considered.

II. The Deep Space Network

We begin this section with an overview of DSN operations. We then focus in some detail on several key administrative activities. This leads us to formulate general requirements for a DSN information management system.

A. An Overview

The Deep Space Network (DSN) of the National Aeronautics and Space Administration (NASA) is responsible for the guidance and control of all of NASA's unmanned spacecraft at planetary and interplanetary distances, as well as for the receiving and processing of the vast amounts of information these spacecraft acquire and send back to Earth. This network is made up of tracking stations around the world, a central control organization at the Jet Propulsion Laboratory in Pasadena, California, and the ground communications linking them together. The three station groups, called Deep Space Communications Complexes (DSCC), are located approximately 120 degrees apart in longitude, so that a spacecraft is always in view of at least one antenna as the Earth rotates. These locations are Goldstone, California; Madrid, Spain; and Canberra, Australia. These stations function as autonomous organizations under management at JPL. This management is decentralized at JPL in various locations both on and off the laboratory, including a secondary site at Hill Street in Pasadena.

A variety of administrative activities in the DSN require the management of data. These activities are presently supported by separate application systems, each of which has its own set of data. However, for many of these activities the data overlap. At present, there are few automatic mechanisms for these activities to share data. It is cumbersome as well for an activity to span several systems. In addition, it is difficult for the three Deep Space Stations to cooperate in the performance of these administrative functions or to interchange data amongst themselves.

The need for improving this situation has been recognized by JPL management. To this end, an extensive study has been undertaken, under outside contract, which has resulted in a proposal of a hardware and communications configuration for improved operations. The proposed system is called the Network Information Management (NIM) System. It is for this system that we will address our database design. The NIM assembly is described in detail in Refs. 1 and 2. The proposed worldwide network will initially consist of four nodes, one at JPL and one at each of the three station complexes. In addition, each node is itself an internal local network. Each NIM node will have hardware, software, and communications to provide a distributed computing environment for the DSN.

The NIM study, undertaken by the Aaron-Ross Corporation for JPL, has produced an extensive survey of all of the components of the DSN (Ref. 3). This survey identifies the responsibilities and requirements of various DSN activities in terms of their database needs. We shall avail ourselves of its contents throughout this paper in formulating our own recommendations.

B. Some Important Administrative Activities

In this discussion we will focus on some important administrative activities of the Deep Space Network in order to determine a design for an information management system which will permit these activities to function efficiently, and which will give management the overview and knowledge it needs to do its job well. Because the totality of these activities is much too great for the scope of this report, we will concentrate on several important operations which span the entire organization.

1. Engineering Change Management (ECM). Engineering Change Management is a complex, far-reaching DSN activity. It is coordinated at present by a group in section 377 at JPL, directed by a Change Control Board, and involves a large number of personnel throughout the JPL/DSN organization. The process of instituting an engineering change involves the initiation of an Engineering Change Request (ECR). This request is carefully assessed by representatives from all other systems that might be affected by the change. The assessments are then brought before the Change Control Board, which passes judgement on the request. The request may either be denied, approved, or sent back for further evaluation. Once a request is approved, one or more Engineering Change Orders (ECOs) are issued to design and implement the change. Each ECO is then planned in detail, with costs and schedules developed for each phase. At this point the evaluation phase is complete and the design and implementation phase begins.

During the design and implementation phase the ECM group functions to collect status information about the actual schedule performance, and to alert management if any rescheduling will be required. A general awareness of the progress of the ECO is needed by everyone involved, including logistics, maintenance, and mission planning personnel who need to know when the installation will actually occur. As the Aaron-Ross survey points out "... an ECR has the potential to affect nearly every aspect of DSN operations and support, ranging from mission performance analysis to spare parts provisioning and from maintenance personnel scheduling to DSN utilization forecasts. As a consequence, there is a large constituency of personnel, with widely varying needs, all of whom absolutely require or can benefit by a conveniently obtained status and schedule forecast for ECOs.

When the design and implementation of the ECO are complete, the installation at the tracking sites begins. The scheduling of the installation of the equipment must be coordinated with the tracking schedules, so as not to interfere with any mission, and yet be there in time for any future missions that require it. There are also, in addition, some temporary ECOs whose removal must be scheduled similarly. When the equipment is finally installed and running, or when the temporary equipment is removed, the ECO is closed out.

The Engineering Change Management is clearly an important activity, having the capability to affect the DSN in many ways. The data representing the initiation and assessment phases are of interest to a variety of people at JPL, while the data for the development, implementation, and installation schedules may be needed by a variety of personnel throughout the entire organization.

2. **Equipment and Materials Management.** This DSN activity is responsible for the management of JPL property, DSN tracking equipment, repairable spare parts, other maintenance spares, and consumables. Management of equipment and materials involves obtaining them in the first place (provisioning and procurement), keeping track of their location and status (inventory and control), and moving them from place to place (transportation and shipping). These activities are distributed among several organizational elements at JPL and at the Deep Space Stations.

3. **Anomaly Reporting Services.** The knowledge and control of anomalies occurring from time to time throughout the DSN is an important activity for its well being. To this end the DSN has procedures for the reporting of various anomalies. Two categories of reports which are processed are Discrepancy Reports and Failure Reports. The ultimate goal of these reports is to provide DSN engineering, operations, support, and management the information on which to make changes in equipment, technology, procedures, and policy. There are three basic activities connected with these reports. These are (1) filling them out, (2) validating, investigating, and analyzing them, and (3) summarizing the status of the anomalies to management.

4. **Other DSN activities.** In addition to the three activities highlighted above there are many more too numerous to list. These include energy management, financial management, personnel management, scheduling, maintenance, production control, as well as activities that pertain to the tracking sites only, such as operations, repairs, maintenance and integration, cabling, etc. For each of these activities the efficient management of data is extremely important.

C. General Requirements for a DSN Information Management System

There are three general requirements for an information management system for the DSN. First of all, the system must permit the necessary interchange of data between the various administrative activities, as well as between the various physical sites of the organization. Secondly, the system must allow each of these activities to develop and function autonomously. And, thirdly, the system must be capable of evolving incrementally over time.

1. **The need for sharing.** There are some obvious relationships between the various DSN management activities. For example, there is a natural interaction between the ECM system, the equipment management system, and the anomaly reporting system. An ECO almost always will affect the equipment database. Either new equipment will be installed or old equipment will be removed, or both. Anomaly reports can and often do result in the initiation of an ECR. Furthermore, the implementation of an ECO can result in anomalies. And so on. At present none of these interactions can take place automatically. Relationships are maintained manually, if at all.

In addition to the need for having activities share information, JPL management needs to have an overview of DSN operations. This overview requires the integration of data from separate sources within the DSN. JPL might need to know, for example, which stations have completed installation of ECOs for a given ECR, or it might need to compare the cost of the installation from one site to another. The ability to discover unknown relationships is also desirable. There is currently no easy way for management to determine if, for example, a particular piece of equipment causes the same problems at each site where it is used, or for two sites with the same problem to benefit from each other's experience.

To overcome, in part, this problem of management's difficulty in deriving composite information from various sources within the DSN, a pilot system is currently being developed at JPL which will provide integrated data concerning DSN operations, maintenance, and repairs at Goldstone (Ref. 5). This system, called the Productivity Information Management System (PIMS), will provide its users, both managers and management scientists at JPL, a set of tools for manipulating data in a variety of ways. Management scientists will then have the capability to develop and verify operations research models. The implementation of efficient operational policies can then lead to substantial savings and cost reductions. Because the data that management needs is decentralized, and stored in different forms, using different overall methodologies, a major integration effort such as PIMS is presently the only way to provide the overview so badly needed.

2. **The need for autonomy.** In addition to the need for sharing information, there is a conflicting need for activities to remain autonomous. The various DSN management activities are separate and distinct applications. They have developed, and continue to develop, independently of each other, and are under autonomous local control. Integrating the data from all of these activities into a single centralized database is restrictive. Local control of the data is an important aspect of the DSN, as is the independence of one activity from another. It is, therefore, neither desirable nor practical to develop a specification of the totality of operational data for the DSN and to design a logically centralized database.

3. **The need for evolvability.** Coupled with the need for autonomy is the need for evolvability. Administrative activities evolve with the growth of the organization. Some functions are replaced, others are added. The information management of these activities must be capable of evolving also. The database must at all times be an accurate reflection of the organization. It must therefore be dynamic, capable of changing and growing as the DSN changes and grows.

Evolvability of the information management system is important for financial reasons also. Funding comes not all at once, but in small increments over time. The information management system must be capable of incremental development.

III. Database Management Systems

In this section we consider available choices for database management systems in the 1980s. We begin with a brief discussion of data models. We then present a historical development and description of database management system architectures.

A. Data Models

A data model is an abstract representation of the information content of the database. As such, its main function is to insulate the user from the implementation details of the database. Typically, the data in the database is represented using a "conceptual" schema, which is an instance of a given data model. (The relationship of database schema to data model is analogous to that of a program to a programming language.) The data model provides both data structures for representing data and operations for manipulating them. The three best known data models are the hierarchical model, the network model, and the relational model. We now give a brief description of each of these, and site some of the more well-known implementations of each.

1. **Hierarchical data model.** In the hierarchical data model, the data are represented using trees and links. One designated record type occupies the top node of the tree, while its dependent record types are at nodes on lower levels of the tree. The links connect occurrences of these records. These structures model one-to-many relationships, since every dependent record can have at most one parent record. As an illustration of the use of this model, let us consider the canonical example of suppliers and parts. To represent the relationship of suppliers to parts supplied we would have a forest of trees, with a particular supplier at the top of each tree, and the parts supplied by that supplier at the nodes on the next level.

Some of the longest established database management systems adopt the hierarchical approach to data organization. These include the Information Management System (IMS) of IBM, System 2000 of MRI, and Mark IV of Informatics.

2. **Network data model.** Many of the relationships inherent in a database are not one-to-many, but many-to-many. To capture these kinds of relationships a more general structure, called a network, was introduced. A network can be viewed as a graph containing nodes and bidirectional links. Although this allows more flexibility than the hierarchical model and is more efficient in some cases, it is considered more complex.

The most important example of network systems is provided by the proposals of the CODASYL Data Base Task Group, DBTG. Two commercial systems based on these proposals are DMS 1100 by Univac, and IDMS by Cullinane. Other network systems include TOTAL by Cincom, and IDS by Honeywell.

3. **Relational data model.** In the relational model data are organized into tables, called relations, which closely correspond to traditional files. The rows of a table correspond to the records of a file, and the columns correspond to the fields of a record. Associations between the rows are represented solely by data values in columns drawn from a common domain, or pool of values. All of the information in the database, entities as well as relationships, is represented in a single uniform manner, namely in the form of tables. This uniformity of data representation results in a corresponding uniformity and simplicity in data operations.

In contrast to the hierarchical and network models there are no interrecord links in the relational model. This feature gives the relational model an independence from the underlying physical realization of the database. The physical dependence of the hierarchical and network systems stems from the encouraged association between the physical access paths and the logical interrecord links. The absence of such links gives the relational model an added degree of flexibility.

Relational systems are historically the most recent. Some of the better-known relational systems are SQL/DS (System R) and Query-By-Example from IBM, INGRES from Relational Technology, NOMAD from National CSS, ORACLE from Relational Software, and ENCOMPASS from Tandem.

4. **Semantic data models.** In addition to the three commercially available data models described above, there have recently been developed some higher-level models which allow the meaning, or semantics, of the database to be incorporated more completely into the schema. These models differ from the record-oriented models above by employing constructs that are more user oriented, such as objects, types of objects, and attributes of objects.

There are many semantic models currently in use, but their usage is mainly academic. That is, there are no direct implementations of any of them as products. Some of the more well-known of these models are the Entity-Relationship Model (Chen 1976), the Semantic Database Model SDM (Hammer and McLeod 1978), the Extended Relational Model RM/T (Codd 1979), and the Event Database Model (King and McLeod 1981).

B. DBMS Architectures

It is useful to classify databases according to whether they are logically and physically centralized or decentralized. Using this framework, four classes of data base architectures can be identified. Logically centralized and physically centralized databases include conventional integrated databases. Logically centralized and physically decentralized databases include "distributed databases", as well as a number of recent approaches to composite database support. Logically decentralized and physically centralized or decentralized databases are the domain of federated databases.

1. **Logically centralized, physically centralized systems.** Nearly all database management systems in use today, including all of those mentioned in Section III-A, manage databases that are both logically and physically centralized. This means that a single conceptual schema, derived from a formal data model such as the ones mentioned above, is used to structure all of the data in the database. It also means that all of the data in the database are stored in one location. There are three main reasons for integrating data thusly, from separate sources and varying applications, into a unified, coherent whole. One reason is that duplication of the data from one source to another is greatly reduced. The second is that the data becomes logically and physically independent of the application programs that use it. This means that the physical details of data storage and access methods are handled by the system. The entire collection of data in the organization is now an important resource, easy to access and use for a variety of diverse applications. The third reason is that the data are now under control of a centralized authority, who makes decisions for the good of the organization as a whole rather than any one application.

2. **Logically centralized, physically decentralized systems.** The advantages of integrated databases were widely recognized. However, in some applications, the organization itself is geographically distributed. Having the data stored in one central location means high communications costs and degraded system performance. Therefore, the next step taken in DBMS research was to take the data in a logically centralized database and physically distribute it among the various nodes of a computer network. This physical distribution is totally transparent to the user of the system. That is, to the user of the database it is as if all of the data were in one place. The system's performance is improved because the data is located where it is accessed most frequently. Distribution of the database to optimize parallel processing becomes a key design issue for distributed systems. Another key feature of distributed systems is the possibility of increased reliability. A company can reduce the disaster of a computer failure by duplicating the data at more than one site. These features of distributed systems make them highly desirable for many of today's application environments. Therefore, much research and development on distributed systems is currently taking place. Added complexities involving consistency of redundant data, recovery from a failure at any site, and control of concurrent processing pose some difficult research problems. Prototypes have, however, been built, most notably SDD-1 by Computer Corporation of America. It should not be long before a distributed DBMS will be commercially available.

3. **Logically decentralized systems.** Both conventional and distributed systems, though they differ in their physical realization, are logically the same. A single conceptual schema defines all of the data in the database, and the control of the database is centralized, even though the data may not be. This can pose, and has posed, some problems. It has been, in some environments, very difficult to integrate data from many applications because the views they have of the data are different. Logical centralization can force the coupling of data where the retention of some individual autonomy is desirable. Each user of a centralized system is forced to surrender the control of the structure of his data to a central authority, who has the task of organizing all of the parts into a coherent whole. This can have drawbacks. Many individuals are very reluctant to relinquish control of their data, so much so that many an attempted database effort has failed for this reason. Even where this is not the case, centralized control often creates a large bottleneck through which all

requests for change must pass. Changes, therefore, occur reluctantly and slowly, resulting in inaccuracies and anachronisms in the database. In addition, the job of the central authority is an enormous one, for this person (or persons) must understand every aspect of the organization thoroughly in order to model the data well, and must also have a thorough knowledge of DBMS software. The database administrator(s) must choose a design for the system which optimizes usage for the whole collection of users, a design which, however, is often much less than optimal for any one user. Thus, the benefits of integration can have a very high cost.

The notion of a federated database architecture was introduced to remedy these problems. A federation is a union of two or more logically decentralized sources of data which may be, but need not be, physically decentralized. The essential difference from the systems above is the logical decentralization. The individual components of a federation remain under autonomous local control with, however, a certain amount of sharing and coordination. One component of the federation is distinguished as the federal controller. It keeps track of the topology of the federation, and aids in the entrance or departure of a component into or from the federation. The components themselves, through the communications facilities provided by the federation, define the system and negotiate their interactions. Each component has its own schema, which states which of its data is private and which is to be shared in the federation. Individual members of the federation may change internally so long as their interface to the federation is maintained. The federated architecture is both dynamic and modular, with components coming and going at any time. It therefore carries with it all of the well-known benefits of dynamic modular systems.

The research on federated systems is relatively new, and to date there is only a small working prototype at the University of Southern California. However, as a compromise between total integration on the one hand and total autonomy on the other, it is highly desirable for many of today's applications. In addition, the trend today away from large mainframe computers toward networks of smaller machines makes the federated approach to database organization particularly appropriate

IV. A Database Management System for the DSN

In this section we bring together the requirements of Section II and the system characteristics of Section III to recommend a system for the DSN. We then describe in some detail the nature of this system.

A. A Federated System for the DSN

In choosing an architecture for the DSN we must satisfy the three previously stated requirements. These are (1) applications must be able to share data and activities; (2) applications must retain individual autonomy and control of their data; and (3) applications must be able to change with time.

Logically centralized systems fail to meet the second requirement. If we were to adopt a centralized database architecture for the DSN, all of the data from all of the applications would have to be under centralized control. As we have seen from the examples in Section II, this is impractical.

Logically centralized systems also do not meet our third requirement very well. Because at any one time the totality of the database must be represented in a single logical schema, changes in the database require a redesign of the schema.

The characteristics of federated databases, on the other hand, seem to be perfectly matched to the needs of our DSN environment. Federations allow for local autonomy, while facilitating the sharing of data and activities. Federations are also capable of evolving over time. Let us take a closer look at what a federated information management system for the DSN would be like.

1. The topology of the federation. The components of a federation are the logically autonomous units of an organization that sometimes need to share data or jointly to perform some action. In the case of the DSN, these components are the various administrative entities described earlier, such as Engineering Change Management, Equipment and Materials Management, Anomaly Reporting, Repairs, Cabling, etc. A distinguished component, which can be distributed among the sites or be resident at JPL, is the federal dictionary, whose task it is to record the topology of the federation. The dictionary acts in establishing, maintaining, and terminating the federation, as well as in monitoring structural changes.

Each application that needs autonomy, whether it be ECM or Repairs or anything else, will be a logical component. Some of these components will reside on the same machine, others may reside at other NIM nodes, while still others may be distributed throughout the NIM computer network. The degree and nature of sharing and cooperation among these components will be expressed in the component schemas. The federation provides an integrated set of intercomponent communication facilities. These are data importation for data

sharing, message passing for transaction sharing, and negotiation for cooperative activities.

2. **The component schemas.** Each component of a federation is a logical entity having its own component schema. This schema describes the information of concern to the component and has three parts: an export schema, which specifies the information it is willing to share with other components; an import schema, which specifies the information in the federation that the component wishes to access; and a private schema, which specifies local information, which the component is unwilling to share at all.

The export schema for the ECM component would likely contain most of its data, since ECM is a network-wide activity. Its import schema would contain the items exported from the equipment database, cabling database, anomalies database, and possibly others. Other components, such as Repairs, would have a larger private schema while exporting relatively less information. The actual content of these schemas will be decided through the negotiation mechanisms of the federation according to the needs of the components.

It is highly possible that one component importing data from another will need to have a different view of the data. The federated model also provides operators for deriving both types and attributes. This means that is is not necessary for components to agree on a common view of the data for sharing to take place.

The federal dictionary component is the repository of information global to all components, which includes information describing the structure of the federation. Its import schema is used to collect this global information, while its export schema is used to share it with the other components. Any component of the federation can find out from the dictionary what components currently constitute the federation, and how it may communicate with them, as well as obtain a summary of the kinds of information available.

3. **The data model.** The federated architecture requires a common data model to be used throughout the federation, although a component may use any data model of its choice for internal use. Each component uses the federation model to define its export, import, and private schemas. The federal dictionary component uses the model to define the structure of the federation.

While it is possible that any data model could be made to work as the federation model, a semantic model, such as the Event Database Model (Ref. 4) is preferable because many kinds of relationships between the data can be represented. In addition, since the model is not tied to any particular physical representation, the underlying physical implementation of the database can change without affecting its logical expression.

If the logical components of the federation use a different model than the federation model, a translation can be made between the two. This is important if components are to be managed with DBMS software commercially available today. Because of the simplicity and structural independence of the relational model, it is the best commercial choice available today for the components to use.

B. Evolving the Federation

One of the advantages of adopting the federated approach to database organization is that the database can be developed incrementally. Components can come into or depart from the federation at any time. A component can also change internally, so long as it supports its interface to the federation. This evolvability is particularly appropriate for the DSN, as funding is easier to obtain in increments. The federation can grow both within the NIM system and beyond.

1. **Within the NIM.** The extendibility of the federated architecture means that as the NIM communications and hardware expands, so does the federated information management system. The federation for NIM can evolve from the components themselves. They will each express their own export, import, and private schemas, and will use the negotiation mechanism of the model to achieve a desirable configuration. This configuration need by no means be static. Components can negotiate for their entrance or removal from the federation, as well as restructure themselves internally. This means that as new applications are added to the NIM system they can easily become a part of the federation, and assures that the federation will always be an accurate model of the application environment. The basic lines of autonomy and the patterns of interaction are the governing design principles to be embodied.

2. **Beyond the NIM.** The federated architecture can also be extended to include components outside the NIM assembly. This is particularly desirable, for the DSN needs to interact with various JPL institutional systems from time to time. These include institutional systems for financial management, personnel management, property management, work scheduling, mission planning, etc. A higher-level federation, one with the NIM federation as one component (the DSN component) along with these other institutional systems, can be envisoned. The principles of design are the same. All that is needed is the necessary hardware and communications to link them together.

V. Comparison of an Alternative Choice

In this final section we consider a proposed alternative to a federated system. In our evaluation, we focus on two basic features. The first of these is the desirability of a general-purpose vs a special-purpose system. The second is the desirability of a dynamic vs a static system. Also, the life-cycle costs of both choices must be considered.

The database system now under consideration for the DSN is a system of separate, independent databases for each application on the NIM. This approach is an electronic counterpart of the situation that exists now, and can be achieved with little research or development effort. As before, each application will own its data. However, because of the communications provided by the NIM network, any application will be able to peruse the data from any other application's database. Nevertheless, one application will not be able to use the data in another database without writing a program to incorporate that data into its own system. The imported data will have to be duplicated, interpreted, and restructured before it can be used. Composite information will still be extremely difficult to obtain. This is because each application will have data in different and incommensurate forms. The task of utilizing these different views of the data is not trivial, and standardizing these views is tantamount to centralization. In addition, keeping redundant copies of data creates a problem of consistency with updates that must be dealt with. The costs of duplicate storage space, and of time to transmit copies back and forth across the network must also be considered.

While it is true that the interactions provided by the federated model can be realized on a case-by-case basis by means of ad hoc application programs, this can become extremely costly in the long run. For, as the number of components increases, the cost of application software to tie them together grows geometrically, whereas the cost of the federated software stays the same. The federation provides a general-purpose system for maintaining autonomy while facilitating sharing. The distinction between the ad hoc alternative approach and the federated approach is the distinction between generality and flexibility on the one hand, and specificity and rigidity on the other.

Also, if it were possible to state at any one time all the ways in which the data are to be shared amongst the users of the NIM then one could implement the necessary programs to do this. However, obtaining such a specification is unrealistic. Changes are a fact of life, and the ability to respond to changes is a highly desirable feature, saving much cost over the years. Only a federated system offers the ability to change these interapplication, intercomponent relationships dynamically. Therefore, it is the life-cycle cost of each alternative that must be compared. The added time, effort, and dollars necessary to implement the federated information management system, from first principles, is more than likely to pay for itself as time goes by.

VI. Conclusion

In this paper we have examined several key DSN administrative functions. We have seen how these activities need to have a data management system which will allow them to retain their individual autonomy and which will also allow them to share data. We have also seen that these activities need to be able to grow and change independently of each other. They therefore require a data management system that is dynamic.

We feel that the federated approach to database organization is particularly appropriate to this situation. We also feel that the benefits of implementing it far outweigh the costs. The development of a federated information management system is an ambitious undertaking, but one worthy of such an important organization as the DSN. The results are very exciting to contemplate.

Acknowledgments

This paper was written under the supervision of Dr. Dennis McLeod, Department of Computer Science, University of Southern California, in partial fulfillment for the Master of Science degree.

I would like to thank Dr. Dennis Heimbigner of USC for his direction of this effort, and Dr. Richard Hull of USC and JPL for his valuable comments. I would also like to thank the various people at JPL for their information and/or support: Bob Iverson, George Madrid, Conrad Chatburn, Russ Hedges, Merle McKenzie, and Larry Hawley. In addition, I am grateful to the people of the Aaron-Ross Corporation for conducting and making available to me their exhaustive user survey.

References

1. "Deep Space Network Subsystem Functional Design: Deep Space Network Information Management (NIM) Assembly," Aaron-Ross Corporation, Glendora, Calif., 835-2, Feb. 1982.

2. "NIM Fourth Progress Review: Level 'D' (Backup Information)," Aaron-Ross Corporation, Glendora, Calif., Feb. 1982.

3. "NIM Program Specification Document," Aaron-Ross Corporation, Glendora, Calif., Feb. 1982.

4. King, R., and McLeod, D., "The Event Database Specification Model (A Preliminary Report)." In *Database Modelling*, editors M. L. Brodie, J. Mylopoulos, and J. Schmidt, 1982 (to be published).

5. Hull, R. "An Introduction to the New Productivity Information Management Systems (PIMS)," *TDA Progress Report 42-70*, Jet Propulsion Laboratory, Pasadena, Calif., Aug. 15, 1982.

Bibliography

Aaron-Ross Corporation, "Deep Space Network General Requirements and Plans: DSN Network Information Management (NIM) Assembly (1982 through 1989)," 821-6, Revised Dec. 1981, Glendora, Calif.

Date, C. J., *An Introduction to Database Systems*, Addison-Wesley, 1981.

Hammer, M., and McLeod, D., "On the Architecture of Database Management Systems." In *Infotech State-of-the-Art Report on Data Design*, 1980.

Heimbigner, D., and McLeod, D., "Federated Information Bases (A Preliminary Report)." In *Infotech State-of-the-Art Report on Data Design*, 1981.

Heimbigner, D., and McLeod, D., "An Approach to Logical Database Decentralization." Computer Science Department, University of Southern California, Los Angeles, Calif., Dec. 1981.

King, R., and McLeod, D., "Semantic Database Models," *Database Design*, editor S. B. Yao, 1981.

King, R. and McLeod, D., "A Semantics-Based Methodology For Database Design, Use, and Evolution," Computer Science Department, University of Southern California, Los Angeles, Calif., Dec. 1981.

McLeod, D. and Heimbigner, D., "A Federated Architecture for Database Systems." In *Proceedings of the National Computer Conference*, AFIPS, pp. 283-289, Anaheim, Calif., May 1980.