2

MASTER

**Wilson-Hill
Associates, Inc.**

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

DISCLAIMER

This book was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

ENERGY EMERGENCY
MANAGEMENT INFORMATION
SYSTEM (EEMIS)

FUNCTIONAL REQUIREMENTS

Contract Number DE-AC01-80 EI11581

October 17, 1980

Prepared for:

THE DEPARTMENT OF ENERGY
OFFICE OF THE ENERGY EMERGENCY MANAGEMENT
INFORMATION SYSTEM
1000 Independence Avenue
Washington, D. C.

Prepared by:

WILSON HILL ASSOCIATES
1025 Vermont Avenue, N.W., Suite 900
Washington, D. C. 20005

Revised December 12, 1980

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

TABLE OF CONTENTS

	<u>Page</u>
1.0 INTRODUCTION.....	1-1
1.1 Background.....	1-2
1.2 Purpose.....	1-3
1.3 Organization.....	1-3
2.0 EEMIS STRUCTURE.....	2-1
3.0 TELEPROCESSING MONITOR.....	3-1
3.1 Multiple Simultaneous Users.....	3-1
3.2 Multiple Terminal Protocols.....	3-2
3.3 Interactive Processing.....	3-2
3.4 Job Submission to Batch Processing.....	3-2
3.5 Command Control Files.....	3-3
3.6 Security/Privacy.....	3-3
3.7 Response.....	3-3
3.8 Recovery.....	3-3
4.0 EEMIS-S FILE REQUIREMENTS.....	4-1
4.1 EEMIS-S File Structure.....	4-1
4.2 Physical Data Storage Requirements.....	4-2
4.3 Data Access Methods.....	4-3
5.0 DATA BASE MANAGEMENT REQUIREMENTS.....	5-1
5.1 General Requirements.....	5-1
5.1.1 Software Interface.....	5-2
5.1.2 Integrity.....	5-3
5.1.3 Growth and Change.....	5-4
5.1.4 Hardware Support.....	5-5
5.1.5 Security.....	5-5
5.1.6 Processing Capabilities.....	5-6
5.1.7 Internal Structuring Capabilities.....	5-7
5.2 EEMIS-S Specific Requirements.....	5-8
5.2.1 Support of a Large Number of Files.....	5-8
5.2.2 Multiple Coupling of Files.....	5-8
5.2.3 Multiple Access to Data.....	5-9
5.2.4 Variable Length Records.....	5-9
5.2.5 Phonetic Checking.....	5-9

TABLE OF CONTENTS

(Cont'd)

	<u>Page</u>
6.0 APPLICATION PROGRAM REQUIREMENTS.....	6-1
6.1 Information Retrieval System.....	6-1
6.1.1 Program Statements.....	6-2
6.1.2 Program Organization.....	6-3
6.1.3 Operational Statements.....	6-4
6.1.4 Data Management.....	6-5
6.2 File Maintenance System.....	6-7
6.2.1 Program Statements.....	6-7
6.2.2 Program Organization.....	6-10
6.2.3 Processing Modules.....	6-11
7.0 UTILITY SOFTWARE.....	7-1
7.1 The Sort/Merge System.....	7-1
7.1.1 Input/Output Devices.....	7-2
7.1.2 File Types.....	7-2
7.1.3 Record Types.....	7-2
7.1.4 Sort Configurations.....	7-3
7.2 The File Handling System.....	7-3
7.2.1 Input/Output Devices.....	7-4
7.2.2 File Types.....	7-4
7.2.3 Record Types.....	7-4
7.2.4 Editing and Reformatting.....	7-5
8.0 JUSTIFICATION FOR PROCUREMENT UNDER TSP-MASC.....	8-1
8.1 Procurement Lead Time.....	8-2
8.2 Response Time.....	8-2
8.3 System Life.....	8-2
8.4 System Backup.....	8-3
8.5 Secondary Advantages.....	8-3
8.5.1 Government-wide Volume Discounts.....	8-4
8.5.2 Reduce Administration Costs.....	8-4

1.0 INTRODUCTION

The Office of Energy Emergency Management Information System (EEMIS) of the Department of Energy (DOE) has responsibility for providing special information and communication services to government officials, at Federal and State levels, who must deal with energy emergencies.

The EEMIS System contains proprietary information presently residing on the Energy Information Administration (EIA) computer that must be safeguarded from intentional or inadvertent disclosure. Hence the states are not allowed direct access to the EEMIS System in its present environment. To satisfy state needs a separate EEMIS system, EEMIS-S, is being prepared for their use.

To avoid security/privacy problems and to alleviate pressure on the EIA computer, the Office of EEMIS intends to temporarily acquire teleprocessing services from a time-sharing commercial vendor to make EEMIS-S available to state users. These services are to be procured under the General Services Administration (GSA) Teleprocessing Services Program (TSP) schedule contracts. The process for procurement must meet with the guidelines for approval by EIA's Office of ADP Services, Department of Energy (DOE)'s Office of Computer Services and Telecommunications Management, and GSA.

These guidelines demand (1) that a functional system requirements document be produced highlighting crucial DOE processing characteristics which will be specified as mandatory requirements; (2) that the evaluation criteria used to select the winning bidder be specified; and (3) that the script for the benchmark demonstration by each bidder be provided.

In order to achieve these objectives the DOE has retained Wilson Hill Associates (WHA) under contract number DE-AC01-80 EI11581, to develop an EEMIS functional requirements document, a list of mandatory requirements, the benchmark evaluation criteria, and the benchmark scenario and script.

This report deals with the functional requirements of the Energy Emergency Management Information System (EEMIS) as it is defined for State level use.

1.1 BACKGROUND

The Energy Emergency Management Information System for State level use (EEMIS-S) embodies a large aggregate pool of energy specific information. This information is continually gathered and updated from many and oftentimes discrete sources. Following the removal of the source identity data and the aggregation of individually identifiable data the information will be transferred to the EEMIS-S data base for direct use by inquiring users within State Energy Offices. These users will have the freedom to communicate directly with the EEMIS-S data base through the use of telecommunications and local computer terminals. This facility will place four major categories of resources at the users disposal. They are:

- Energy Data - These are items of information on energy resources, from source to end-use, pertinent to managing energy-related matters. They are indexed and stored (recorded) in the computer's memory.
- Microfilmed Printed Matter - Microfilmed printed matter is background information to energy data. It is indexed within the data base like ENERGY DATA, but is recorded on microfiche, which are available at the user's location.
- Communications - These are plain-English messages sent and received amongst EEMIS users. Messages have the same format as memoranda (e.g., To, From, cc) and are filed in the computer's memory.
- Contacts - These are names of key individuals in Federal and State Government and in industry. Names are stored in the computer along with information needed to contact these people in an emergency (e.g., telephone numbers, affiliations, addresses, etc.) and are indexed like ENERGY DATA for quick retrieval.

The EEMIS-S system is intended to reside in a national timesharing vendor's computing environment. All State users will have access to EEMIS-S through the vendor's communications network.

The functional requirements stated in this document will describe the software systems which the vendor will be required to support.

1.2 PURPOSE

The purpose of this document is to provide a technical description of the EEMIS-S requirements, as stated in functional terms, to be included as part of the RFP for timesharing services under the guidelines of the GSA TSP Multiple Award Schedule Contracts (MASC). These guidelines state that in order to create the widest practicable competition the system's requirements, with few exceptions, must be expressed in functional terms without reference to specific hardware or software products, and that wherever exceptions are made a statement of justification must be provided. In addition, these guidelines set forth a recommended maximum threshold limit of annual contract value for schedule contract procurements. Justification for exceptions to the threshold limit under MASC must also be provided.

The EEMIS-S functional requirements and statements of justification for exception where required are presented in the following sections.

1.3 ORGANIZATION

Section 2.0 presents a general overview of the EEMIS structure in terms of requirements for vendor support. The intent is to provide the reader with an understanding of the relationships between the following Sections, 3.0 through 7.0, and

the various EEMIS structural components to be described in functional terms. The functional requirements for each component are developed by section as: Teleprocessing Monitor Requirements, Section 3.0; EEMIS File Requirements, Section 4.0; Data Base Management Requirements, Section 5.0; Application Program Requirements, Section 6.0; and Utility Program Requirements, Section 7.0.

The final Section, 8.0, justifies the use of the TSP-MASC procurement process. The intent of this section is to substantiate, in this instance, the desirability of obtaining timesharing vendor services to support EEMIS under a schedule contract, even if certain TSP-MASC threshold limits might be exceeded.

2.0 EEMIS STRUCTURE

This section is intended to provide the reader with a foundation for understanding the functional requirements presented in the following sections. This discussion should be interpreted only as a statement of the basic EEMIS-S structural components and not as the overall EEMIS functional design. For a detailed design description the reader is referred to the EEMIS-I Detailed Design document, (M.10), prepared for the Department of Energy, Office of EEMIS, by the Planning Research Corporation, McLean, Va.

Within the EEMIS-S environment there are five basic structural components. These establish the time-sharing vendor's software support requirements. They are:

1. The Teleprocessing Monitor
2. The System of EEMIS Files
3. The Data Base Management System
4. The EEMIS Application Programs
5. The Utility Programs.

These components are structurally related as shown in Figure 2-1.

The Teleprocessing Monitor, the Data Base Management System and the Utility Programs will be supplied by the timesharing vendor. The data to be loaded into the files and the application programs will be supplied by the Department of Energy, Office of EEMIS.

The System of EEMIS Files, i.e., the file structure and relationships, are both conceptual and procedural and therefore require that the Vendor's Data Base Management System support the designed file organization and properly execute the functions which manipulate the data.

FIGURE 2-1. EEMIS-S BASIC STRUCTURAL COMPONENTS

In like manner, the EEMIS Application Programs are both conceptual by virtue of design and procedural in their operation. It is required, therefore, that the vendor's Teleprocessing Monitor, Data Base Mangement System and Utility Programs, interface with the application software so as to produce the required procedural effect.

The Application Programs are written in a high level compiler language which supports certain desired EEMIS functional and manipulative processes. In order for the vendor to prepare core resident, executable load modules compatible with the object hardware it is required that the vendor's compiler or interpreter recognize this language and convert the coded intelligence to the proper procedural sequence of executable instructions.

The following sections of this report state the functional requirements for each of the above mentioned structural components.

3.0 TELEPROCESSING MONITOR

The vendor is required to supply a teleprocessing (TP) monitor which operates within the environment of the object hardware and which will support the total EEMIS-S system. Specifically, the TP monitor must satisfy the minimal support requirements as follows.

- Multiple, Simultaneous Users
- Multiple Terminal Protocols
- Interactive Processing
- Job Submission to Batch Processing
- Command Control Files
- Security/Privacy
- Response
- Recovery.

Each of the requirements listed above must meet the criteria as described in the following subsections. These requirements and the following criteria can be met under the IBM Timesharing Option (TSO), teleprocessing monitor or its equivalent.

3.1 MULTIPLE SIMULTANEOUS USERS

The TP monitor must be able to service a minimum of thirty simultaneous users within a single central processing unit (CPU). The vendor, however, is not restricted from distributing the workload between multiple CPU's provided the disk resident EEMIS-S data base is made simultaneously available to each CPU and thus to each user.

3.2 MULTIPLE TERMINAL PROTOCOLS

The TP monitor must be able to recognize the various industry standard data transmission protocols generated by terminals of recognized manufacturers. These may include such devices as hard copy terminals, CRT's, and graphic displays. The TP monitor must also be able to respond to data transmission in either the full or half duplex mode.

3.3 INTERACTIVE PROCESSING

The TP monitor must be able to support the execution of application programs called by remote terminal users and to provide the facility for interaction, (read/write capability) between the remote terminal user and the application program. The TP monitor must be able to maintain surveillance over the application program being processed and to report to the user any malfunction that may occur. In addition the TP monitor must be able to intercept and act upon user generated interrupt commands, i.e., control functions, attention and break, during the processing of the application program.

3.4 JOB SUBMISSION TO BATCH PROCESSING

The TP monitor must be able to respond to a remote terminal user's command for batch processing job submission and be able to pass on to the operating system's virtual reader the job control file specified by the users command. In addition the TP monitor must have the facility to support the user in the creation and maintenance of job control language files for the purpose of submission to the batch processing environment. The TP monitor must also have the facility to notify the remote terminal user upon completion of a batch processed job.

3.5 COMMAND CONTROL FILES

The TP monitor must have the facility to execute a program file of its own commands. These commands must include input/output instructions, logical decisions, branch instructions, calls to application programs, file definitions and file allocations. In addition the TP monitor must have the facility to support the user in the creation and maintenance of command control files.

3.6 SECURITY/PRIVACY

The TP monitor must be able to protect the EEMIS-S system and certain EEMIS-S files from intrusion by unauthorized users. Specifically, the TP monitor must support a logon procedure which requires both a user identification and password. Further, it is required that the TP monitor obliterate the password at the remote terminal printer or display device to protect it from inadvertant disclosure. It is also required that the TP monitor protect the EEMIS-S files under direct control of the TP monitor with file names concatenated to the user ID.

3.7 RESPONSE

The TP monitor must be able to respond to the remote terminal user within three seconds following the release of a data transmission interrupt. This requirement must hold true for an average of 95% of the time that the system is in use.

3.8 RECOVERY

The TP monitor must be able to recognize a catastrophic interrupt in communications between itself and the remote terminal user. In this event it is required that the TP monitor execute an orderly shutdown of all such user operations,

closing files and terminating application program functions,
(with the exception of batch processing) and reinitializing
itself for user re-entry through normal logon procedure.

4.0 EEMIS-S FILE REQUIREMENTS

The vendor is required to supply the data storage media and an associated operating system that will meet the demands of the EEMIS-S file requirements. This section describes the file requirements and establishes the criteria for the media and the system.

4.1 EEMIS-S FILE STRUCTURE

The overall EEMIS-S file structure encompasses two index file systems and four data file systems which, as an aggregate, include approximately eighty individually defined files. Most of the individual files are integrated under the control of the data base management system (Section 5.0). Those remaining are under the control of the teleprocessing monitor (Section 3.0).

The two indexed file systems include the user index files and the integrated index dictionary. There are as many user index files as there are users. These retain information for each users session and include the users ID verification, labels and formats for printed output and the user generated pointers to the energy data base as constructed during any twenty-four hour period.

The integrated index dictionary contains keywords and pointers to the energy data files. It is through this dictionary that user requests are constructed and energy information is gathered, expanded or summarized.

The four energy data file systems constitute the greater of the EEMIS-S file requirements. They consist of , 1) a reference energy data system that is organized by the

facets of resource, process, geographic location and time.

- 2) a communications system that contains textual information to be exchanged between users, 3) a microform index that refers to documents stored on microfiche, and
- 4) a contacts information file that contains the names and addresses of individuals who may provide valuable assistance during an energy emergency.

The data files are hierarchically indexed and are of variable record length. Other files, including those containing user session keywords, are either index sequential or sequential depending upon their use.

4.2 PHYSICAL DATA STORAGE REQUIREMENTS

Because of the nature of the EEMIS-S files and of the EEMIS-S application requirements it is required that the vendor be able to provide on-line disk storage space of the order of 1300 megabytes. It is also required that additional space be available when needed. As an outcome of the complex EEMIS-S indexing requirements, the data storage media must be of a high density in order to minimize the imposed seek time.

In addition to the dedicated EEMIS-S data file storage requirements there are also requirements for both transient and permanent areas controlled by the teleprocessing monitor. These files contain command control lists and user specific information.

It may be required that certain data and/or programs be retained in archival files. In this event the vendor is required to provide storage media for tape resident files.

4.3 DATA ACCESS METHODS

It is required of the vendor's operating system that it support the various data access methods upon which the EEMIS-S files are based. These include sequential access, index sequential access, and direct (random) access. The latter of these must also be supported by the operating systems ability to discriminate between fixed length records and variable length records.

5.0 DATA BASE MANAGEMENT REQUIREMENTS

The vendor is required to supply a data base management system (DBMS) which operates within the environment of the object hardware and which will properly control all EEMIS-S files and interface with the EEMIS-S software. The functional requirements for the data base management system are stated in this section.

The section is divided into two parts. The first part describes the general requirements for the DBMS while the second part describes the specific requirements considered to be essential to the EEMIS-S application. Within each part the requirements are identified, listed and expanded according to the criteria that must be met for the DBMS to properly function in the total EEMIS-S environment.

Both the general requirements and the specific requirements can be met to the extent of the criteria under the ADABAS data base management package or its equivalent.

5.1 GENERAL REQUIREMENTS

The data base management system must satisfy certain general requirements to be able to support the functional EEMIS-S application. These requirements have been identified and grouped into the following major categories.

- Software Interface
- Integrity
- Growth and Change
- Hardware Support

- Security
- Processing Capabilities
- Internal Structures.

Each of the requirements listed above must meet the criteria as described in the following subsections.

5.1.1. Software Interface

The data base management system must run within the vendor's current software environment. It must be able to interface with other CPU resident programs including the teleprocessing monitor, user defined data dictionary systems, various host languages, user defined query languages, and other application software. These are expanded upon in the following paragraphs.

- Teleprocessing Monitor - The DBMS must be able to interface with the vendor's teleprocessing monitor. The method of interface may be accomplished either through conversion bridges or more directly to the DBMS intrinsics.
- Data Dictionary System - The DBMS must interface with the EEMIS-S data dictionary system for proper control and administration of data. It is preferable that the data dictionary be integrated with the DBMS for greater efficiency.
- Host Language Support - The DBMS must support the most commonly used high level source languages which are identified as industry standards.
- Query Languages - The DBMS should support an interactive query facility for ad-hoc retrieval of information. The query language must be easy to use and efficient in execution.
- Application Software - The DBMS must be able to interface with other application software and,

more specifically, must be able to interface with the vendor supplied utility programs.

5.1.2. Integrity

The DBMS must insure the integrity of the data under its supervision. It must support controls ranging from preventive measures through backup and recovery. Specifically, the DBMS must provide a central DBMS monitor, interlock resolution, backup and recovery, utility support, audit trail log, and control masks. Each of these control functions are expanded upon in the following paragraphs.

- Central DBMS Monitor - The DBMS software must have the capability of a central processing monitor for both data base and data communications. This will allow concurrent processing of both on-line and batch programs.
- Record Processing Interlock Resolution - The data base software must recognize that a deadlock (or deadly embrace) has occurred and take appropriate action. Deadlock occurs when two programs lock up files or records needed by each program such that neither program can complete processing. The method of resolving deadlock should minimize restricted access to the data base.
- Backup and Recovery Facilities - It is essential that the DBMS have the capability to recover the data base from a system, application or device failure after changes have occurred. Backup is the generation of a copy of the data base at some point in time and recovery is the restoring of the data to a previous state.
- Utility Support - Utilities are required and necessary for proper control and recovery of the data base. Checkpoint, restart, dump/restore, and log utilities should be available for data base reconstruction. Utilities should also be available for data base modification and reorganization.
- Audit Trail Log for Applications - Audit trails provide a summary of data base activity and are examined for verification of transactions. Audit trails must

provide additional information for recovery or reprocessing of transactions.

- Application Processing Control Masks - The application program must be able to reference only those data elements which it requires. The DBMS should not allow the application program to modify any portion of the actual data structure which may destroy the integrity of the data base.

5.1.3 Growth and Change

The DBMS must support growth and change in terms of users and data. It must be able to accomodate modification, verification and extension of the data base with minimal reorganization. Specifically, the DBMS must provide data independence, reorganization capability, data base expansion, tuning, and structural change. Each of these are expanded in the following paragraphs.

- Application Program/Data Independence - The DBMS must allow modifications to be made to the data base with minimal impact on the application software. Field independence is important and must facilitate changes such as record or field expansion. The addition of new relationships to obtain increased data integration must be transparent to the existing applications.
- Reorganization - The DBMS must allow the data base to be easily reorganized. However, the expansion of records or fields and the addition of new logical relationships should require that a minimal reorganization should occur.
- Data Base Expansion - The DBMS should not inhibit growth or change due to arbitrary physical or logical size restrictions. There should be no practical limitations on data base size.
- Tuning Capabilities - The capability to tune the data base is required to provide optimum performance for the vendor's particular environment. The DBMS should provide options to make performance versus storage trade-offs.

- Structural Change - The DBMS must provide data structures with versatile relationships and self-contained capabilities to minimize restructuring of the data base due to changes in data relationships.

5.1.4. Hardware Support

The DBMS must make efficient use of the vendor's available hardware. It should support multiple direct access devices, use minimum central processor (CPU) resources, use minimum disk storage and make use of multiple CPU's where available. These requirements are highlighted in the following paragraphs.

- Direct Access Devices - The DBMS must be able to address multiple direct access data storage devices. These devices should not be restrictive with regard to physical limits placed on data fields, record lengths or block sizes. It is desirable that the total capacity of each device be in the range of two-hundred to three-hundred mega bytes.
- Central Processor Resources - It is required that a minimum amount of main memory be used to support the vendor's data base software. It is also required that a minimum number of CPU cycles be used to perform the data base functions.
- Disk Storage Space - It is required that the DBMS use a minimum amount of disk storage space to support the data base. Data storage space should only be allocated for the storage of significant data. Unused disk space should be returned to working storage.
- Distributive Data Processing - The DBMS should be able to support a distributive data base concept where data bases may be located in multiple computers.

5.1.5 Security

The DBMS must insure the security of the data base and

control user access through multiple levels of authorization. The data base must be protected from inadvertent or intentional misuse and/or destruction. The criteria for security are expanded as follows.

- Levels of Security - Because of the sensitivity of the EEMIS data, an effective DBMS security system is required. For the EEMIS-S application users authorization is required. This protection must be selective via authorization codes at the record level, the field level and the data element level.
- Transaction Processing Security Interface - Generally, terminal security and password security are provided for batch and on-line processing, see Section 3.0, Teleprocessing Monitor. Security locks and password protection include the ability to change passwords. The logging of unauthorized attempts to access the data base and notification to the master terminal of repeated violations is required.

5.1.6 Processing Capabilities

The DBMS must effectively and efficiently support varied processing requirements. These include the various methods of data base access and the configurations of conditional constraints required for the proper selection of data. The following paragraphs describe in more detail the criteria for these requirements.

- Sequential Processing of Data - The DBMS must be able to access and retrieve data base records in a sequential manner. This capability will be required for the EEMIS batch processing needs and should be accomplished in an efficient manner.
- Keyed (Selective) Processing of Data - Keyed access is the ability to access a record by designated identifiers within one or more records. This is a definite requirement to support on-line processing and the information retrieval capabilities of EEMIS.

- Multi-Thread Processing of DBMS Requests - The DBMS must be able to process more than one request while waiting for data to become available from the data base on another request.
- Multi-Requests Outstanding per Requestor - The EEMIS-S application program may issue more than one request to the DBMS at a time. The DBMS must, therefore, overlap data access with other internal processing requirements.
- Boolean and Conditional Record Selection - The DBMS should support simple and complex value searches using combinations of relational conditions (<, =, ≠, >) and the boolean operations (AND, OR, NOT).

5.1.7 Internal Structuring Capabilities

The DBMS must provide a variety of access paths to the data. Internal structuring should support natural relationships among data as required by the retrieval function. These are expanded upon as follows.

- Data Structures - The DBMS must support both hierarchical and network structures. These are required to supply the proper access paths to the data.
- Index Access Method - Indexed files must be supported. They are designed for direct processing. They contain tables of keys and corresponding addresses of data records which are arbitrarily stored in the data base.
- Direct Access Method - Direct access files must be supported. They are designed for random processing. Keys are processed by an address generating algorithm to locate the data record. This method generally provides fast access for on-line processing.
- Record-to-Record Relationships - Natural data relationships such as one-to-many and many-to-one record-to-record relationships must be supported. Desirable ordering capabilities include the retrieval of first, last, next, prior, and sorted records. The sort option

should cause the new record to be automatically positioned in ascending or descending order.

5.2 EEMIS-S SPECIFIC REQUIREMENTS

The EEMIS-S application requires that the DBMS support certain special functions. These functions are necessary to the application in its support of a massive data base to be accessed by numerous simultaneous users who are inexperienced in data processing techniques. It is required, therefore, that the DBMS support the following.

- A large number of files
- Multiple coupling of files
- Multiple access to data
- Variable length records
- Phonetic checking.

Each of these requirements are expanded in the following subsections.

5.2.1 Support of a Large Number of Files

The EEMIS-S energy data and the related communications and polling functions require that the DBMS support as many as eighty files. These files are hierarchically organized and are referenced via a multi-level data dictionary containing index values (pointers) to the contents of files.

5.2.2 Multiple Coupling of Files

In response to a user's request (query) EEMIS-S may need to

gather data from several files where the key configuration has not been defined. In order to service these requests in the most efficient manner, the DBMS must allow for the logical coupling of multiple files.

5.2.3 Multiple Access to Data

The EEMIS-S system is designed to support multiple users and to provide them with simultaneous access to the data base. In order to achieve responsiveness under these conditions, the DBMS must provide alternate paths for multiple access to the data field level.

5.2.4. Variable Length Records

The nature of the data residing in the EEMIS-S data base is such that fixed length records would generate inefficiencies and wasted space. In addition, the introduction of new data should not be constrained by the length of a record. It is required, therefore, that the DBMS support the use of variable length records.

5.2.5 Phonetic Checking

The EEMIS-S data dictionary contains many unusual technical, commercial, foreign, and rarely used words. In order to assist users in their requests for information, a phonetic check is run against the terminal entries and the data dictionary. This is to assure the user of a system response even in the case of misspelled words. The DBMS must support the phonetic checking of field data.

6.0 APPLICATION PROGRAM REQUIREMENTS

The vendor is required to provide the necessary compilers, interpreters or assemblers needed to convert all EEMIS-S source language statements to executable load modules which will perform the required EEMIS-S functions within the vendor's hardware environment. The functional requirements for these vendor supplied programs are stated in this section.

The section is divided into two parts. The first part describes the compiler requirements for the EEMIS-S information retrieval systems while the second part describes those requirements for the EEMIS-S file maintenance system. Within each part, the requirements are identified, listed, and expanded according to the criteria that must be met in order for the EEMIS-S source language programs to be successfully transformed to the required executable load modules.

6.1 INFORMATION RETRIEVAL SYSTEM

The compiler program must satisfy certain program language requirements in order to recognize the source statements for the EEMIS-S Information Retrieval System and faithfully to convert their intelligence to the required object program. These requirements have been identified and are grouped within the following major categories:

- Program Statements
- Program Organization
- Operational Statements
- Data Management.

The specific requirements for each of the categories and their associated criteria are described in the following subsections. These requirements can be met to the extent of the criteria through the use of a PL/I compiler of latest version and level or its equivalent.

6.1.1 Program Statements

The compiler program must support certain program statements, structures and configurations. It must also be able to recognize the various syntax, formats, comments, abbreviations, and assignments. These are expanded upon in the following paragraphs.

- Structure - The compiler must be able to recognize a continuous stream of statements. Each line may contain a part of, a single or several statements, each separated by a delimiting character. Interspersed blanks are accepted, however statement labels must begin with an alphabetic character. The length of label names may vary from one to thirty-one alphanumeric and break characters.
- Comments - Comments can appear separately or within a statement and are enclosed by beginning and ending delimiters each two characters in length.
- Abbreviations - Various compiler keyword abbreviations can exist. The compiler must be able to recognize the abbreviated (alias) form or the expanded form.
- Assignment - The assignment statement assigns to the variable on the left the value of the expression on the right. One or more variables can be placed on the left of an assignment statement to receive the value on the right.
- Pseudovariables - A function subroutine name supported by the compiler can exist on the left side of an assignment statement as a pseudovvariable.
- Variables - Variable names may appear anywhere within a statement. They may contain up to thirty-one alphanumeric and break characters with the provision that the first character be alphabetic. Variable

names beginning with the letters I through N default to the data type of binary fixed point values with minimum precision of $+(2^{16}-1)$ and $-(2^{16})$. Variable names beginning with the letters A through H and O through Z, \$, #, and @ default to decimal floating point values with six decimal digits of precision.

- Control - Conditional and unconditional control statements can be used. Conditional transfer of program control can occur by assignment, by conditional relationships or by logical tests. For conditional transfer by assignment, subscripted labels can occur. For conditional transfer by conditional relationships or logical tests, other statements can be grouped under "if, then" for a true condition or "if, else" for a false condition. Parenthetical and nested conditional statements can occur.
- Subroutine - Subroutines consist of a collection of self-contained statements that can be compiled separately from other parts of the program. Entry to the subroutine is accomplished by a CALL or a function reference statement accompanied by the subroutine name. Exit from the subroutine will occur upon the recognition of a return or an END statement. Values may be passed from the calling program to the subroutine, and in the reverse order through lists of arguments following the subroutine name, or through a previously defined common data storage area. A subroutine name may appear in the argument list of another subroutine.

6.1.2 Program Organization

The compiler program must support various configurations of source program organization. It must be able to recognize groups and/or blocks of statements and take the necessary steps to establish or reorganize data storage areas and/or initialize and establish control procedures for repetitive execution of statement groups. Program organization requirements are expanded upon in the following paragraphs.

- Blocks - Blocks are executed in sequence like single statements, and must be entered at the beginning statement. Block exits can be made from an ending statement or from a transfer statement. Names defined within a block are known only within the block, but names defined in an outer block are also known within the inner block. Automatic storage is allocated when a block is entered, and the storage is released upon exit from the block.
- Groups - Groups consist of statements enclosed between a starting and ending pair, and may be placed wherever a single statement may go. They can be associated with conditional statements so as to execute or avoid a sequence of statements based on the results of a conditional test. They may also be indexed for repetitive execution and controlled by step variables and limiting variables. Groups of indexed instructions can be nested.
- Internal Functions - Functions and subroutines can be internal procedures, and can be placed anywhere within the source program. The compiler must recognize these and provide for transfer around them and linkage to them where required.

6.1.3 Operational Statements

The compiler program must be able to recognize certain operational sequences encoded in the source language statements and faithfully to convert them to object coded instruction sequences which will perform a desired function. Specifically, the compiler program must be able to recognize relational operations, logical operations, arithmetic operations, bit string operations, and character string operations. These are expanded in the following paragraphs.

- Relational Operations - Relational operations are expressed symbolically and are composed of the conditional operators of less than, greater than, less than or equal to, greater than or equal to, equal, not equal, not less than, and not greater than.

- Logical Operations - Logical operations are expressed symbolically and are composed of the conditional operators of AND, OR, and NOT.
- Arithmetic Operations - Arithmetic operations are expressed symbolically and are composed of operators that define the functions of addition, subtraction, multiplication, division and exponentiation.
- Bit-String Operations - The logical operations previously defined can be applied to contiguous bit strings within a single operational statement.
- Character-String Operations - A single string of characters can be created from two or more concatenated strings within a single operational statement. In addition, the relational operations and/or the logical operations can be applied to full character strings or character-string segments within a single operational statement.
- Hierarchy of Operations - The compiler must generate object code that will conform to a hierarchical evaluation of expressions. These are, first from right to left, ordered as: function subroutines, exponentiation, prefix sign, and logical NOT. Second, from left to right, they are ordered as: multiplication and division, addition and subtraction, concatenation, relational operators, logical AND, and logical OR. Nested parenthetical expressions can be used to modify the hierarchy and are evaluated from the innermost closed parentheses to the outermost closed parentheses.

6.1.4 Data Management

The compiler program must be able to organize and classify the various types of data specified in the source language statements and allocated the appropriate processor storage areas for the internal management of the data. The compiler programs must also be able to recognize source language statements that will initiate the conversion of data from one type to another and the transfer of data from one storage medium to another. The compiler requirements for the management of data are expanded in the following paragraphs.

- Data Types - The compiler must be able to implicitly and explicitly recognize source statements that define data types for numeric constants, alphanumeric constants and variable names. The various types of data can be represented in an internal coded structure generated by the compiler as decimal fixed point numbers, binary fixed point numbers (integer), real floating point numbers, double precision floating point numbers, logical variables, character literals, character variables, character string expressions, and varying-length character strings.
- Data Conversion - The compiler must support the provision to convert data automatically when operations are performed on different types of data. The hierarchy of conversion, from lowest to highest is in the order of bit strings, character strings, binary fixed point numbers, decimal fixed point numbers, floating point numbers, double precision floating point numbers and complex numbers.
- Data Declaration and Storage - The compiler must be able to recognize source-language statements that declare the type and extent of internal data storage. It must be able to implicitly and/or explicitly identify data types. In addition, it must be able to allocate areas of internal data storage ranging in size from single variables to arrays of up to fifteen dimensions.
- Input/Output - The compiler must be able to recognize and construct object code for source language statements that direct the transfer of data from one device to another. These statements can identify and direct either record oriented or stream oriented transfers. Record-oriented transfer statements allow data to be transferred in the form of one or more complete logical records during any given execution. Stream-oriented statements allow data to be transferred in the form of continuous streams of characters, ignoring physical record boundaries such as line size or card columns. Stream-oriented I/O can be processed in three modes; data-directed, list-directed and edit-directed.

6.2 FILE MAINTENANCE SYSTEM

The EEMIS-S File Maintenance System is written in a high-level computer language which is transaction oriented and relatively machine independent. The compiler program must recognize this language and convert its intelligence to the required object programs. The requirements for the compiler have been identified and are grouped within the following major categories.

- Program Statements
- Program Organization
- Processing Modules.

The specific requirements for each of the categories and their associated criteria are described in the following subsections. These requirements can be met to the extent of the criteria through the use of an ANSI COBOL compiler of latest version and level or its equivalent.

6.2.1 Program Statements

The compiler program must support certain program statements, structures and configurations. It must also be able to recognize the various syntax, formats, reserved words and qualifications. These are expanded upon in the following paragraphs.

- Structure - The program language is based on English; it uses English words and certain syntax rules derived from English. Functional statements (commands) are constructed as sentences which may be composed of many statements. Each sentence is terminated with a period. In the procedural form, statements

contain standard action words and user defined words to be acted upon. User defined words are composed of a combination of not more than thirty alphanumeric and break characters.

- Reserved Words - Reserved words exist for syntactical purposes and will not appear as user-defined words. There are three types of reserved words. These are, key words, optional words, and connections. A key word is a word whose presence is required in a statement. Included in these are the action words such as READ, WRITE, ENTER, ADD, SUBTRACT, etc. Also included are words that have specific functional meaning such as ZERO, NEGATIVE, SECTION, TALLY, etc. Optional words are words that are entered for clarity at the users option. These include IS, ON, WITH, etc. Connective words are used to qualify as in the use of, OF and IN, to define logical relationships as in the use of AND, OR, NOT, etc., and to define a series, as through the use of commas.
- Names - There are three types of names used in the source language. They are data names, condition names and procedure names. A data name is a user defined word that contains at least one alphabetic character and identifies a data item. A condition name is a user defined name given to a specific value, set of values or range of values. A procedure name is either a paragraph name or a section name. A procedure name may be composed solely of numeric characters.
- Constants - There can be two kinds of constants. The first are user-defined and may consist of either numeric values or character strings. Character strings, also known as non-numeric literals, are not used for computational purposes. The second are figurative constants and are identified by reserved names. These include ZERO, SPACE, HIGH-VALUE, LOW-VALUE, etc.

- Qualification - A name may be made unique if the name exists within a hierarchy of names and the name can be singled out by mentioning one or more of the higher levels of the hierarchy. The higher levels are called qualifiers. Qualification is applied by placing after a data-name or a paragraph-name one or more phrases, each composed of a qualifier preceded by IN or OF. IN and OF are logically equivalent. Only one qualifier is allowed for a paragraph-name. The name associated with a level indicator is the highest level qualifier available for a data-name. (A level indicator (FD, SD, RD) specifies the beginning of a file description, sort-file description, or report description.) A section-name is the highest (and the only) qualifier available for a procedure-name. Thus, level indicator names and section-names will be unique in themselves since they cannot be qualified. The name of a conditional variable can be used as a qualifier for any of its condition-names. In addition, a conditional variable may be qualified to make it unique. Each qualifier may be of a successively higher level, and may be within the same hierarchy as the name it qualifies. The same name may not appear at two levels in a hierarchy so that the name would appear to qualify itself. A paragraph-name will not be duplicated within a section. When a paragraph-name is qualified by a section-name, the word SECTION will not appear. A paragraph-name need not be qualified when referred to within the section in which it appears. A data-name will not be subscripted when it is being used as a qualifier. A name can be qualified even though it does not need qualification; if there is more than one combination of qualifiers that ensures uniqueness, then any of these combinations may be used.
- Subscripting - Subscripts may be used only when reference is made to an individual element within a list or table of elements that have not been assigned individual data-names.
- Indexing - References may be made to individual elements within a table of elements by specifying indexing for that reference. An index is assigned to a given level of a table by using an INDEXED BY clause in the definition of the table. A name given in the INDEXED BY clause is known as an index-name and is used to refer to the assigned index.

6.2.2 Program Organization

The compiler must be able to support the specific configuration of source program organization. This configuration requires four distinct divisions. Each has its necessary logical function and each uses information developed in the divisions preceding it. These organizational requirements are expanded upon in the following paragraphs in the order of their logical sequence.

- Identification Division - The Identification Division provides introductory documentation to the source program. Specifically it identifies the program by name, the programmer or development team by name and the date of its creation or revision. The program name is recognized by the compiler as being unique and provides the vehicle to pass the name on to the object program module.
- Environment Division - Although the language is, to a large degree, machine independent, there are some aspects of the program that depend on the particular computer being used and on its associated input/output devices. In the Environment Division, the characteristics of the computer used are identified. The location of each file referenced in the program, and how each one of them will be used, must be described. To fulfill this requirement, two distinct sections are used; the Configuration Section and the Input-Output Section.
- Data Division - The Data Division gives a detailed description of all the data to be used in the program; whether to be read into the machine, used in intermediate processing, or written as output. It is a requirement of the Data Division to establish internal storage areas for either single variables, input/output arrays and indexed arrays. Further, the Data Division is used to define numeric constants and literal strings of alphanumeric characters to be used during processing.

- Procedure Division - The Procedure Division contains the instructions or sentence-like statements needed to solve the application problem. These include arithmetic statements, conditional statements, data-manipulation statements, input/output statements, procedural branching statements and statements to open and close files. The Procedure Division is the final division in the source program.

6.2.3 Processing Modules

The compiler must support the processing modules defined by the American National Standards Institute (ANSI). These processing modules are identified in the following paragraphs.

- Nucleus - Defines the permissible character set and the basic elements of the language contained in each of the four divisions: Identification Division, Environment Division, Data Division, and Procedure Division.
- Table Handling - Allows the definition of tables and reference to them through subscripts and indexes.
- Sequential Access - Allows the records of a file to be read or written in a serial manner. The order of reference is implicitly determined by the position of the logical record in the file.
- Random Access - Allows the records of a file to be read or written in a manner specified by the programmer. Specifically defined keys, supplied by the programmer, control successive references to the file.
- Sort - Provides the capability of sorting files in ascending and/or descending order. This also includes procedures for handling such files both before and after they have been sorted.
- Report Writer - Allows the programmer to describe the format of a report in the DATA DIVISION, thereby minimizing the amount of PROCEDURE DIVISION coding necessary.

- Segmentation - Allows large problem programs to be split into segments that can then be designated as permanent or overlayable core storage. This assures more efficient use of core storage at object time.
- Library - Supports the retrieval and updating of pre-written source program entries from a user's library, for inclusion in the program at compile time. The effect of the compilation of library text is as though the text were actually written as part of the source program.

7.0 UTILITY SOFTWARE

The vendor is required to supply certain utility software (utilities) which will support the EEMIS-S system during the file maintenance and updating process. Specifically two types of utilities are required. They are:

1. A sort/merge system
2. A file handling system.

Within each of these there are specific functional requirements which must be met. These requirements are identified and expanded upon in the following subsections.

7.1 THE SORT/MERGE SYSTEM

The vendor is required to supply a sort/merge utility program to be used for the proper sequencing of EEMIS-S data during certain file maintenance operations. This utility must satisfy the minimal support requirements listed as follows.

- To support various input/output devices
- To support various file types
- To support various record types
- To support various configurations of sorted order.

Each of the requirements listed above must meet the criteria as described in the following paragraphs. These requirements and their criteria can be met through the IBM utility IERCO000 or its equivalent.

7.1.1 Input/Output Devices

The sort/merge utility must be able to service the full complement of input/output (I/O) devices included in the vendors hardware environment and must operate, for the most part, as a device independent system. It is mandatory, however, that the system access both disk-resident files and tape-resident files and that these be acceptable for assignment in any one of a number of I/O configurations. Further the sort/merge utility must be able to accept the assignment of multiple files and multiple devices for both input files and output files.

7.1.2 File Types

The sort/merge utility must be able to support the various file types identified by their method of access. Specifically, the utility is required to support the direct, sequential, and index sequential access methods for disk files and the sequential access method for tape files.

7.1.3 Record Types

The sort/merge utility must be able to support files of various types of records. Specifically, the utility is required to support files composed of either fixed-length records in blocked or unblocked format or variable-length records in blocked or unblocked format. Further, the utility must be able to re-block records routed to the output file according to user specifications. It is also required that the utility recognize header-label records and trailer-label records in both standard and user-defined formats.

7.1.4 Sort Configurations

The sort/merge utility must be able to reorganize input data records according to user specifications and produce a resultant file of output records. It is required that the utility be able to support user specifications which will direct the utility to accomplish any one or all of the following.

- Control the sorted order of the output file using one or more (multiple) user-defined, sort-control fields. The sort-control fields will reside within the data record and be recognized in any hierarchical order as defined by the user.
- Select the proper sequence of consecutive records based on user-specified comparisons of data within each sort-control field. On a field for field basis the specifications must include the definition of ascending or descending sorted order and comparisons on fields defined as numeric, binary, alphanumeric (character), and logical; each of user defined length.
- Conditionally select or reject input records from the sorted output file based on user specifications of data limits or values within user defined fields.

7.2 THE FILE HANDLING SYSTEM

The vendor is required to supply utility programs capable of copying and otherwise manipulating files according to user specifications. These utilities must be able to satisfy the minimal support requirements listed as follows.

- To support various input/output devices
- To support various file types
- To support various record types
- To support editing and reformatting functions.

Each of the requirements listed above must meet the criteria as described in the following paragraphs. These requirements and their criteria can be met through the IBM utilities IEBGENER and IEHMOVE or their equivalents.

7.2.1 Input/Output Devices

The file handling utilities must be able to service the full complement of input/output (I/O) devices included in the vendors' hardware environment and must operate, for the most part, as a device-independent system. It is mandatory, however, that the systems access both disk resident files and tape resident files and that these be acceptable for assignment in any one of a number of I/O configurations. Further the utilities must be able to accept the assignment of multiple files and multiple devices for both input files and for output files.

7.2.2 File Types

The file-handling utilities must be able to support the various file types identified by their method of access. Specifically, the utility is required to support the direct, sequential, and index-sequential access method for disk files and the sequential access method for tape files.

7.2.3 Record Types

The file handling utility must be able to support files of various types of records. Specifically, the utility is required to support files composed of either fixed-length records in blocked or unblocked format or variable-length records in blocked or unblocked format. Further, the utility must be able to reblock records routed to the output file according to user

specifications. It is also required that the utility recognize header-label records and trailer label records in both standard and user-defined formats.

7.2.4 Editing and Reformatting

The file handling utilities must be able to edit and/or reformat input data records according to user specifications and produce a resultant file of output records. It is required that the editing function include the removal and/or addition of data fields within each record and that the reformatting function include the rearrangement of data fields within each record.

8.0 JUSTIFICATION FOR PROCUREMENT UNDER TSP-MASC

The EEMIS-S system is intended to reside in a national timesharing vendor's environment. An RFP for timesharing services under the GSA-TSP guidelines will be made available to all interested vendors. The guidelines explain GSA's policies for the acquisition of time-sharing services and hence the methods of RFP preparation. Presently there are two methods. They are:

- TSP Schedule Contracts (MASC)
- TSP Basic Agreement (BA).

The TSP guidelines recommend that the MASC should be used if:

1. The selecting organization's full technical and system life requirements can be met, and
2. The nature of the requirement is such that there is no reasonable expectation of obtaining a lower price from negotiated procurement.

The guidelines also recommend that the BA should be used if:

1. The requirement cannot be met by the TSP schedule contracts, (such as a projected annual (system life) cost which exceeds the anticipated maximum order limitation of most TSP schedule contractors), or
2. The nature of the requirement is such that there is a reasonable expectation of obtaining a lower price from the competitive (BA) procurement.

Under the TSP-MASC the recommended maximum threshold for the costs of vendor supplied services is \$500,000 annually or \$1,500,000 for the system's life. Although GSA has established these and other parameters as a guide to the selection of a procurement method there are no absolute or rigid rules to be followed. If, however, a decision in the selection of a procurement method appears to be incorrect or capricious, and without justification, GSA may not approve the release of the RFP.

This section presents the justifications for the DOE Office of EEMIS' decision to select the TSP-MASC method of procurement.

8.1 PROCUREMENT LEAD TIME

A distinct advantage to MASC is the opportunity to reduce procurement lead time. Because of the nationwide urgency in the need for accurate and timely energy data, it is desired that EEMIS-S be available to state-level users by June 1981. This objective cannot be met under the guidelines of the TSP-BA procurement process. It can, however, be met under the TSP-MASC.

8.2 RESPONSE TIME

EEMIS-S is an emergency management system that provides energy information to federal, state, and local government officials in crisis situations. This system must, by definition, be highly responsive to its users. Presently the DOE EIA computing facility is overburdened and lacks responsiveness. Hence, the facility is presently undergoing the prescribed studies and analyses prerequisite to upgrading its equipment and services. It is expected that their process will require eighteen months to complete. In the meantime, the EIA computing facility remains less than desirable in meeting the emergency response requirements of EEMIS-S.

While the EIA facility is being upgraded, EEMIS-S would best serve its users in a national timesharing vendor's environment where twenty-four hour system availability and immediate system response can be assured.

8.3 SYSTEM LIFE

The active system life cycle for EEMIS-S in the timesharing vendor's environment is considered as a temporary measure and

is expected to last for a period of eighteen months. Following that the EIA computing facility should be able to support the EEMIS-S response requirements.

Under the TSP-BA guidelines it is recommended that the minimum life cycle for systems operating in the vendor's environment be a period of three years or more. Otherwise the TSP-MASC is recommended. It is obvious that if the life cycle of the system is short the additional time and expense involved in the BA procurement may well undermine the intended purpose and quite possibly negate any savings realized from a more competitive bid.

8.4 SYSTEM BACKUP

Because EEMIS-S responds to emergency situations it is highly desirable that a backup system exist. Even upon the return of the active EEMIS-S system to the EIA facility the vendor resident system can be allowed to lie dormant with the exception of periodic updates to maintain its data base. Thus the vendor resident system, though inactive, can remain ready to respond to emergency situations as it would in the temporary loss of the EIA facility.

The TSP-MASC procurement method is ideal for establishing backup systems. Unlike the BA, where the vendor will bid on some guaranteed or expected workload, the MASC will provide a mechanism for the constant availability of the system without constant costs; these to be incurred only at the time of its use.

8.5 SECONDARY ADVANTAGES

The justifications for the use of the TSP-MASC procurement method as presented in Subsections 8.1 through 8.4 carry with them certain secondary advantages. These are as follows:

8.5.1 Government-wide Volume Discounts

EEMIS-S is based on a very large resource of machine readable data. This volume will encourage vendors to use data storage pricing structures that will be advantageous to the government. In addition the government will be able to realize a greater overall volume discount.

8.5.2 Reduced Administration Costs

Inherent to the TSP-MASC method of procurement is the reduction of the contract administration costs. The EEMIS-S system requirements are easy to satisfy and therefore the applied purchasing method of fixed-unit pricing with government-wide volume and quantity discounts is expected to reduce these costs considerably.