RADC-TR-76-259
Final Technical Report
August 1976

TRANSPARENT INTELLIGENCE LANGUAGE FACILITY

INCO, INC.

Approved for public release;
distribution unlimited.

ROME AIR DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
GRIFFISS AIR FORCE BASE, NEW YORK 13441

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

This report has been reviewed and approved for publication.

APPROVED: Patricia M. Langendori

PATRICIA M. LANGENDORI
Project Engineer

APPROVED: Howard Davis

HOWARD DAVIS
Technical Director
Intelligence & Reconnaissance Division

FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

Do not return this copy. Retain or destroy.

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: RADC-TR-76-259	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): TRANSPARENT INTELLIGENCE LANGUAGE FACILITY	5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report
	6. PERFORMING ORG. REPORT NUMBER: N/A
7. AUTHOR(s): Lelia Irby, Karen Hsing, John Voss	8. CONTRACT OR GRANT NUMBER(s): F30602-75-C-0347
9. PERFORMING ORGANIZATION NAME AND ADDRESS: INCO, INC. 7916 Westpark McLean VA 22101	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62702F 45941219
11. CONTROLLING OFFICE NAME AND ADDRESS: Rome Air Development Center (IRDA) Griffiss AFB NY 13441	12. REPORT DATE: August 1976
	13. NUMBER OF PAGES: 59
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office): Same	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

16 AF-459H 17 459H12

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)
Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer:
Patricia M. Langendorf (IRDA)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Network	Interactive Query	Query Responce
Data Sharing	Data Location	Normalization
Data Base Transparency	Query Translation	
Data Description	Query Distribution	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This final technical report presents the system architecture and functional objectives for a Transparent Integrated Intelligence Network (TIIN) and the design objectives and capabilities of the (TIIN) Data Description software which was produced for contract F30602-75-C-0347.

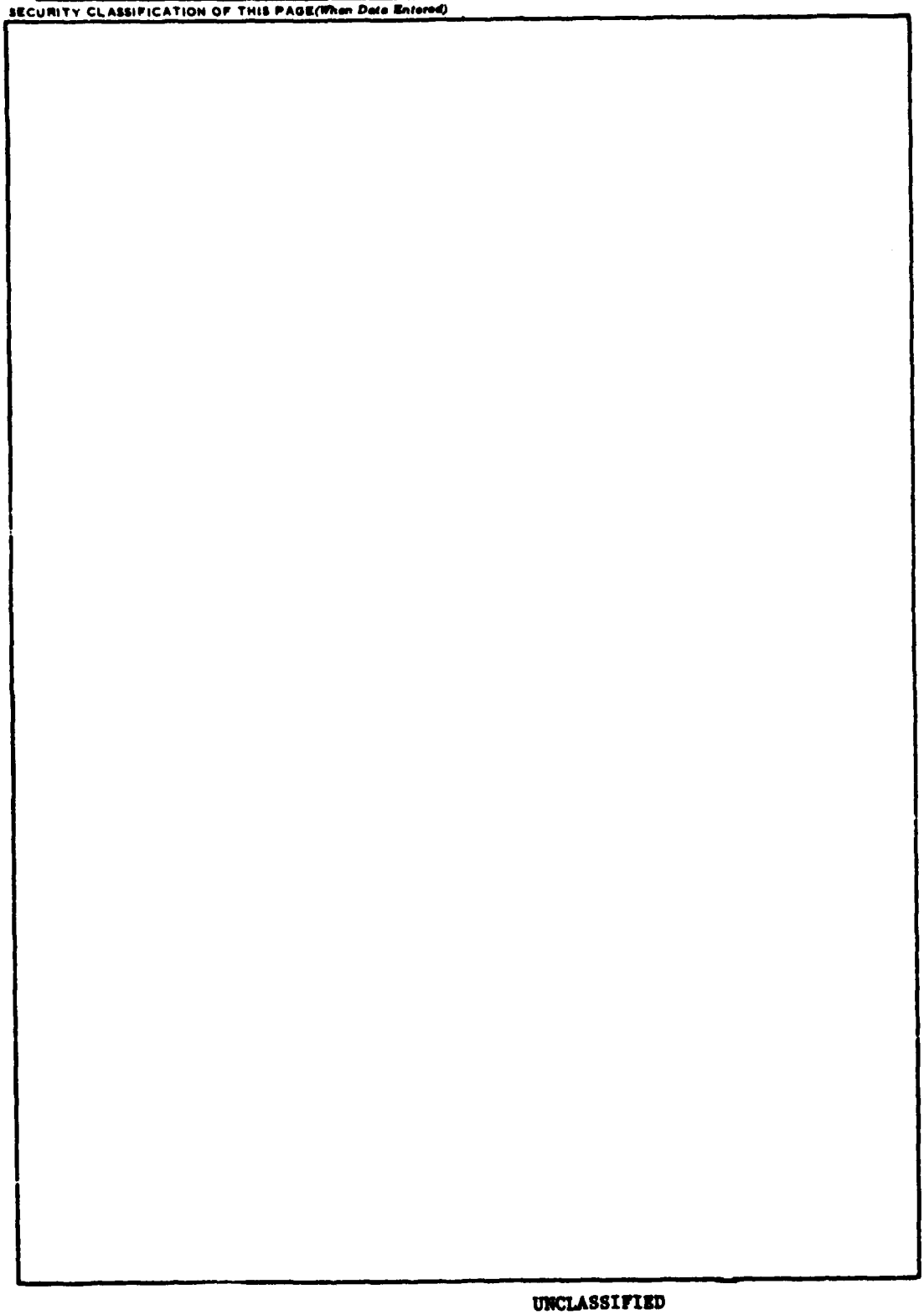
DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

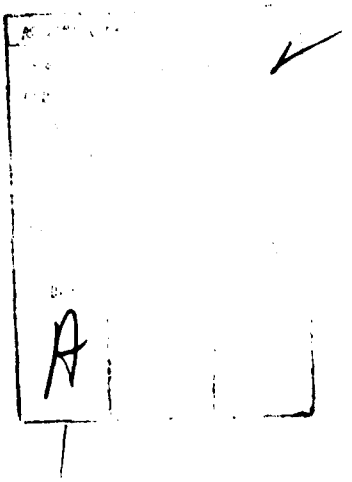
407275

LB



ABSTRACT

The Transparent Intelligence Language Facility addresses the need to share data residing in widely-distributed and discretely-maintained intelligence data bases. The system architecture and functional description of a Transparent Integrated Intelligence Network based on the TOSS concepts of transparency and distributed processing are presented in this report. Also included are descriptions of the design features and functional capabilities of the network system software developed to provide up-to-date information about data residing in data bases accessible to the network.



PREFACE

This final technical report on the Transparent Intelligence Language Facility has been provided by INCO, INC. of McLean, Virginia, under the cognizance of Rome Air Development Center (RADC), Griffiss Air Force Base, Rome, New York. This document presents the system architecture and functional objectives for a Transparent Integrated Intelligence Network (TIIN) and the design objectives and capabilities of the TIIN Data Description software which was produced for contract F30602-75-C-0347.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

EVALUATION

This effort is an integral part of RADC's long-range development of effective processing capabilities to support intelligence analysts.

It is the first effort concerned with development of a network capable of linking existing intelligence data bases, each of which was developed independently with it's own hardware, data managenment system, access procedures, and query languages. The hardware vehicle is the AN GYQ-21V. The software goal is to enable authorized users to access the disparate data bases using a single query language.

Many technological problems associated with this proposed software are still unsolved. This effort developed a logical plan to attack these problems and successful solutions are anticipated.

Patricia M Langendorf

PATRICIA M. LANGENDORF
Project Engineer

SECTION I

INTRODUCTION

1. OVERVIEW

This Final Technical Report details the design and development of the Transparent Intelligence Language Facility under Rome Air Development Center Contract Number F30602-75-C-0347. The project is an integral part of the long-range development of a Transparent Integrated Intelligence Network(TIIN), a highly user-oriented distributed data access and processing capability encompassing world-wide resources. The purpose of this report is to provide a thorough, accurate account of the developmental effort to date, and to provide a baseline for further efforts in this area. The Final Technical Report is the culmination of the present TILF effort, an outgrowth of INCO's on-going work in developing a Terminal Oriented Support System (TOSS). Relevant documents which address the current TILF development are a planning paper,[1] a proposal,[2] a development plan,[3] a management plan, and a series of memoranda.

2. DEVELOPMENT APPROACH

The TILF Project was INCO's first contract effort concerned with the development of the Transparent Integrated Intelligence Network. During the first part of this effort, project personnel researched and analyzed the general problems presented by the fact that a network of intelligence data bases would be required to link and integrate a number of existing data bases, each of which was developed independently with its own hardware, data management system, access procedures and query language. Before any actual development could be undertaken, it was necessary to clarify the network concepts, research user requirements, and produce a proposed system architecture and management plan for the staged development of the network.

[1] Development Planning Factors for a Transparent Integrated Intelligence Network, INCO, INC., 1974

[2] Technical Proposal for a Transparent Integrated Intelligence Network, INCO, INC., 5 March 1975.

[3] Transparent Integrated Intelligence Network Developent Plan, INCO, INC., 15 November 1975.

In November 1975, TILF project personnel produced an interim report, the Transparent Integrated Intelligence Network (TIIN) Development Plan, which proposed linking the various intelligence data bases on their own host computers through PDP-11/45 front-end processors and presented an initial architecture for a software system to reside in each of the minicomputer network nodes. The proposed system architecture is that shown in Figure I-1. On the input side of the minicomputer system are intelligence analysts who are experts in the analysis of intelligence data, not in the complexities of computerized data management; and on the output side are communication lines connecting the minicomputer with multiple data bases, each capable of responding only to queries phrased in a specific and, perhaps, unique language. The objective of the Transparent Integrated Intelligence Network is to act as nexus between the two, making it appear to an analyst that he is conversing comfortably with a single data base when, in fact, he is conversing with the TIIN software which interprets his data needs, finds out where in the network the required data is located, and sends out properly phrased queries to retrieve the data from the appropriate network data bases.

The technological problems associated with the development of the proposed software are many and varied. Some of these problems have been handled for specific network configurations or for specific areas of application and some have never been solved effectively in a practical application. In fact, designing and developing software to deal with any one of them constitutes a major research and development effort in itself. Consequently, the TIIN Development Plan included a logical division of the software development into several separate projects and specifically outlined the software development task to be performed during the remainder of the TILF contract.

3. PROJECT OBJECTIVES

The objectives of the TILF project were:

- To research and analyze the problems associated with the sharing of data in a network of intelligence data bases;
- To produce functional specifications for a Transparent Integrated Intelligence Network (TIIN) based on the TOSS concepts of transparency and distributed processing;
- To outline an orderly plan for the development of new software required to support the TIIN; and
- To design and develop the software and data structures associated with the data base description capability of the TIIN software.

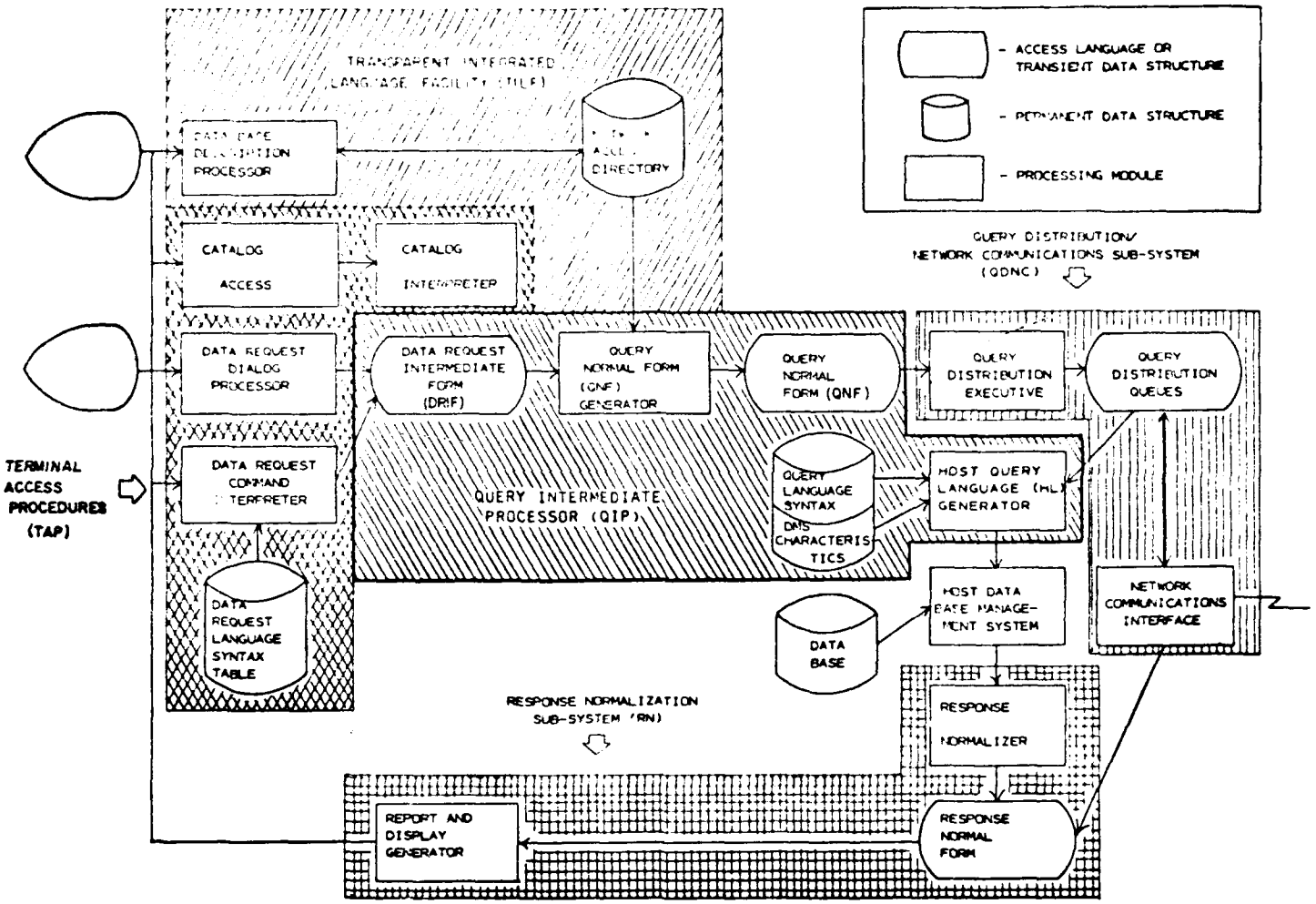


Figure I-1. Transparent Integrated Intelligence Network (TIIN)

4. RESEARCH AND DEVELOPMENT ACTIVITIES

The following research and development activities were performed in compliance with the contract:

- o Analysis of user requirements;
- o Analysis of system requirements;
- o Functional-level specification of system architecture;
- o Delivery of TIIN Development Plan;
- o Design of a Network Access Directory to supply information about data bases;
- o Design and implementation of a Data Description Processor to create and update the Network Access Directory from data descriptions supplied on-line by local data base administrators;
- o Delivery of final project documentation.

5. DEVELOPMENT CONFIGURATION

TIIN system specifications are based on a local node configuration consisting of a PDP-11/45 (AN/GYQ-21(V) processor equipped with the TOSS Standard Software Base. The development configuration for the Data Description Processor included the following hardware and software:

- o PDP-11/45 (AN/GYQ-21(V) Computer w/128K Memory
- o LP11 Line Printer
- o CR11 Card Reader
- o RPØ3 Disk Pack
- o LA30 DECwriter
- o Tektronix Terminal
- o TOSS Standard Software Base, including RSX-11D, ICM, TTDL, and File Handler.

6. REPORT ORGANIZATION

This report describes the TILF research and development work which was performed between November 1975, when the interim report was produced, and the end of the contract period.

The following paragraphs summarize the sections into which the document is organized.

Section I. The first section is an introduction to the TILF effort as detailed in the Final Technical Report. It places the project in context and is intended to make that which follows more readily understood by the reader. Important concepts in the TILF design are identified and discussed.

Section II. Section II defines the scope of the software development undertaken for the TILF project, the design and implementation of a data base description function for the TIIN.

Section III. Section III defines the structure and content of the Network Access Directory.

Section IV. Section IV describes the process by which local data bases are described to the TIIN system for network access.

Section V. Section V addresses those problems which are outstanding as of the end of the contract and which constitute challenges for the ongoing TIIN/TILF design. These stress value translation and the handling of errors.

Section VI. This section describes the system design. Those considerations which affect system design are discussed. The network configuration is detailed and includes TIIN and communications software relating to TILF. The system architecture is illustrated graphically. Human support and user/analyst interfaces with the system are discussed.

Appendix A. This appendix details some of the background and requirements which led to the development of the TIIN and TILF concepts.

Appendix B. This appendix provides functional descriptions of the modules included in the TIIN system design.

SECTION II

THE DATA DESCRIPTION FUNCTION

1. DEVELOPMENT OF THE DATA DESCRIPTION FUNCTION

The development task outlined for TILF was to design and implement a method for creating and maintaining up-to-date information about the data residing in individual data bases available to the network, including:

- Specific names and network synonyms for data elements;
- Names of the files to which the data elements belong;
- Identification of data bases to which files belong, location of data bases, and native query language by which they are accessed;
- Information to help in validating the usage of element names and values associated with specific elements.

A repository for this information, the Network Access Directory (NAD), was designed, and a technique for creating and updating the NAD from data base descriptions supplied on-line by the local administrator of each data base was developed. Software produced under the TILF contract processes descriptive information entered at a terminal and creates and updates the NAD, which is the only interface between the TILF software and other system modules.

Section IV of this report describes the processing of the data description function, and Section III defines the content and structure of the NAD.

2. THE USER OF THE DATA DESCRIPTION FUNCTION

Unlike the network query and retrieval functions available at a TIIN node, which are intended for use by any analyst who has access to the local computer facility, the only authorized user of the data description function is the local data base administrator (DBA) for the node. Data base administration is an important human function in a computerized environment where the data base includes data which is shared by many users. In such an environment, the data base is a compromise between the needs of various users and requires a means of mediating conflicting needs. The data base administrator's function is to create and maintain

both the data base and its data structure schema in such a way that the data base may satisfy efficiently the data requirements of all of its users. This function may include the following tasks:

- Organizing, that is designing and assembling, the data base;
- Loading the data base;
- Assigning privacy locks and issuing privacy keys to users who need to use portions of the data base;
- Monitoring the use and performance of the data base;
- Reorganizing and restructuring the data base so as to improve its performance;
- Recovering the data base after system malfunctions.

The linking of discretely-maintained data bases with networking techniques will inevitably extend the data base administration function to include legislating decisions governing what local data is to be made available to network users and providing sufficient information about local data to make it accessible to the network.

SECTION III

THE NETWORK ACCESS DIRECTORY

1. COMPOSITION OF THE NAD

The Network Access Directory (NAD) is composed of five files: a file called BASES, containing information about data bases and files; a file called ELMNTS, containing descriptive information about data elements and synonyms; a file called NAMES, which is used to search for an element by name; a file called VALUES which contains information about legal values for selected elements; and a file called TEXT, which contains prose descriptions of data bases, files, and elements. Each NAD file name is modified with NAD, and local versions of the NAD files have version numbers corresponding to a site code assigned to the node. For example, the NAD files for the TIIN Node with site code ØØ1 are named as follows:

BASES.NAD;ØØ1
ELMNTS.NAD;ØØ1
NAMES.NAD;ØØ1
VALUES.NAD;ØØ1
TEXT.NAD;ØØ1

Each data base defined at the local node, whether it resides on the minicomputer or on a host system, has an entry in the site's data base table in the BASES file. In addition, each data base has its own file table, element table, value table, and text table. A file, element, value, or text table consists of a chain of blocks in the corresponding file; file tables are contained in the BASES file, element tables in the ELMNTS file, value tables in the VALUES file, and text tables in the TEXT file. Each site also has a table of local usage synonyms for real element names, which is a chain of blocks in the ELMNTS file; and a name table which is a sorted list in the NAMES file used to search for any name or synonym defined locally. File blocks which are chained to form individual tables are linked through the last word of each block, which contains a number corresponding to the virtual block number (VBN) of the next block in the chain.

2. THE BASES FILE

The BASES file consists of four parts:

- o A table of local configuration information, of which all is used by the local description module and part is of significance to the data location modules at all network nodes
- o A table of fixed-length entries, each containing information about a data base available at the local node, either on the minicomputer or on a host system

- o A table of variable-length data base name entries
- o Tables of variable-length file description entries.

Of the tables described above, the first two reside on virtual block number (VBN) 1, the third resides on VBN 2, and the fourth occupies VBNs 3-n.

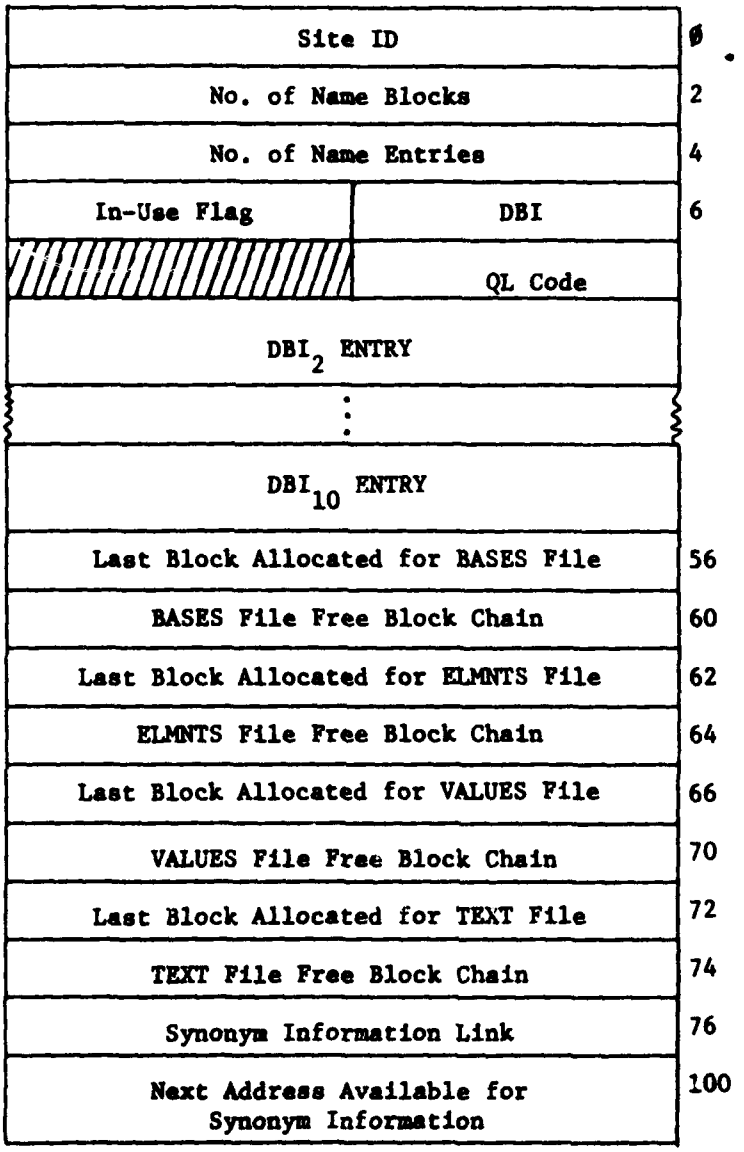
a. Virtual Block Number 1

Virtual block number 1 of the BASES file contains the SITE table and the DBTAB table. These two tables are read into core at the beginning of a description session and remain in core for the entire session.

(1) SITE Table

The SITE table, which occupies the first 68 bytes of VBN 1, has the following structure:

Starting Byte (Octal)	Size (Decimal # Bytes)	Contents
0	2	Unique numeric code assigned to the local site when it becomes a participating network node.
2	2	Total number of blocks in the local element name table (NAMES.NAD;nnn).
4	2	Total number of entries in the local element name table.
6	40	Up to ten 4-byte entries, each designating a unique DBI which may be assigned to a data base residing at the local node. The DBI codes are entered when the node goes on-line to the network. Each entry contains a DBI and a flag indicating whether the DBI has been assigned. For assigned DBI's, the entry contains a code designating the native query language for the data base.
56	2	VBN of the last block allocated for this file (BASES.NAD;nnn).

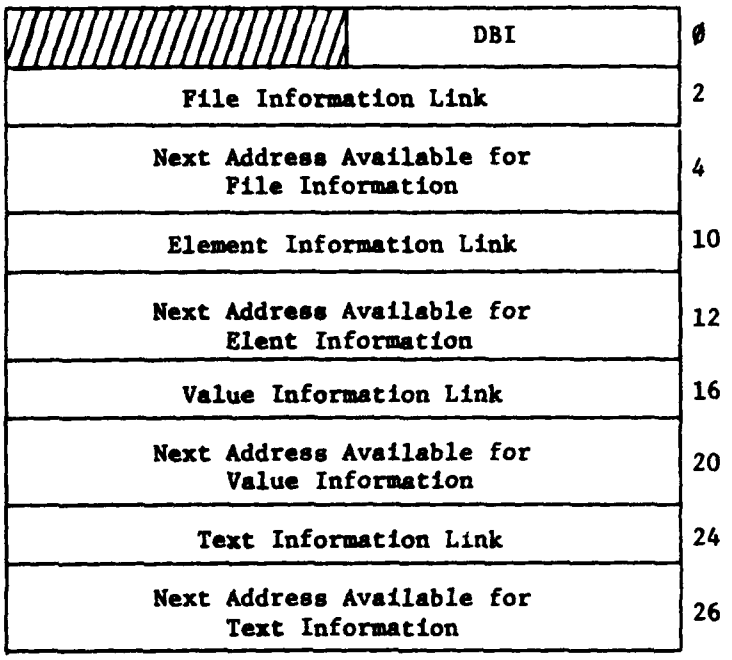

Site ID
Ø
No. of Name Blocks
2
No. of Name Entries
4
In-Use Flag
DBI
6
QL Code
DBI2 ENTRY
DBI10 ENTRY
Last Block Allocated for BASES File
56
BASES File Free Block Chain
60
Last Block Allocated for ELMNTS File
62
ELMNTS File Free Block Chain
64
Last Block Allocated for VALUES File
66
VALUES File Free Block Chain
70
Last Block Allocated for TEXT File
72
TEXT File Free Block Chain
74
Synonym Information Link
76
Next Address Available for Synonym Information
100

SITE TABLE

Starting Byte (Octal)	Size (Decimal # Bytes)	Contents
60	2	VBN of the first block in a chain of free blocks in this file (BASES.NAD;nnn). A "free" block is one which has been allocated, used, and subsequently released for re-use.
62	2	VBN of the last block allocated for the element description file (ELMNTS.NAD;nnn).
64	2	Free block chain for the element description file.
66	2	VBN of the last block allocated for the value information file (VALUES.NAD;nnn).
70	2	Free block chain for the value information file.
72	2	VBN of the last block allocated for the file of prose descriptions (TEXT.NAD;nnn).
74	2	Free block chain for the prose description file.
76	2	VBN of first block of this location's table of synonyms for element names; the table is a chain of blocks in the element file (ELMNTS.NAD;nnn).
100	4	Next available address for a synonym entry (VBN and offset).

(2) DBTAB Table

The DBTAB table has a 26-byte entry available for a maximum of 10 data bases residing at the local node. When a DBI from the SITE table is assigned to a data base, a DBTAB entry is created. Each DBTAB entry, when is use, has the following structure:

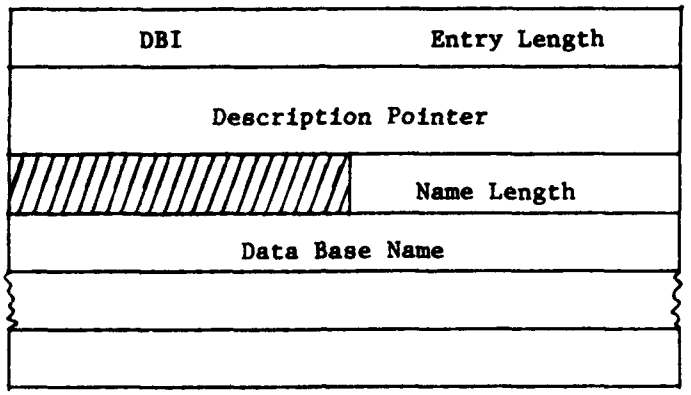


DBTAB TABLE ENTRY

Starting Byte (Octal)	Size (Decimal # Bytes)	Contents
0	1	DBI
2	2	The file table for each data base is a chain of blocks in the file description portion of the BASES file. This word contains the VBN of the first block in the chain.
4	4	Next available address for a file entry (VBN and offset).
10	2	VBN of first block of this data base's element table, which is a chain of blocks in the element file (ELMNTS.NAD;nnn).
12	4	Next available address for an element entry (VBN and offset).
16	2	VBN of the first block of this data base's value table, which is a chain of blocks in the value file (VALUES.NAD;nnn).
20	4	Next available address for a value entry (VBN and offset).
24	2	VBN of the first block of this data base's text table, which is a chain of blocks in the TEXT file (TEXT.NAD;nnn).
26	4	Next available address for a text entry (VBN and offset).

b. Virtual Block Number 2

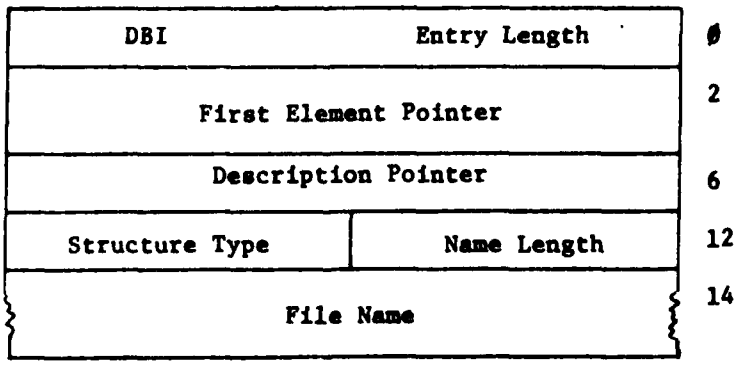
Virtual block number 2 of the BASES FILE contains the data base name table (DBNAM). The first word of the block specifies the number of bytes currently in use, and the remainder of the block is used for name entries, which have the following structure:



Starting Byte (Octal)	Size (Decimal # Bytes)	Contents
∅	1	Length of entry (# bytes)
1	1	DBI
2	4	Address of this data base's prose description in the TEXT file (VBN and offset).
6	1	Length of data base name (# characters).
10	As designated by name length	ASCII name of data base (if name has odd # characters, last character is followed by a zero-fill byte.)

c. Virtual Blocks Number 3 Through N

The remainder of the BASES file is used for file description entries, which have the following structure:



Starting Byte (Octal)	Size (Decimal # Bytes)	Contents
∅	1	Length of entry (# bytes)
1	1	DBI of parent data base
2	4	Address of first element entry in a chain of elements belonging to this file. Element description entries are in file ELMNTS.NAD;nnn, and address is carried as VBN and offset.
6	4	Address of this file's prose description in the TEXT file (VBN and offset).
12	1	Length of file name (# characters).
13	1	Code value designating file structure type.
14	As designated by name length	ASCII name of file (if name has odd # characters, last character is followed by a zero-fill byte.)

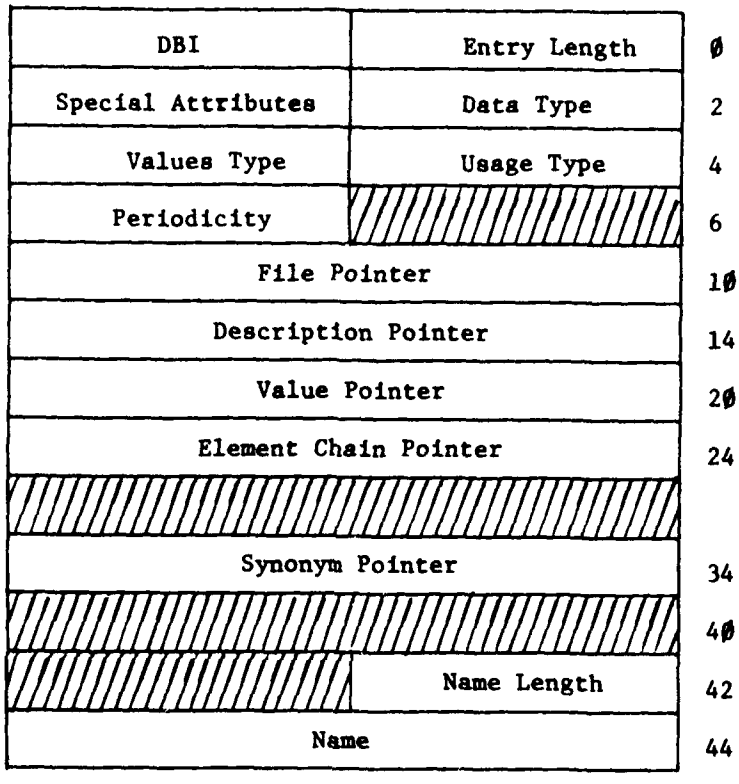
3. *THE ELMNTS FILE*

The ELMNTS file contains information about network data elements. Each data base residing at the local node, whether on the minicomputer or on a host system, has its own "table" of element descriptions; each individual table consists of a chain of blocks in the ELMNTS file. In addition, the ELMNTS file contains a chain of blocks for entries describing synonyms, which are names external to the native DBMS. There are two varieties of synonyms:

- Network standard names, which are used in QNF (Query Normal Form) queries sent to remote nodes for processing;
- Local standard names, which are local designations assigned to any network standard names that do not have corresponding real data base elements locally.

Element description entries are variable in length and are not sorted within blocks or chains of blocks. However, each description entry has a corresponding fixed-length entry in the element name table (file NAMES.NAD;nnn), which is maintained in sort order and is used to locate an element or synonym by name.

Each entry in the element table has the following format:



Starting Byte (Octal)	Size (Decimal # Bytes)	Contents
Ø	1	Specifies the total number of bytes occupied by the entry.
1	1	Data Base Identifier; a code number used to link the element with information about its parent data base; each network data base is assigned a unique DBI.
2	1	Designates the form in which element occurrences are represented. Additional data type classifications may be added. = 1, numeric integer = 2, numeric real

Starting Byte (Octal)	Size (Decimal # Bytes)	Content
		= 3, numeric scientific notation = = 4, alphabetic string = 5, alphanumeric string = 6, date = 7, coordinates = 8, Boolean
3	1	Contains up to 8 bit-flags indicating presence or absence of significant characteristics. Bit Ø , set to indicate inversion Bit 7 , set to indicate this entry is for a synonym rather than an actual element.
4	1	Refers to the way an element name may be used in a query = Ø, name may be used for both qualification and retrieval = 1, name may be used for retrieval only = 2, name may be used for qualification only.
5	1	Describes the value table entry for this data element. = Ø, no value table entry = 1, value table entry contains a list of legal values for the item = 2, value table has high and low values delimiting the range of possible values for the element.

Starting Byte (Octal)	Size (Decimal # Bytes)	Content
		= 3, value table contains a string of format characters defining the acceptable format for element values
		= 4, value table entry contains minimum and maximum number of characters in a legal value for a string element.
7	1	Refers to the way in which an element may occur within a logical file record.
		= Ø, simple element (limited to one occurrence per record)
		= 1, periodic element (may have multiple occurrences per record)
		= 2, relational periodic element belongs to a group of related elements which may, as a group, have multiple occurrences per record)
10	4	Pointer to the file table entry for the file to which the element belongs. Consists of a 2-byte block number and a 2-byte offset.
14	4	Pointer to the prose description of the data element. Consists of a 2-byte block number and a 2-byte offset.
20	4	Pointer to the value information for the data element. Value type indicates the nature of that value information. Consists of a 2-byte block number a 2-byte offset.
24	4	Pointer to the next element entry in a chain of elements belonging to the same file.
42	1	Specifies the number of characters in the element name.

Starting Byte (Octal)	Size (Decimal # Bytes)	Content
44	n	ASCII representation of the element name (zero-fill in the last byte if the name has an odd number of characters).

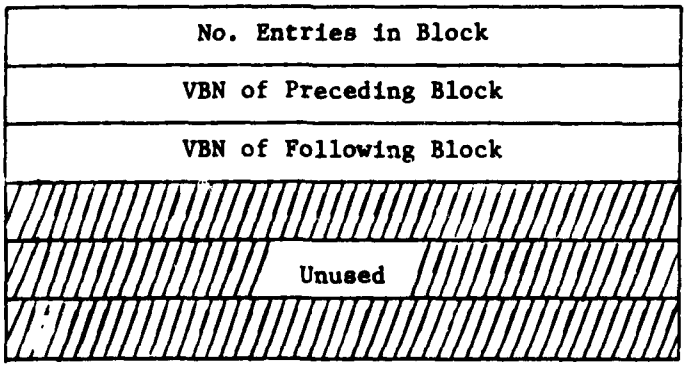
The unused spaces in the element entry are reserved for design enhancements. One of the single-byte open areas may be required for a security code; the 2-byte area following the synonym pointer may be required for additional identification of the synonym (e.g., DBI or location code); and the 4-byte area may hold another pointer (e.g., to link with a record-structure description).

Each entry is stored in a variable length record including a 36-byte fixed length portion and the variable length element name. If we assume that the average name length is 16 characters, an average element table entry will consist of 52 bytes.

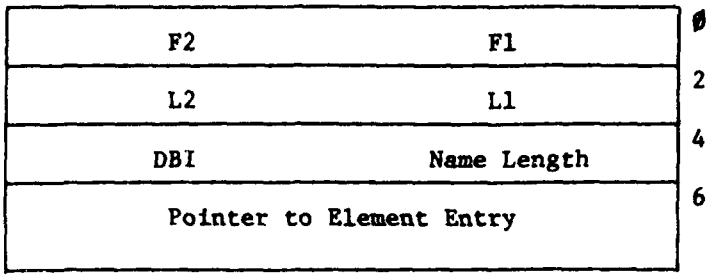
4. THE NAMES FILE

The NAMES file contains a list of fixed-length, shortened versions of element names and synonyms to aid in locating an element description when only the name or a synonym is known. The first 12 bytes of each block is reserved for a block header; the remainder of the block contains up to 50 10-byte name entries.

The block header has the following format:



Each name entry is structured as follow:



Starting Byte (Octal)	Size (Decimal # Bytes)	Contents
Ø	1	First character of element name.
1	1	Second character of element name. (Set to zero if length < 2.)
2	1	Next-to-last character of element name. (Set to zero if length < 4.)
3	1	Last character of element name. (Set to zero if length < 3.)
4	1	Number of characters in element name.
5	1	DBI of parent data base.
6	4	Address of element entry in ELMNTS file.

5. THE VALUES FILE

The VALUES file contains information which may be used in checking the validity of values used in queries against items having either finite value sets, special value formats, or bounded value sets. There are four types of value information entries. Each value entry has the entry length in the first word, but the remainder of the entry differs from type to type. An entry containing a list of values has each value as a character string, with the character strings separated by bytes containing an octal 377. An entry containing a range of values has two character strings divided by a separator byte. An entry containing a picture string contains a single character string. An entry containing a range of string lengths has two numeric values, each contained in a full word.

6. THE TEXT FILE

The TEXT file contains prose descriptions of defined data entities. Each text entry contains a word designating the number of characters of text in the entry, followed by the text itself. Each text entry starts on an even-numbered byte; if the number of text characters is odd, the entry is followed by a zero byte.

SECTION IV

THE DATA BASE DESCRIPTION PROCESS

1. INTRODUCTION

The purpose of this section is to describe how a local data base administrator (DBA) uses the Data Description Processor to define a data base for network access. The data description process is interactive, with the initiative being taken by the system. When the DBA calls in the Data Description Processor, a menu of available data description transactions is flashed to his terminal screen, and he is requested to select the transaction he wishes to perform. Upon receiving the transaction selection, the Processor guides the DBA in entering the information which he must provide to complete the transaction. Information is entered to blank fields in description forms associated with each type of transaction. Cursor movement from field to field is controlled by the system. Each time a transaction is completed, the system returns with the transaction selection menu, and the description session continues until the DBA indicates that he has no more transactions to perform.

The only rules associated with describing a data base are those governing the order in which entities may be defined:

- A data base must be defined before any of its component files
- A file must be defined before any of its component elements.

The DBA is under no compulsion to allow network access to all of the files in a data base or to all of the elements in a file; he merely refrains from defining those files or elements which are intended for local use only.

In the full system implementation, Data Base Description will be a password function to prohibit its use by any person other than the autorized data base administrator.

2. TRANSACTION SELECTION MENU

The transaction selection menu is flashed to the terminal screen when the Data Description Processor is initiated and after each transaction is completed. The following illustration of the selection menu includes those transactions available in the initial implementation. Additional transactions may be added.

```
SELECT TRANSACTION TYPE:

     1.  DEFINE DATA BASE          2.  DEFINE FILE
     3.  DEFINE ELEMENT            4.  CHANGE ELEMENT DESCRIPTION
     5.  DELETE DATA BASE          6.  DELETE FILE
     7.  DELETE ELEMENT            8.  EQUATE ELEMENT NAMES
     9.  ENTER LOCAL NAME FOR     10.  NO TRANSACTION
           ELEMENT
     ENTER CHOICE _____
```

The available transactions have the following functions:

1: Describe the characteristics of a data base residing at the local node, either on the minicomputer or on a host system.

2: Describe the characteristics of a file belonging to a local data base.

3: Describe the characteristics of an element belonging to a local file.

4: Change the description of a local element.

5: Delete the descriptions of a local data base and its component files and elements.

6: Delete the descriptions of a local file and its component elements.

7: Delete the description of a local element.

8: Equate a local element name with its equivalent network standard name.

9: Provide a local synonym for a network standard name which has no equivalent element in the local data base.

10: Terminate the current description session.

3. DESCRIPTION FORMS

In order to perform the processing required for each transaction, the Data Description Processor must elicit from the DBA the information that is pertinent to the transaction. This is done by flashing to the terminal screen forms with blank fields to be filled in by the DBA. This section describes the form(s) used for each transaction.

a. DEFINE DATA BASE Transaction

If the DEFINE DATA BASE transaction is selected, the following form is flashed to the DBA's terminal screen:

```
DATA BASE NAME:_____________________________________________
NATIVE QUERY LANGUAGE:
     1. DIAOLS                    2. TIMS
     3. OTHER
     ENTER CHOICE_____________
DESCRIPTION: _______________________________________________
_____________________________________________________________
_____________________________________________________________
_____________________________________________________________
_____________________________________________________________
```

The native query language selection is the system's way of determining which local data base management system controls the data base being defined. The list of query languages is site-dependent, the criteria for "availability" of a query language at a given site being that there is a local data base management system utilizing that query language and that a TIIN module to translate from the Query Normal Form to the native query language has been installed at the site.

b. DEFINE FILE Transaction

If the DEFINE FILE transaction is selected, the following form is flashed to the DBA's terminal screen:

```
FILE NAME:__________________________________________________
PARENT DATA BASE NAME:______________________________________
DATA STRUCTURE TYPE:
     1. SEQUENTIAL                2. HIERARCHICAL
     3. NETWORK                   4. RELATIONAL
     ENTER CHOICE_____________
DESCRIPTION:________________________________________________
_____________________________________________________________
_____________________________________________________________
_____________________________________________________________
```

c. DEFINE ELEMENT Transaction

If the DEFINE ELEMENT transaction is selected, the following form is flashed to the DBA's terminal screen:

```
ELEMENT NAME:_____
PARENT FILE NAME:_____
PARENT DATA BASE NAME:_____
ELEMENT USAGE:        __RETRIEVAL        __QUALIFIER
DATA TYPE:
     1.  NUMERIC INTEGER                  2.  NUMERIC REAL
     3.  NUMERIC SCIENTIFIC NOTATION      4.  ALPHABETIC
     5.  ALPHANUMERIC                     6.  DATE
     7.  COORDINATES                      8.  BOOLEAN
     9.  OTHER
     ENTER CHOICE _____
SPECIAL ATTRIBUTES:      __PERIODIC
       ___RELATIONAL PERIODIC          ___INVERTED
VALUE INFORMATION TO BE PROVIDED:
     1.  LIST OF VALUES                   2.  RANGE OF VALUES
     3.  PICTURE                          4.  STRING LENGTH RANGE
     5.  NO VALUE INFORMATION
     ENTER CHOICE _____

DESCRIPTION:_____
_____
```

If the DBA indicates he has value information to provide (any choice other than 5. on the second selection menu), an appropriate value form will be flashed to the terminal screen.

(1) Type 1 Value Form (List of Values)

The following form allows the DBA to enter a list of the values which may be assumed by the element he is describing:

```
ENTER VALUE LIST (USE COMMA AS VALUE SEPARATOR; ENCLOSE
STRING VALUES IN QUOTES):
_____
_____
_____
_____
_____
_____
_____
_____
_____
```

(2) Type 2 Value Form (Range of Values)

The following form allows the DBA to specify the range of the values which may be assumed by a numeric element by designating the smallest and largest possible values:

```
ENTER MINIMUM LEGAL VALUE:_____
ENTER MAXIMUM LEGAL VALUE:_____
```

(3) Type 3 Value Form (Picture String)

If the legal values of a character-type element have a specialized format, the DBA may specify the format using the following form:

```
ENTER PICTURE STRING:_____

EACH CHARACTER IN THE PICTURE STRING DESCRIBES THE
CORRESPONDING CHARACTER IN A LEGAL VALUE FOR THE DATA
ELEMENT.  PICTURE CHARACTERS MAY BE FOLLOWED BY A
REPETITION FACTOR, WHICH IS AN UNSIGNED DECIMAL INTEGER
CONSTANT, N, ENCLOSED IN PARENTHESES, TO INDICATE
REPETITION OF THE CHARACTER N TIMES.  THE PICTURE
CHARACTERS THAT MAY BE USED ARE AS FOLLOWS:

A  SPECIFIES THAT THE ASSOCIATED POSITION MAY CONTAIN ANY
ALPHABETIC CHARACTER OR A BLANK CHARACTER.

X SPECIFIES THAT THE ASSOCIATED POSITION MAY CONTAIN ANY
CHARACTER OF THE DATA SET.

9 SPECIFIES THAT THE ASSOCIATED POSITION MAY CONTAIN ANY
DECIMAL DIGIT.

THE APPEARANCE IN A PICTURE STRING OF ANY CHARACTER OTHER
THAN A, X, OR 9 SPECIFIES THAT THE ASSOCIATED POSITION
MUST CONTAIN THAT CHARACTER.
```

(4) Type 4 Value Form (String Length Range)

The following form allows the DBA to specify the minimum and maximum number of characters allowed in a legal value for a character-type element:

```
ENTER MINIMUM NUMBER OF CHARACTERS:_____
ENTER MAXIMUM NUMBER OF CHARACTERS:_____
```

d. CHANGE ELEMENT DESCRIPTION Transaction

Two forms are used for the CHANGE ELEMENT DESCRIPTION transaction. First, the DBA must identify the element description to be changed by filling in the following form:

```
ELEMENT NAME:_____
PARENT FILE NAME:_____
PARENT DATA BASE NAME:_____
```

The Definition Processor locates the element description and moves the current descriptive information into the blank fields of an element description form (see DEFINE ELEMENT transaction). Fields which the DBA is allowed to change are indicated on the form, and the filled-in form is flashed to the terminal screen. If a change in the value information is indicated, the appropriate value form is also used.

e. DELETE DATA BASE Transaction

If the DELETE DATA BASE transaction is selected, the following form is used for identification of the data base:

```
DATA BASE NAME:_____
```

f. DELETE FILE Transaction

If the DELETE FILE transaction is selected, the following form is used for identification of the file:

```
FILE NAME:_____
PARENT DATA BASE NAME:_____
```

g. DELETE ELEMENT Transaction

If the DELETE ELEMENT transaction is selected, the following form is used for identification of the element:

```
ELEMENT NAME:_____
PARENT FILE NAME:_____
PARENT DATA BASE NAME:_____
```

h. EQUATE ELEMENT NAMES Transaction

If the EQUATE NAMES transaction is selected, the following form is flashed to the DBA's terminal screen:

```
ELEMENT NAME:_____
PARENT FILE NAME _____
PARENT DATA BASE NAME:_____
NETWORK STANDARD NAME:_____
```

i. ENTER LOCAL NAME FOR ELEMENT Transaction

If the ENTER LOCAL NAME FOR ELEMENT transaction is selected, the following form is flashed to the DBA's terminal screen:

NETWORK STANDARD NAME: _____

LOCAL SYNONYM: _____

SECTION V

OUTSTANDING DEVELOPMENT PROBLEMS

1. INTRODUCTION

Some areas which have not been resolved in the current level of the system specifications are mentioned in this section. They are suitable subjects for consideration in any further TIIN development.

2. VALUE TRANSLATION

Value translation poses a problem for further TIIN/TILF design. The general lack of correspondence between values in comparable files of data bases with differing applications presents a challenge to a network integrating effort. Multiple files which are identical with respect to structure, data element names, and coding conventions for values present no problem in value translation.

Multiple similar files ("similar files" indicates files which have differing data element names and coding conventions, but with a 1-1 correspondence for most elements, so that translation is possible) will be handled, at first, in such a way that files containing values irreconcilable with those in data requests will result in nonperformance of a query against that particular file. The overlapping of values, the problem of interpreting value codes, the lack of precise definitions for many values, and the sheer volume of possible values for some elements present technical challenges yet to be met.

3. MULTIPLE QUERIES

Another problem is that of multiple queries. The term "multiple queries" is general and includes a number of ways for sending queries, all of which constitute outstanding technical issues to be resolved.

Yet to be addressed is the sending of two or more simultaneous parellel queries to separate files. These queries would represent identical data requests, but would be routed to that number of data bases which contain relevant files.

Another capability to be considered would be that of splitting data requests and sending portions of queries to relevant files. Responses would be merged later in the TIIN configuration.

There should be consideration given to providing a cross-file reference capability. By "cross-file reference" we mean a situation where a sequence of queries is performed; each query generates a hit file which contains keys for use in the next query which would generally be on a different file.

The technology required to provide a cross-file reference capability for TIIN has not been addressed in detail; however, it should be completely transparent to the user, except for the additional elapsed time required for sequential retrievals. At present, no DMSs allow such a capability.

4. ERROR HANDLING

The question of how errors should be presented to the user becomes a significant one as the range of system capabilities expands. With transparency at the query language and data base levels, more than one option for handling error messages becomes apparent. There is the possibility of devising a common format for error messages which would be understood systemwide.

The data management systems of a given extended network will have varying error-message formats. If standardization of DMS and operating system messages can be achieved, the standardization of error messages would seem to be a natural corollary.

A format for error reports could take the form of the QNF. One possibility would be a format which provides for an error-message number augmented with arguments naming those data element names in question. The provision of a filtering function for system-errors is important. The user need not be concerned with the former.

5. AUTHORITY OF DBA

Yet to be resolved, as well, is the scope of the data base administration role in the network. Data base administration involves the mediation of conflicting needs and applications at both the node and system levels. The local DBA is responsible for determining what local data are to be made available to network users and providing sufficient information about local data to make them accessible to the network. However, there appears to be a need also for administrative authority above the node level, in other words, a network data base administrator. This role would involve, at the minimum, assigning network standard names and network standard value representations for synonymous data elements in different data bases. In addition, the network DBA should be the ultimate authority on all questions regarding synonymity between network data elements; for example, whether or not one data element is a legitimate synonym for another, and whether or not there is a viable algorithm for translating the value representation used for one "synonym" to that used for another.

SECTION VI

SYSTEM DESIGN

1. INTRODUCTION

This section discusses some of the basic considerations which went into the system design. Before basic design decisions could be made, research into a number of areas where work has already been done had to be performed. These areas included user-languages, common query languages, data base management systems, security considerations, standardization of data structures, integration of data management systems, and data description languages. A bibliography has been compiled which comprises work done in these areas. (It is not included here.)

This portion of the report also includes a discussion of the network configuration, with illustrated diagrams. This discussion puts TILF/TIIN in a context with other TOSS modules.

The system architecture is described together with charts.

Finally, interactions between the system and the users, and between the system and human support are discussed.

2. PHILOSOPHICAL AND PRACTICAL CONSIDERATIONS AFFECTING THE SYSTEM DESIGN

a. User Language Trade-offs

Consideration has been given to previous research in the area of user-oriented languages. The DRIF and the QNF will be transparent to the user. As languages, they are incomprehensible to the analyst. They would appear to him merely as a sequence of numbers. The existence of an intermediate language does not obviate the need for fairly precise knowledge on the part of the user of data bases being accessed. If the intermediate common language is to be functionally expressive, then relative user-sophistication is required in expressing data requests.

It is, however, desirable to enhance the ease with which increasingly complex data sources may be accessed. This suggests the utility of splitting the data request into two parts--a user-interface characterized by ease of use, and a powerful intermediate, common language.

Ideally, an analyst/user would express his data request in simple English. Research done previous to this effort indicates the limited utility of an English-like front-end. However, the idea of using natural languages suggests certain system needs which

would obtain even if a somewhat more network-oriented user language were to be employed. One need is that of representing descriptive information about network data bases in store. The descriptions would reflect the user's conceptualization of the distributed data bases and would include syntactic information on the files. The system needs this information in order to process a query which otherwise would be incomprehensible. This data base description information can be distributed among nodes in an integrated network, or it can be stored centrally for common access by all nodes. The TILF development, of course, employs the former configuration.

Regarding the development of a common language facility, it is important to devise a language which optimizes simple, commonly-used applications without ruling out the possibilities for more sophisticated usage in a wider range of applications. This implies that a so-called "natural" language is not the best candidate for a common language; in the current state of the art, natural languages have been developed only within rather narrow usage contexts. However, freedom from rigid syntactical rules in a wide range of applications can be provided by the use of selection menus and terminal display forms and enhanced by the addition of a user command mode for more complex applications.

b. Normalization of Responses

Since data which are retrieved from a data management system in response to a single query may be from separate files, they are likely to exhibit different characterisitcs with regard to coding conventions, field width, and numeric representation, etc. There is an obvious need for a facility which converts responses to standard format, enabling collation, and reconciliation of coding and/or substantive conflicts and duplications. Response normalization also facilitates storage of data in the analyst's private file where it would be available for further processing (such as refining the original request, sorting, statistical analysis, graphic display, etc.).

The purpose for which response normalization is more useful is the generation of standard, integrated reports comprising responses from different nodes. Transmission of complex sets of data from one node to another necessitates a common response format analogous to the QNF. The distribution of the response normalization process represents a problem yet to be resolved.

c. Meaningful Refusal of Data Requests

It has been determined that a refusal of a data request at the target level must be accompanied by a message meaningful to the user indicating the nature of the refusal and the precise inadequacies of the data request. It is believed that a meaningful refusal message would reduce future processing in that it would instruct the user in the proper rephrasing of his query.

Another area of concern was that of error messages and their standardization. This issue was addressed in detail in Section V.

d. Communications and the Control of Message Routing

An integrated network requires the existence of a routing module which coordinates messages according to information contained in data requests and responses indicating the source and the target of the message. Network switching modules do not suffice where integration of responses from external data bases is to be performed.

e. Location of the NAD

The location of the Network Access Directory(ies) is an important consideration with implications for network communications load, processing time, data description integrity, and security. Whether a centralized or distributed NAD is desirable was an issue which had to be resolved at the most basic network design level. As should be obvious from the diagrams and previous discussions, the distributed approach for network data directory was selected. Distribution of NADs and the attendant responsibilities of the data base administrator germane to the directory are implicit in the present TIIN configuration.

3. SYSTEM ARCHITECTURE

The system architecture is illustrated in diagram B-1 and in the diagrams in this section. The network configuration vis-a-vis other TOSS components was discussed in the second subsection. This discussion is concerned with the integrated functioning of the TIIN system.

The TIIN user initiates a data request in one of several ways. This choice is allowed, ultimately, by the Terminal Access Procedures Module (Figure VI-1). It includes perusal of a subject catalog; sending a stored query; expressing a request in a formal syntax command mode; engaging in interest negotiation through a dialogue mode; and selection of a response display mode. The NAD is employed in perusing the catalog. (The TAP is projected at this stage of development.)

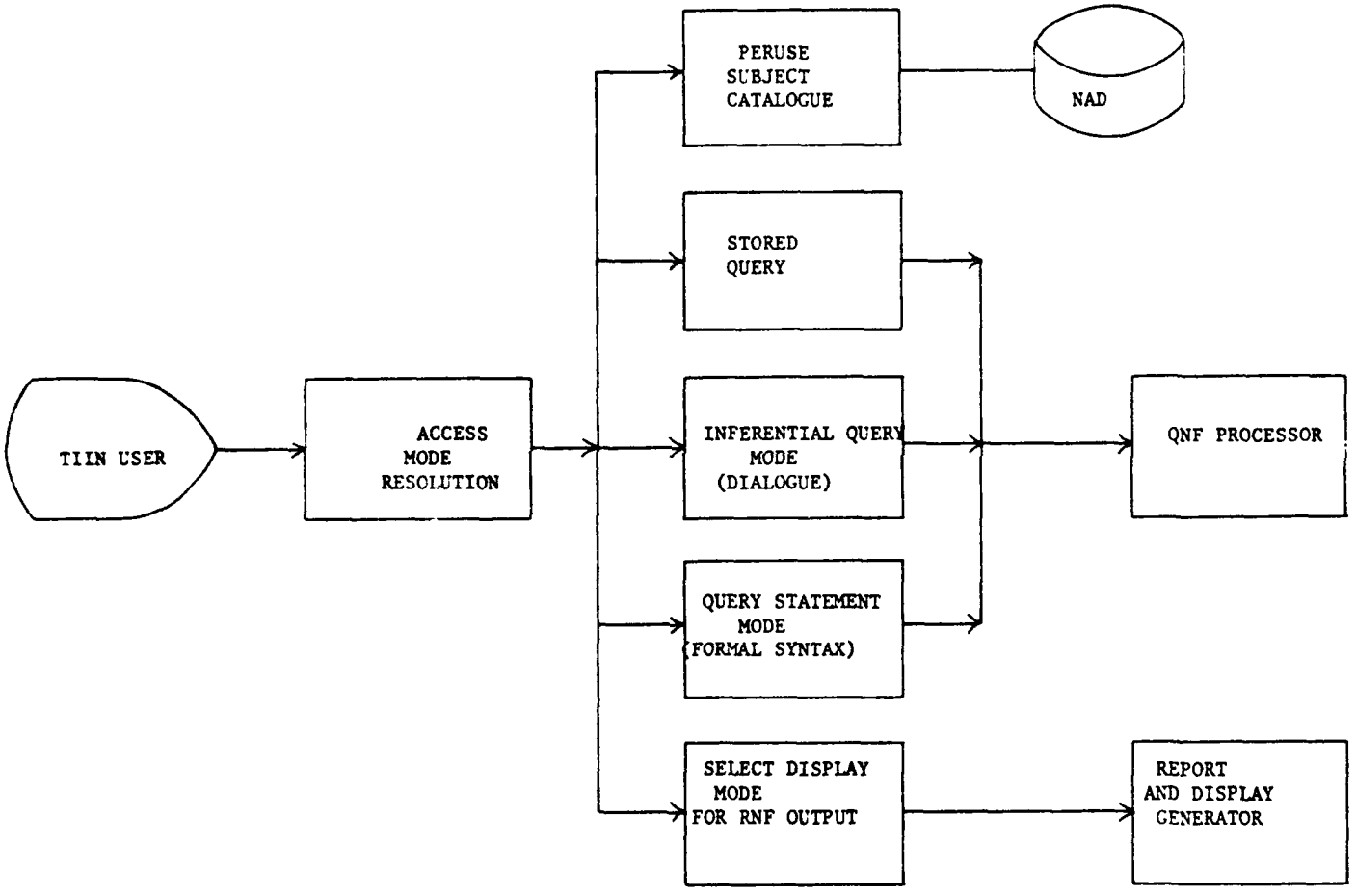


Figure VI-1. Terminal Access Procedures

The query moves along in the DRIF format to the QNF Generator (expressed in Figure VI-2 as "Data Location and Query Translation") where the basic DRIF is altered and augmented to include not simply the information provided by the user in his request, but the correct data element name synonyms and data base address flags. This is supplied by the QNF Generator through interface with the NAD, which contains relevant data element name synonyms and data location information. The result of this "massaging" is the QNF (Query Normal Form), which represents the data request in the form in which it will be shipped to the appropriate data base(s). The QNF proceeds to the query distribution executive (QDE), the function of which is to control the traffic of data requests and responses, assigning tags and logging assigned numbers for accurate tracking. Once the request has been tagged for tracking, it is either sent to the local data management system-oriented query processor (QL-processor) or sent to the network communications interface to be sent out to another node.

When the data request, in the form of the QNF, is received at the QL-Processor, it is translated into the proper query language of the data management system for which the QL-Processor provides a TIIN interface. If translation is inappropriate due to the non-existence of requested information, an error message is sent back through the network, via the QDE, informing the user of the error and/or reasons for refusal.

At this point in the TIIN flow, the final design remains open as far as where response normalization would be most efficient. Two scenarios are envisioned. They are:

- When the data request has been satisfied at the DMS/DB level, the response is "normalized" (put into a standard system format) for shipment back to either the local QDE, or an external, source QDE via the Network Communications interface and other communications software (Terminal Independent Support System and Toss Exchange Center) where it is tagged as a response received. The response is then sent to the user in the form he has requested where it is displayed on his terminal.

- When the data request has been satisfied at the DMS/DB level, the response is sent in "raw" form to the Query Distribution Executive (QDE) where it is tagged as a response received and either sent out to an external, source node via the Network Communications Interface (and other TOSS software), or to the local Response Normalizer, where it is processed for rendering in the system standard format (RNF). After the response has been thus processed, it is sent to the user where it will be displayed at his terminal in the form he has specified.

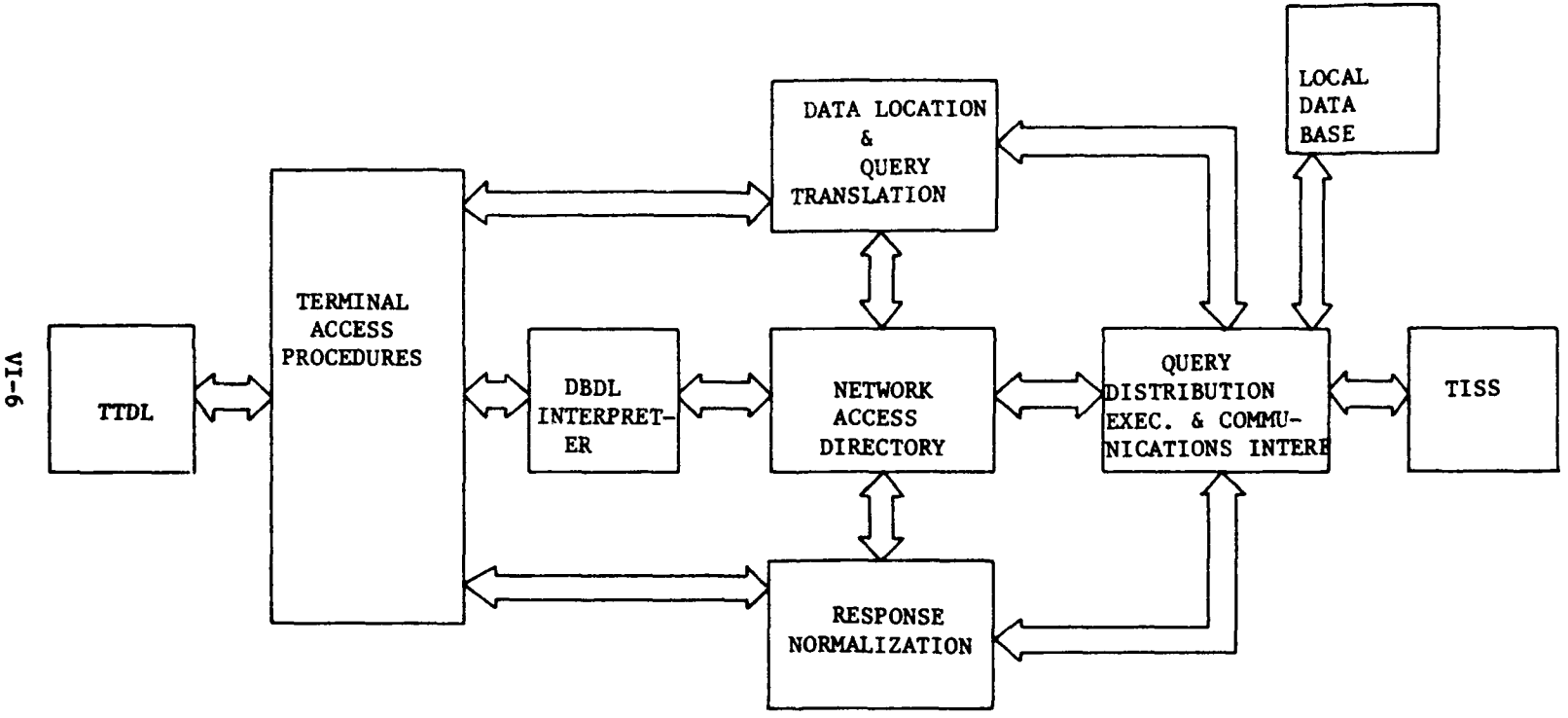


Figure VI-2. Transparent Integrated Intelligence Network

This, in simplistic form, is the flow of the TIIN and is in way of a description of the system architecture.

4. NETWORKING CONFIGURATION

Figure VI-3 illustrates the relationship of the various TOSS software components and their collective function as a "front-end" to the host systems providing the actual data bases. Figure VI-4 (Intelligence Supplier and Consumer Nodes) illustrates the relationship of the minicomputer and its constituent TIIN and communications software to the hosts. A "consumer node" is one which lacks data base facilities and, therefore, must "consume" the data residing in host ("supplies") systems. A node which makes no consumer demands on network data resources (i.e., lacks user terminals), but which contains a data base, functions in a purely "supplier" capacity. These concepts, borrowed from elementary economics, are in way of clarification of networking configurations and are not intended to reflect real world conditions.

Figure VI-5 (TIIN Relationship to other TOSS Components) illustrates the relationship of the various elements of TOSS in particular TISS and TEC communications modules, to TIIN. They reside in minicomputer network nodes and act in concert as front-ends to host systems where data bases reside. TTDL is that software particularly related to the TIIN/user interface (TAP). Terminal Independent Support System (TISS) handles on-line network communications, facilitating the development of on-line application programs. TOSS Exchange Center (TEC) provides network communications control for bulk data and conversational message traffic.

5. INTERACTION BETWEEN THE SYSTEM, USER, AND HUMAN SUPPORT

An integrated network presupposes the existence of data base administrators at each node who are familiar with the responsibilities placed on them by system requirements (i.e., the need to make the data and schema responsive to a multiplicity of user programs throughout the intelligence network). Moreover, the need for a network administrator with system-wide coordination responsibilities becomes apparent with network integration.

The DBA at the local node is responsible for building the NAD and for maintaining it. Figure VI-6 illustrates this human support function.

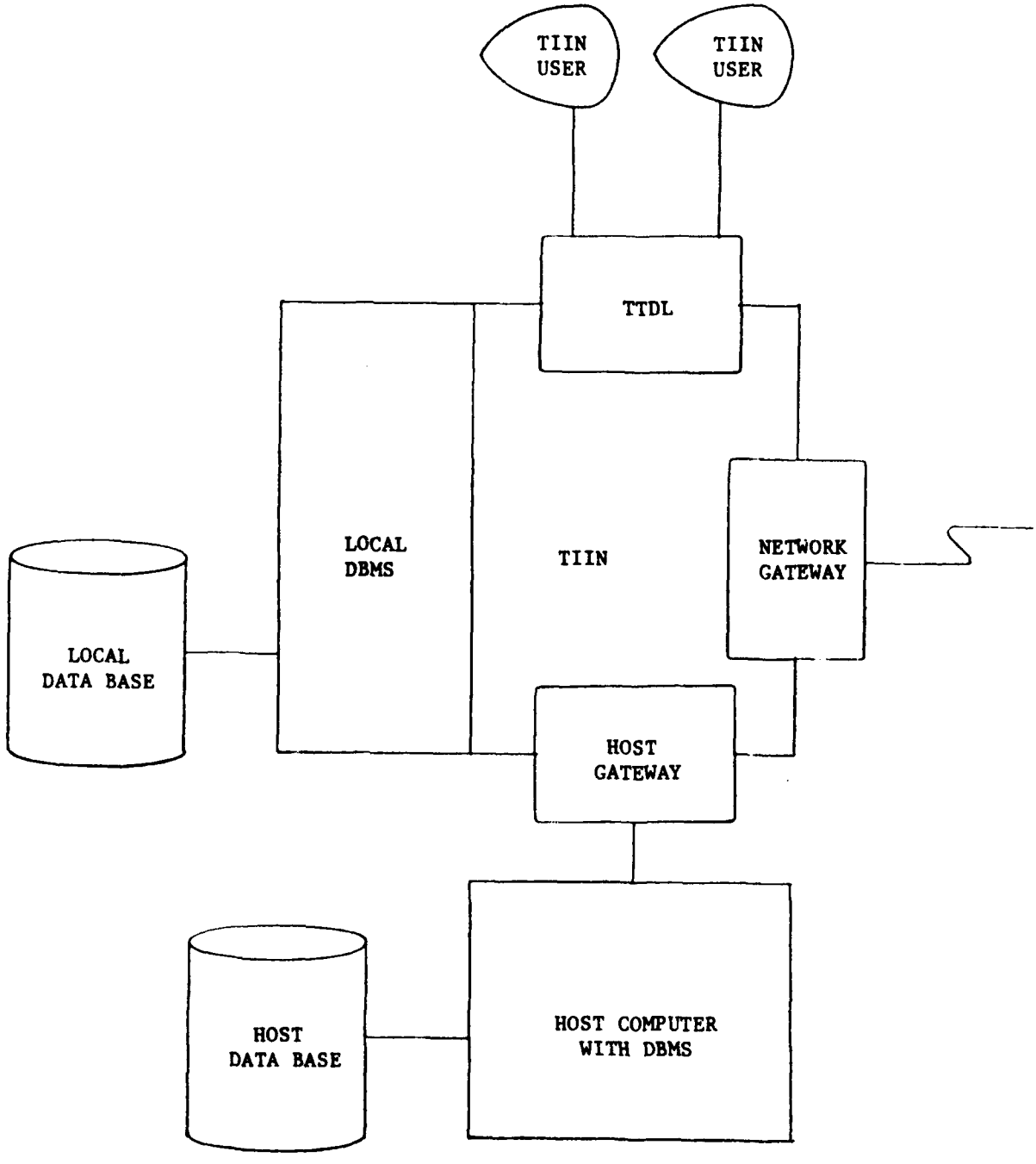


Figure VI-3. TOSS as a Front-end to Host Systems

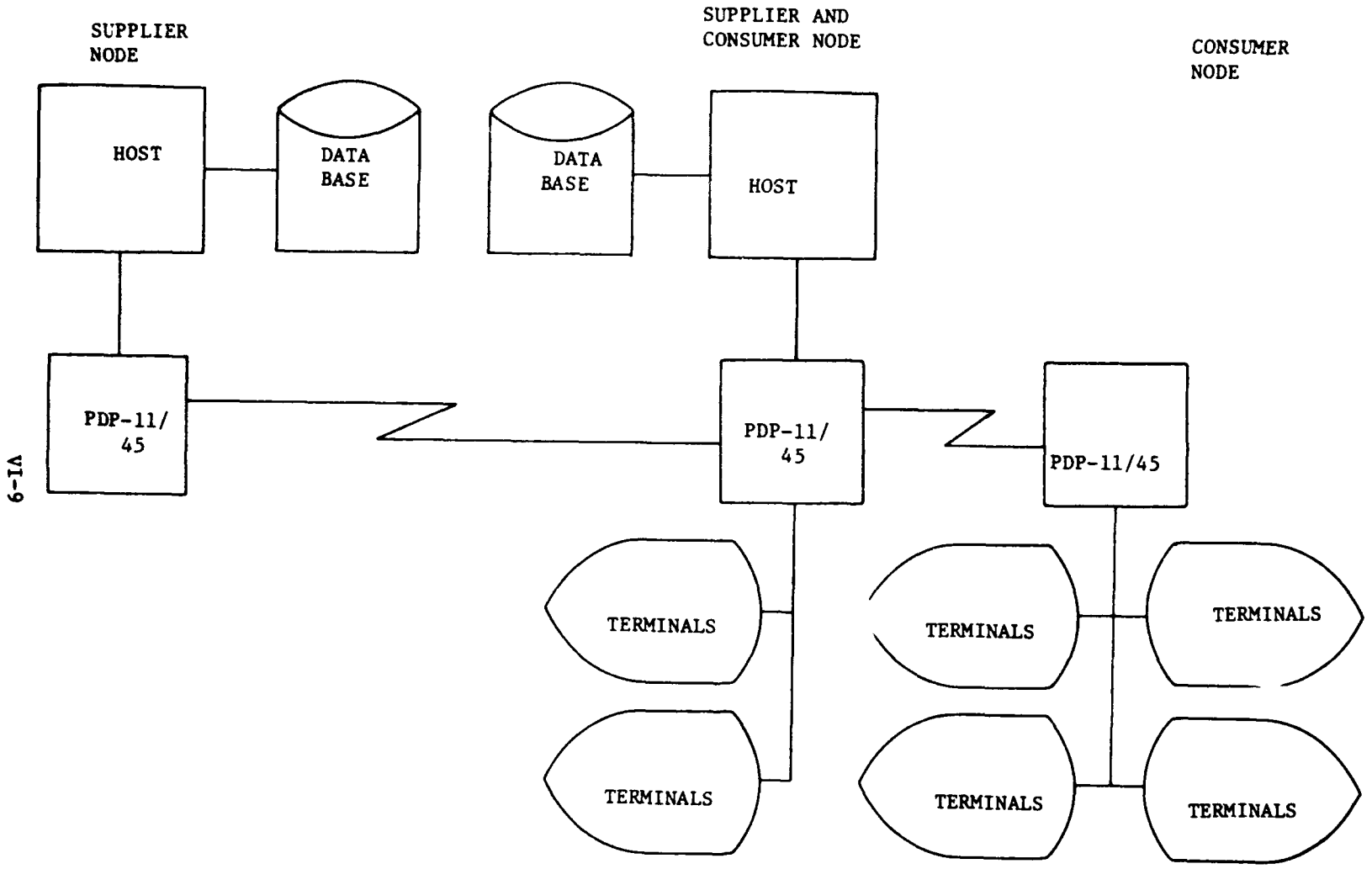


Figure VI-4. Intelligence Supplier and Consumer Nodes

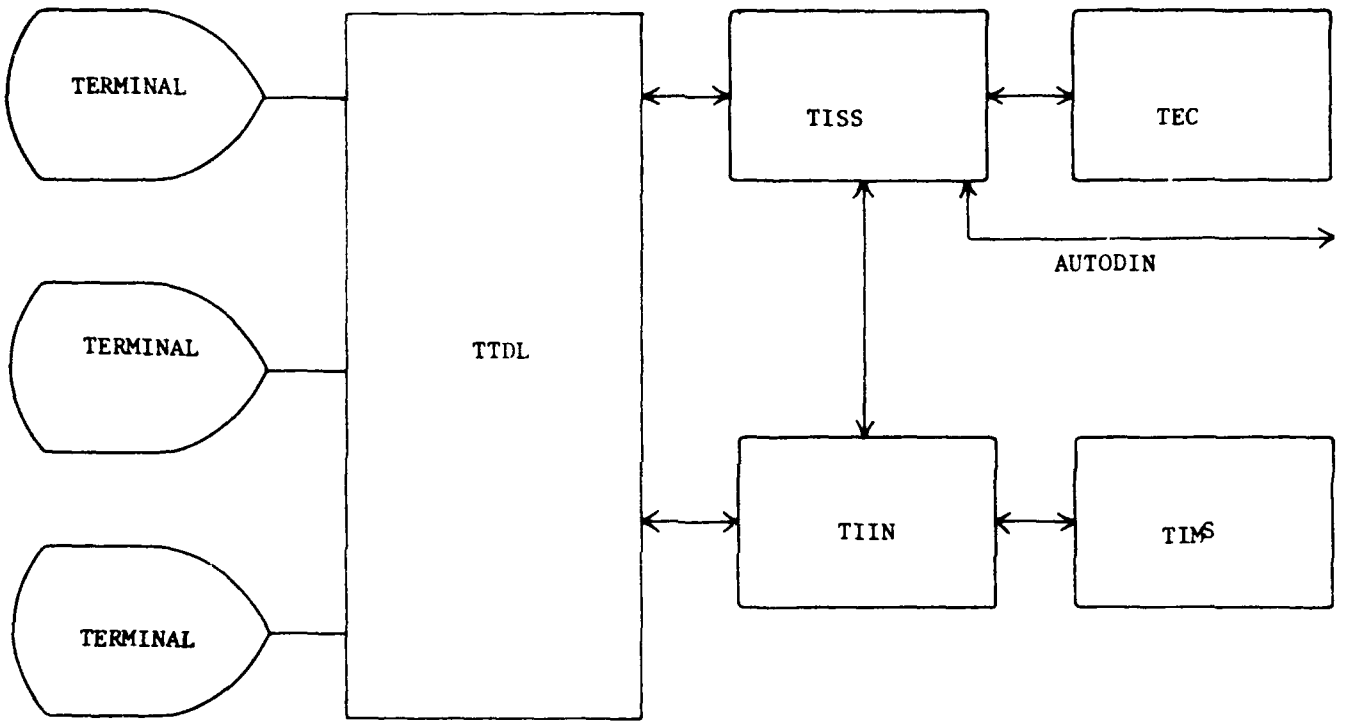


Figure VI-5. TIIN Relationship to other TOSS Components

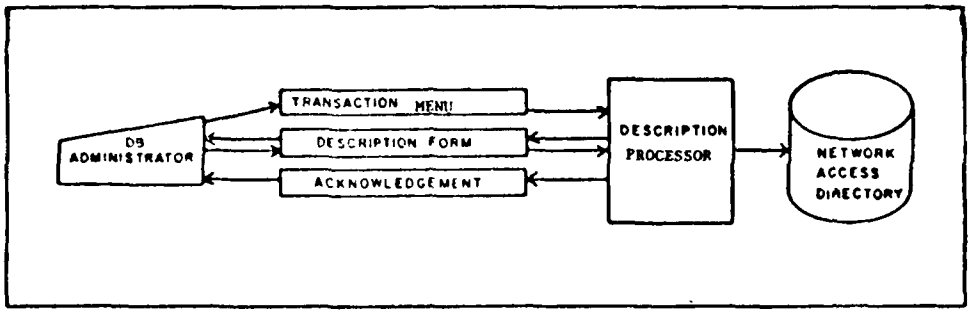


Figure VI-6. Building Network Access Directory

The task of bringing up the NAD entails much initial effort. However, updating, since it is the user's responsibility, should not pose a mojor problem in time expended.

The intelligence analyst is asked to be more than a passive consumer of the TIIN. His needs play an active role in the TIIN design and in the design and maintenance of data base descriptions. The applications for which network data resources are put to use determine the way in which their descriptors are represented in the NAD.

APPENDIX A

BACKGROUND

1. APPROACH

TIIN's development is predicated on the needs of intelligence analysts for automated support for data access and manipulation. Analysts are currently being served by their own skills and increasingly effective data processing capabilities. Development of automated capabilities include TOSS and the NMIC Modernization Plan. Requirements for further automated support arise from the need to make remote and diverse data base resources available to the analyst in a near real-time mode without requiring the analyst to become technically knowledgeable in automated data management systems or to even know the structure and content of the remote sources. TIIN's development adheres to the transparency and distributed processing concepts developed for TOSS.

The approach of TILF personnel thus far has been to establish analyst/user requirements to derive user capabilities for meeting them, to match these posited user capabilities with specific system capabilities, and to identify the technology developments necessary to provide these capabilities. Project personnel have then proceeded to design and code the software indicated by desired capabilities. Five stages representing increasingly powerful levels of user capabilities have been identified in the development plan. This Final Technical Report details those efforts which are in fulfillment of that portion of Stage One relegated to the TILF project. An account of those tasks performed under the related TIIN project (RADC Contract Number F30602-76-C-0090) is forthcoming and will appear in August 1976.

2. NETWORK DEVELOPMENT CONCEPTS

a. General Analyst Requirements

The intelligence analyst is responsible for assessing information related to critical and complex situations which have far ranging political and military significance, both nationally and internationally. The analyst frequently must work under considerable time pressure. To support his analytical and judgmental capabilities, a large volume of diverse information exists in multiple, uncoordinated sources. Analysts have traditionally developed highly personalized and often very sophisticated manual techniques for accessing information, based on experience and knowledge of sources. There is a strong and continuing need for automated support to provide the analyst with ready access to and processing of data. A large measure of automatic data processing support is being developed using facilities already in place. However, these facilities are geographically dispersed and reside on independent hardware and software systems.

A major need is on-line access to and processing support for data bases which are not part of the analyst's immediate system. To be useful to the analyst, ADP support must make minimal demand on him to learn access techniques and processing requirements. The ideal ADP support should unburden the analyst from an already substantial data processing workload. This requires the implementation of interactive, integrated networking concepts with sufficient translating and routing software to permit the analyst to access external systems in his own familiar terms The provision of simplified, real-time access to remote data bases would greatly aid the analyst in cross-correlation, data assessment and reporting functions.

These analyst requirements and the present state-of-the-art of transparency technology have coalesced in the present TIIN/TILF effort.

The TIIN development has employed some important concepts which have gained currency industry-wide. Indeed, the very acronym of the program incorporates at least two of the newer concepts--those of transparency and integration.

b. Guiding Design Concepts

The development of the Transparent Integrated Intelligence Network has proceeded in accord with several major organizing principles. They include transparency, data base integration, and data sharing. Each of these ideas will be dealt with in this subsection.

(1) Transparency

The idea of transparency requires the most explication. It is a concept which has been used widely in ADP related to the Department of Defense. Wide usage, unfortunately, does not necessarily indicate clarity. TILF and TIIN have used the idea of transparency in a decided sense which is probably best communicated in terms of <u>levels</u> of transparency.

The notion of transparency, as applied to an information processing system, refers to those systems characteristics which allow a user to ignore, indeed be unaware of, complexities and diversities which are not relevant to his purpose. For example, problem-oriented languages such as COBOL and FORTRAN make machine language transparent to their users.

With respect to distributed intelligence data base networks, the following levels of transparency represent alternative goals:

(a) No Transparency

Without transparency the analyst must use separate terminals independently connected to each external system. He must use the precise system access procedures as well as that query language associated with the data base he is accessing (the native language of the data base).

(b) Communications Transparency

At this level, the analyst would be able to retrieve data from external systems at his own analyst station. This is, communications and line protocols would be made transparent to the analyst, as well as log-on procedures to the external systems. The analyst would still be required to use the data base access procedures (query languages) of the various external systems in order to retrieve data.

(c) Query Language Transparency

At this level, the analyst is able to retrieve data from diverse data bases and data files using a single query procedure from his own terminal. He does not have to know the query language of the particular external data base. There are, however, demands made on the analyst's time and knowledge in as much as he must be aware of the dispersion, structure and conventions for data bases being accessed.

(d) Data Base Transparency

At this level, the dispersion of data resources among the various files and separate data bases is transparent to the analyst. Data is retrieved using one data access procedure from the analyst's terminal; computer software is used to separate the single query into separate queries to be directed at various data bases and files containing the requested information. In addition, the disparity among similar data element names and coded data field values is resolved.

(2) Integration

The idea of integrated data management systems is concerned with the provision for all users on an extended computer network of a means to access all network data management systems. The concept presupposes the existence of a common language which

is understood system-wide. Briefly, any user in the network can access any remote data base. He will make a data request which will be routed to any data base (remote or not) in the common language. The request is processed at the "target node" by a module which provides an interface between the network and the data management system. This module performs a translating function which accepts a data request from another node and transforms it into DMS-compatible form in accordance with the language of the DMS with which the module is associated. The network/DMS interface module also performs the function of translating responses into the common language. This latter function allows for the ultimate merging of responses from a multiplicity of data bases into a coherent, integrated response.

The net effect of providing a common network language and an interface with remote data management systems is the integration of vast stores of information for use by all members of a community of need. The intelligence community is one such which shares data requirements and, hence, could benefit from implementation of an integrated approach to network data management.

(3) Data Sharing

Data sharing is an idea which has grown out of the existence of a "community of need." The rapidly progressing ADP sciences have spawned a multiplicity of capabilities which, in turn, have stimulated rising expectations. The sheer volume of manipulatable and storable data has suggested the impracticality of duplication and the utility of sharing common data banks. The expense involved in maintaining a completely adequate store of data has risen at a rate which now makes the provision for the sharing of data cost-effective. With the idea of sharing data among various related agencies comes the need for a means for coping with the inevitably varying needs of user agencies. While an organization might use the same or similar data as another organization, the uses to which these data are put militate restructuring according to special applications.

Computer networking has provided the first of several necessary steps in fulfillment of the idea of data sharing. However, because of varying applications, the need for developing techniques for remote access of data remains. It is this need which much of the TIIN/TILF effort is addressing.

One of the most obvious ways to facilitate data sharing is through the use of a common data management language. However, this is but one of several approaches, some of which are less obvious.

3. DEVELOPMENT APPROACH

INCO's approach to the development of a Transparent Integrated Intelligence Network is to build upon and extend the concepts developed for the Terminal Oriented Support System; to develop a system architecture incorporating operational TOSS modules to provide terminal transparency, communications transparency, and distributed processing; and then to proceed with an orderly, staged development of the remaining system modules.

APPENDIX B

FUNCTIONAL DESCRIPTIONS OF TIIN MODULES

1. INTRODUCTION

The purpose of this appendix is to provide a concise description of the function of each module included in the design of the Transparent Integrated Intelligence Network. The modules are in varying phases of development. These descriptions reflect the range of developmental stages and are, therefore, in varying degrees of detail.

The overall network has been divided into subsystems which perform discrete tasks. The modules comprising each subsystem perform tasks which relate to one another in a manner such that it is meaningful to think of them as a unit. The functional descriptions which follow, then, are organized according to subsystems.

Figure B-1 illustrates elements of the TIIN discussed here. It will be referred to throughout this appendix.

2. FUNCTIONAL DESCRIPTIONS

a. Terminal Access Procedures (TAP)

(1) TAP Subsystem

The TAP subsystem is designed to provide a convenient and natural user interface method. The function of TAP is 1) to describe network data resources to a user/analyst, 2) to support the user in the construction of legal queries, and 3) to allow for rapid, interactive dialog with the network.

Development of the TAP subsystem beyond the functional specification stage has not been started; however, this subsystem has been proposed as the next logical development step after the TILF subsystem (completed) and the Data Location and Query Translation Facility (QIP module; currently under contract).

Modules included in the TAP subsystem are cross-hatched with slanted lines in the diagram. They include 1) the Data Request Command Interpreter, 2) the Data Request Dialog Processor, 3) the Network Data Catalogue Processor.

(2) Modules

(a) Data Request Command Interpreter

This processor will provide for a means of expressing a data request via a simple formal language consisting of simple verbs corresponding to Data

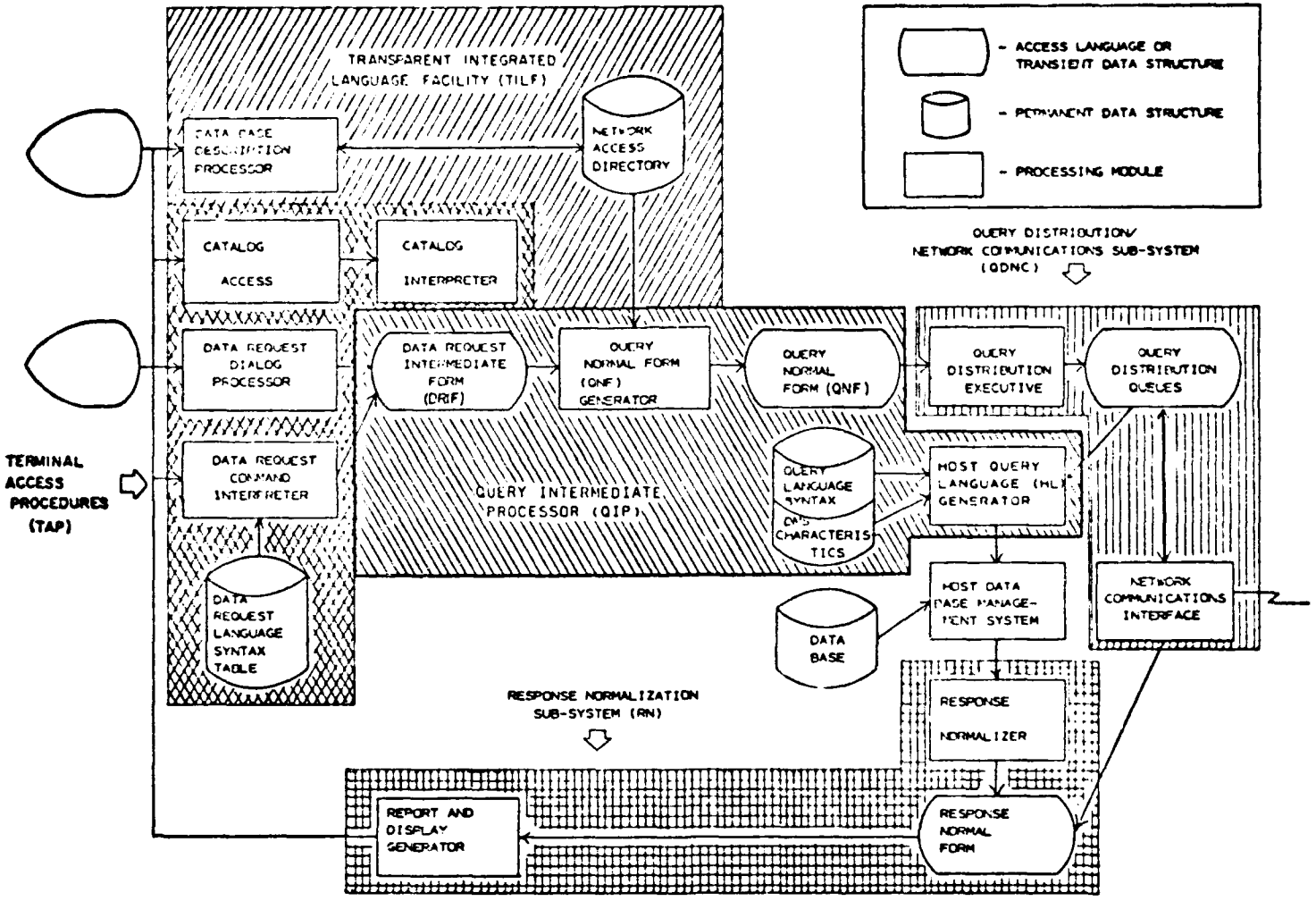


Figure B-1. Transparent Integrated Intelligence Network (TIIN) Relationship of Projects

Request Intermediate Form (DRIF) operators. The module will check the syntactic correctness of language commands from the user and translate formal nouns and verbs to their DRIF correspondents.

(b) Data Request Dialog Processor

This module provides a method for the analyst to interrogate network data bases without using a formal language. The processor guides the analyst through a process of interest negotiation wherein (s)he makes an inquiry through the medium of the Terminal Transparent Display Language (TTDL). The inquiry is based on information provided to the analyst from the Network Access Directory. Using information drawn from the analyst by language-free interaction, the Data Request Dialog Processor will build a query phrased in the Data Request Intermediate Form (DRIF).

The function of this module, then, is to guide the analyst through output specification and query qualification and to translate the analyst's responses into queries phrased in DRIF. The module will interact with the Network Access Directory and feedback helpful information to the analyst. Thus, if synonymous names are given for data elements, the system will feedback the standarized names, if requested. When the analyst does not specify data sources, the system will advise him of possible sources and allow him to specify a choice or to set priorities. The system will further provide the anticipated response time for each file, and advise the analyst when the data was last updated.

(c) Network Data Catalogue Processor

The function of this module is to allow analysts to see descriptions of network data base contents which may help them in identifying the precise data they require. A great deal of the descriptive information which is maintained in the Network Access Directory for system use is also useful to the analyst. The Catalog Access displays the contents of the NAD in a way meaningful to the user. Additional indexing techniques and the provision of a user interface to display NAD information are the concern of the Catalogue Processor.

It will provide data base administrator oriented data base description statements relating to specific subject categories. Update procedures are allowed for in the module's function. Interface procedures for displaying NAD information by subject and interacting with analyst-selected data description requirements are the chief functions of the processor.

With reference to the latter of the above mentioned functions, the processor will display available subject categories, prose descriptions of files related to subjects the analyst selects, prose descriptions of data elements belonging to selected files, and properties and attributes of selected elements.

The nature of the linkage between the catalog and the Network Access Directory is included in the catalog structural design to facilitate the interpreting function between analyst needs and NAD contents.

The catalog is to be implemented as a table driven process. No reprogramming is required to accommodate data base changes.

b. Transparent Intelligence Language Facility (TILF)

(1) TILF Subsystem

This subsystem is devoted to describing the contents of varied data bases to the network and keeping these descriptions up-to-date for real-time access from any node in the network. The subsystem includes the Data Description Processor and the Network Access Directory (NAD). It is indicated in the diagram by NE/SW diagonal lines. The TILF subsystem has been designed and implemented under the current contract.

(2) Elements of the Subsystem

(a) Network Access Directory (NAD)

The Network Directory (NAD) is the repository for all information required by the system about the contents of network data bases. Individual versions of the NAD will be created and maintained at each network node, and either all or part of each local NAD must be able to be easily transmitted to other network nodes on request. This feature will enable each node to periodically gather information from its own and other NADs, consolidate the information, and use it to locate data throughout the network.

The largest segment of a local NAD will be a file each entry of which contains descriptive information about a local data element, a network standard element (which may or may not have a corresponding local element), or a local-usage name for a network data element. Other locally-maintained NAD files will contain descriptive information about the parent files and parent data bases of local elements and about legal values or ranges of values for local elements. The development of a network data catalog will add a subject index file to the NAD, and the development of the query distribution module requires the addition of routing information for remote data bases.

Data element entries must be accessible in two ways--directly by the name of the element and via pointers from other component files of the NAD which categorize the element as being related to specific subjects or as being synonymous with other element name(s). Normally, specific data elements will be "found" when an analyst expresses an interest in a particular subject; however, from frequent use of the network system, the analyst may choose to refer by name to elements he has used before, thus bypassing perusal of subject categories. Consequently, the method chosen for organizing and maintaining the data element file must be one which allows the maximum efficiency possible for each individual mode of access without undue sacrifice of the other mode. Included in the information which may have to be maintained about a data element are:

- Name
- Element type (string or numeric)
- Length and format information (if string)
- Value range (if numeric)
- Pointer to parent file
- Pointer to parent data base
- Pointer to network synonym
- Pointer to value list
- Location (pointer to appropriate routing information)
- Chain pointer(s)

(b) Data Description Processor

The Data Description Processor is one of the user interface modules of the TIIN network. Unlike the network query and retrieval functions available at the TIIN node, which may be used by any analyst who has access to the local computer facility, access to the data description function should be limited by password protection. Its use should be restricted to the authorized administrator of the local data base.

The Data Description processor creates and updates the local NAD segment. Its function is to provide a means for the local data base administrator to describe to the network those elements in local data bases which may be made accessible to other network users. In addition to the initial description facilities, the Data Description Processor must provide update capabilities compatible with the real-time nature of network user requirements.

The mode of operation for the data description function should be interactive and guided by the Processor rather than the user. While the data base administrator is required to have more specialized knowledge than a consumer user of the network, the focus of that knowledge should be on the structure and content of the data base rather than on complex system procedures.

c. Query Intermediate Processor (QIP)

(1) Data Location and Query Translation Facility

This subsystem deals with the data location and query translation facility of the TIIN. It is composed of two main parts--those modules at the source node and those at the target node.

The source node will comprise two modules whose functions are 1) location of data bases, elements, and files containing the information requested by the user as indicated in the DRIF, and 2) translation of queries into a system-wide normal form for transmission to the target node.

The target node will contain one QIP module dedicated to translating the normal form data request (Query Normal Form) into the local DBMS language. The QIP module is currently under development as a separate contract effort.

QIP components are indicated in the diagram with NW/SE diagonals.

(2) Processors

(a) Query Normal Form Generator

The QNF processor is a part of a sequence of modules that transform a query in a user's native language into a query in a different query language. The processor transforms the DRIF (Data Request Intermediate Form) into the QNF (Query Normal Form). At the first stage of TIIN development, the QNF processor has two main functions, element name translation and data location. At later stages, the QNF processor will include more sophisticated data location algorithms as well as additional functions. One such function will be the creation of a file containing query status information.

The sequence of actions through which the query will be processed include:

- Identification of the file or files which contain the data elements
- Substitution of correct file-specific data element names when the analyst phrases his query in terms of synonyms
- Optimization of the query to take advantage of inverted lists which may exist. Inverted lists speed the retrieval process if the elements for which they are provided are searched first
- Creation of equivalent versions of a query for addressing to different files which may contain similar data
- Creation of sequences of queries when several files must be searched in sequence to retrieve data.

The function of the QNF Generator relates to TILF, in particular, in terms of file selection, facilitated through the NAD. The function of file selction according to the requirements of a data request proceeds according to the following criteria:

- o The file must contain all of the data elements referred to in the qualification statement
- o The qualification statement must be true of at least some records in the file, or at least there must be some a priori probability that it will be true
- o At least some of the data elements listed in the retrieval list must be contained in the file
- o The file must meet any conditions of recency or reliability stated by the user.

(b) Query Language Processor

The Query Language Processor will transform data requests received from a source node in the Query Normal Form (QNF) into the query language of the target data management system. The QL processor will receive the QNF and will operate on it in a tree format, obtained by transforming the Polish form expression in the QNF. At various points during the QNF-QL translation process it will be necessary to restructure the QNF in order to make it compatible with or more efficient in the target language.

Manipulations to be performed on the restructured QNF will be:

- o QNF-QL function reconcilation. Distinctive functions in the QNF not provided for in the QL will be reconciled. e.g., inverted names, geographic search

- o Value/Element Name translation

- o Query Validation to check for the presence of QNF values and element names in the target file

- o Query generation in the QL of the target data base.

d. Query Distribution Executive/Network Communication Interface (QD/NC) Subsystem

(1) Subsystem Description

This subsystem deals with 1) routing of queries to data resources, both local and external, 2) tracking and logging of incoming and outgoing network communications, and 3) the initiation of transmission of data from local to external nodes.

Elements of the QD/NC are indicated with vertical lines.

The QD/NC subsystem has not been developed beyond the functional specification stage but is considered to be a logical follow-on to the development of the QIP module (currently under contract).

(2) Modules

(a) Query Distribution Executive (QDE)

This module tracks and routes queries and responses. For incoming queries, it makes an entry in the incoming query queue with status of "awaiting processing". Some type of local-user-simulation software may be required to feed the query into the local DMS and to trap the response for normalization. The Query Distribution Executive polls the outgoing query queue for entries with "response received" status. Upon finding one, the Distribution Executive notifies the report and display function passing the MSN of the response file. The Query Distribution Executive changes the queue entry status to "null".

(b) Network Communications Interface (NCI)

The Network Communications Interface (NCI) is the TIIN element which initiates transmission of data from the local node to any other network node and receives data into the local node from other network nodes. The bulk of the data which will pass through the NCI will consist of:

- Queries being distributed to other network nodes for processing
- Incoming responses to distributed queries
- Queries coming into the node from other network nodes
- Outgoing responses to queries from other nodes.

In addition, periodically, each node will be required to send data from its local network access directory (NAD) to other network nodes and to receive NAD information from other nodes. The types of transmission data associated with NAD activity include:

- Bulletins from one network to all other nodes to announce that an NAD update has taken place at the originating node
- A request from one network node to a second node for transmission of the second node's NAD segment
- An NAD segment transmitted from one network node to another.

For outgoing "messages" (any data transmissions between network nodes), the NCI will set up a TISS-format header block containing the message sequence number (MSN), user identification, address (TIIN), message type (query, query response, NAD bulletin, etc.), and other standard information required by TISS. The NCI will remain in a state of readiness for handling outgoing message traffic by polling three queues: the outgoing query queue (for queries to be sent), the incoming query queue (for responses to be sent). Incoming message traffic addressed to TIIN will be directed through the NCI, with most message types being passed immediately to other network elements for processing.

e. Response Normalization (RN)

The Response Normalizer (RN), in conjunction with the Response Normal Form (RNF), is designed to bring about output standardization in the overall TIIN design. In general, the task of the RN is to provide the capability for converting data to a standard format once it has been retrieved from the local DB. Since data which is returned in response to a single query may be from separate files, it is likely to exhibit different characteristics with regard to coding conventions, field width, and numeric representation, etc. The RN normalizes responses, converting them to a standard format which enables collation and reconciliation of substantive conflicts and duplications. The RN also facilitates storage of data in the analyst's private file where it would be available for further processing (such as refining the original request, sorting, statistical analysis, graphic display, etc.) While this latter function is performed later in the TIIN flow, the RN makes it possible by producing retrieved data in a normalized form.

The RN subsystem is cross-hatched with perpendicular lines in the diagram.

The RN subsystem has not been developed beyond the functional specification stage but is considered to be a logical follow-on to the development of the QIP module (currently under contract).