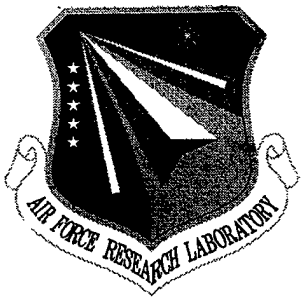
AFRL-IF-RS-TR-1998-161
Final Technical Report
July 1998



TRANSNETS, ENERGYNETS & ADVANCED TECHNOLOGIES

Sterling Software

Andrew W. Farrell and Richard A. Schaaf

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-1998-161 has been reviewed and is approved for publication.

APPROVED: John E Frank
JOHN E. FRANK
Project Engineer

FOR THE DIRECTOR:

JOSEPH CAMERA, Deputy Chief
Information & Intelligence Exploitation Division
Information Directorate

If your address has changed or if you wish to be removed from the Air Force Research Laboratory Rome Research Site mailing list, or if the addressee is no longer employed by your organization, please notify AFRL/IFEB, 32 Brooks Rd, Rome, NY 13441-4114. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	Jul 98	Final Aug 96 - Feb 98

4. TITLE AND SUBTITLE

TRANSNETS, ENERGYNETS & ADVANCED TECHNOLOGIES

5. FUNDING NUMBERS

C - F30602-94-D-0007/22
PE - 31335B
PR - 2183
TA - QJ
WU - 31

6. AUTHOR(S)

Andrew W. Farrell and Richard A. Schaaf

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

PRIME
Sterling Software: Information Technology Division
Beeches Technical Campus, Rt 26N
Rome, NY 13440

SUB
PRC
1410 Wall St.
Bellevue, NE 68005-3687

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/IFEB
32 Brooks Rd.
Rome, NY 13441-4114

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

AFRL-IF-RS-TR-1998-161

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: John E. Frank, IFEB, 315-330-3296

12a. DISTRIBUTION AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Documents evaluations of COTS software and prototypes developed to enhance developing capabilities related to overall objectives of MIDB program to produce, display and disseminate general military information.

14. SUBJECT TERMS

Transportation Networks, Energy Networks, Lines of Communications, Ad Hoc Queries, Spreadsheet Based Data Entry, Organization Chart Software, Temporal Analysis

15. NUMBER OF PAGES

36

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

TABLE OF CONTENTS

SECTION 1 INTRODUCTION

1.1 Identification

This Final Technical Report (FTR) describes the activities performed by Sterling Software under the Air Force Research Laboratory (AFRL) Information Directorate at Rome Research Site (formerly Rome Laboratory) sponsored program entitled "Transnets, Energynets and Advanced Technologies Research and Development (ATD)." This task was a research and development (R&D) effort conducted under Task Ordering Agreement (TOA) contract number F30602-94-D-0007, Task 022 and fulfills contract deliverable A008. Work on the Transnets, Energynets portion of this task was performed by PRC Inc. (subcontract number 7326-PRC-008). The technical report for the Transnets, Energynets portion was provided to the government by PRC Inc. under separate cover. PRC Inc.'s Transnets, Energynets final technical report is provided in Appendix A to this document.

The objective of the ATD portion of this task was to evaluate and apply advanced technologies to the near term Modernized Integrated Database (MIDB) development activities and to enhance the initial prototypes developed under TOA Task 0008. This included an evaluation of Commercial/Government Off-The-Shelf (COTS/GOTS) technology for future integration into MIDB. This final technical report contains a summary of the MIDB program and the ATD objectives, results of the evaluation activities performed under this contract, description of the prototype work developed under this contract, and MIDB analyst comments with regard to the prototype software obtained at an interim demonstration.

1.2 MIDB Background and ATD Objectives

MIDB is the intelligence production migration system identified by the Department of Defense (DoD) and will be used to produce, display and disseminate General Military Intelligence (GMI) information. MIDB 2.0 is a complete redesign of MIDB 1.0 that combined two existing intelligence systems, the eXtended Integrated Data Base (XIDB) and the Military Intelligence Information Processing System (MIIPS). The end result will be a seamless pass-through of intelligence data from national level intelligence centers down to the warfighters. The day-to-day MIDB development activities are managed by the Defense Intelligence Agency (DIA); however, overall program guidance is provided by a Program Management Board comprised of representatives of the user community.

1.3 Task Overview and Scope

The primary objective of the ATD portion of this task is to evaluate related commercial and government technologies that have applicability to the overall objectives of the MIDB program, and to enhance the prototype software developed under TOA Task 0008. This includes the evaluation of GQL (Graphical Query Language) and the NeXS Spreadsheet application and enhancements to the Ad Hoc Query Editor (AHQE), Temporal Analysis System (TAS), and the UNIT Subordination Organization Chart application (Org Chart).

1.4 Organization of Document

This document is composed of five sections:

Section 1, Introduction, identifies the work performed on this task.

Section 2, Advanced Technology Evaluations, contains an evaluation of GQL and NeX3 spreadsheet applications.

Section 3, Prototype Enhancement Description, describes the enhancements made to the prototype software delivered under IRD TOA Task 008.

Section 4, Follow-On Activities, contains suggestions on enhancements that can be made to the prototype software in support of MIDB.

Section 5, Notes, lists and defines acronyms and abbreviations used throughout this document.

Appendix A, is PRC Inc.'s Final Technical Report for the Transnets and Energynets portion of this task.

SECTION 2 ADVANCED TECHNOLOGY EVALUATIONS

Under the Advanced Technology Research and Development task, GQL was evaluated as a potential AHQE for integration with MIDB for purposes of providing the novice user a way to query MIDB, and the NeXS Spreadsheet application was evaluated as a potential Database Administrator (DBA) tool for purposes of performing database maintenance. For each application, the following sections contain a brief introduction of the technology and an evaluation of the product with respect to its particular use with MIDB.

2.1 Ad Hoc Query Technology

An ad hoc query tool is a program that provides a novice user a way to access a database. It also may provide a quick access to data for the experienced user. A well developed query tool can offer many advantages to both the experienced user as well as the novice. There is a substantial learning curve associated with developing the skills to effectively query a relational database using a structured query language (SQL). Ad hoc query tools provide the user with an interface into the database that will speed up the learning process of accessing the data in a relational database as well as, in some cases, provide the user with some insight into the structure of the data.

2.1.1 Andyne GQL Evaluation

GQL is a collection of query and reporting tools that was developed by Andyne Computing Limited of Kingston, Ontario (COTS). The GQL environment presents the user with a visual data model of a specific database that is used to create queries and produce reports. The goal of the GQL line of products is to make running a query or report as easy as possible by hiding SQL concepts and syntax from the user. The user is able to add data items to the report or query criteria from a data model (i.e., Entity Relationship Diagram of the database).

The emphasis of this evaluation focused on the creation of complex queries. The following sections describe a class of SQL query and how GQL could or could not be used to perform such queries. The queries selected as the evaluation criteria were obtained from an article regarding queries to test query tool merit, (*Killer Queries*, Datamation, 1 November 1995, pp 1-8, http://www.datamation.com/ PlugIn/issues/1995/nov1/11aevext.html).

2.1.2 Counting and Correlated Subqueries

Aggregate counting functions in conjunction with subqueries provide the capability to restrict the rows returned in a query based on the results (e.g., sums, averages, or counts) of a subquery. For example, given a typical company database that contains a Department table and an Employee table such that each Department has many Employees, show the departments that have more than 2 secretaries and fewer than 15 employees. The key in specifying this query is using a correlated subquery in a HAVING clause to attach the count of secretary positions in Employees to each department. The SQL for this query follows:

```
SELECT D1.deptno, D1.deptname
FROM Departments D1
GROUP BY D1.deptno, D1.deptname
HAVING COUNT(*) < 15
AND 2 < (SELECT COUNT(*)
FROM Employees E1
```

```
WHERE E1.job = 'secretary'
AND E1.deptno = D1.deptno);
```

GQL cannot perform this type of query without modifying the SQL directly since no graphical user interface (GUI) mechanism is provided to correlate a subquery with the main query.

2.1.3 Quantified Queries

Quantified queries restrict the results returned from a query given some quantity. For example, given a sales database that contains a Sales Rep table and a Sales table such that each Sales Rep has many Sales, find all reps for whom every sale has been more than $500. One way to create this query is to restrict the main query with a subquery and the quantity 500. The SQL follows:

```
SELECT S0.repno, S0.repname
FROM Sales S0
WHERE 500.00 < ALL (SELECT salesamt
FROM Sales S1
WHERE S1.repno = S0.repno);
```

This is another type of the correlated subquery listed above. In addition to not supporting the correlated subquery, GQL also does not support adding the "ALL" predicate without modifying SQL code directly. Another way of performing this query is to specify the quantity in the subquery as opposed to the main query. The SQL follows:

```
SELECT repno, repname
FROM Sales
WHERE repno NOT IN (SELECT repno
                    FROM Sales
                    WHERE salesamt <= 500)
AND repno IN (SELECT repno
              FROM Sales
              WHERE salesamt >500);
```

GQL can support this type of query but would require the user to create two subqueries in addition to creating the main query. With GQL, subqueries are created before the main query and must be saved to disk before they can be included in the main query.

2.1.4 Ranking Results

Ranking the results of a query is more complicated than simply ordering the results. Often the results of a query need to be grouped and ordered to provide meaningful results. Also, if the user is not interested in all rows of a table, the results need to be restricted for each group. For example, given a customer database where a CUSTOMERS table contains the State and the Total Sales for the current year, for each state, find the top ten customers based on this year's sales. The SQL follows:

```
SELECT *
FROM Customers C0
WHERE totalsales IN (SELECT C1.totalsales
                     FROM Customers C2, Customers C1
                     WHERE C1.state = C2.state
                     AND C1.state = C0.state
                     AND C2.totalsales >= C1.totalsales
```

```
        GROUP BY C1.totalsales
        HAVING COUNT(DISTINCT C2.totalsales) <= 10)
ORDER BY state, totalsales DESC;
```

Again, the user would need to modify the SQL directly to perform the correlated subquery required by this type of query.

2.1.5 More on Correlated Subqueries

With SQL, subqueries can also be placed in the select statement of a query. Often this is required to show different sets of aggregate data such as sums or counts in the same query. For example, in a Sales database that contains a SALESPERSON table and SALES table, show last years total sales and this years total sales for each salesperson. This type of query requires the use of scalar subqueries in the select statement of the main query. A scalar subquery is a subquery that returns one row. The SQL follows:

```
SELECT P1.salesperson,
   (SELECT sum(S.sale)
    FROM SALES S, SALESPERSON P2
    WHERE P2.salespersonid = S.salespersonid
    AND P2.salespersonid = P1.salespersonid
    AND S.saledate < "1/1/98") as Total97,
   (SELECT sum(S.sale)
    FROM SALES S, SALESPERSON P2
    WHERE P2.salespersonid = S.salespersonid
    AND P2.salespersonid = P1.salespersonid
    AND S.saledate >= "1/1/98") as 98Sales,
FROM SALESPERSON P1
```

Again, the user would need to modify the SQL directly to perform the correlated subquery required by this type of query.

2.1.6 Conclusion

GQL was evaluated under TOA Task 0008 with respect to administrative requirements, reporting capabilities, and its ease of use. We concluded that the interface was cumbersome for creating complex queries; the reporting capabilities were robust; and that a significant amount of administrative effort would be required in maintaining a GQL model of MIDB (reference Task 0008, Final Technical Report). We found the current version of GQL to be no different in these regards. In addition, GQL lacks a way to correlate subqueries from the GUI interface. As an AHQE, this severely limits GQL's ability to specify complex queries without modifying the SQL directly.

2.2 Spreadsheet Technology

Spreadsheet products provide the user with a set of business analysis tools. Most packages will contain the tools to perform data analysis through sorting, calculations, and reporting. These tools consist of worksheets, charts and graphs, drawing tools, and macros. A worksheet is the main work area in a spreadsheet application. It provides the user a place to view data as well as perform any data manipulations. The data is enhanced within the worksheet by applying functions, macros, and including graphical elements to enhance the data visually.

To aid in analyzing the data displayed within a spreadsheet, many spreadsheet applications provide the capability to view the data in different formats such as charts or graphs. The use of graphs and charts can offer unique insight into the data that may not be recognized when viewing the data in its raw format. Once the data has been entered into a spreadsheet, it can be formatted through the use of drawing tools, font styles, and other formatting options. These tools allow the data to be presented in a format that makes it easy to understand.

2.2.1 NeXS Spreadsheet Evaluation

The NeXS spreadsheet application was chosen for this study due to its full feature set and the plug and play type architecture that allows extensions to be written in the C or the Perl programming languages. Using the NeXS application program interface (i.e., Connections API), a client server application can be fully integrated with the spreadsheet. The Connections API allows a program to remotely control all aspects of the spreadsheet, including drawing graphs, printing, and performing calculations.

The issues involved with integrating NeXS with a MIDB include getting data into the spreadsheet from the database and updating the database as changes are made to the spreadsheet.

2.2.2 Getting Data into a NeXS Spreadsheet

The Perl API utilized by NeXS provides access to the full set of NeXS API extensibility functions. For example, the API provides functions that allow a Perl program to insert data into a specific cell of a worksheet and modify the main NeXS menu.

The AHQE was used as the front end to NeXS since the Perl API was easily integrated with this software. The results of a query generated by the AHQE is exported to NeXS via the Export To... button menu on the main AHQE display. The NeXS application is executed automatically by the AHQE. A worksheet called "MIDB_SHEET" is automatically created and is updated with the results of the query. The report item names are also included in the worksheet. For example, if the AHQE was used to retrieve UNIT_IDs and UNIT_NAMEs from the UNIT table, the MIDB_SHEET worksheet would contain the column headings UNIT_ID and UNIT_NAME in the first row of the worksheet.

2.2.3 Modifying Spreadsheet Data

Three functions are provided on the MIDB_SHEET worksheet that allow the user to modify MIDB data based on modifications to the worksheet, Modify Cell..., Modify Cells..., and Modify Column....

Modify Cell... Allows the user to modify the value of one cell in the worksheet. Upon execution of this function, the user is prompted to select the cell to be updated and enter the new value for the cell. A message will appear on the bottom left of the worksheet indicating how many database rows have been updated.

Modify Cells... Allows the user to modify the value of all cells in a column that equal the value of the selected cell. Upon execution of this function, the user is prompted to select the cell to be updated and enter the new value for the cell. A message will appear on the bottom left of the worksheet indicating how many database rows have been updated.

Modify Column... Allows the user to modify the value of all cells in the column of the selected cell. Upon execution of this function, the user is prompted to select a cell in the column to be updated and enter the new value for the cell. A message will appear on the bottom left of the worksheet indicating how many database rows have been updated.

NOTE: These functions are provided under a custom menu called "MIDB" that will appear to the right of NeXS menus on the main menu bar. Changes made directly to the spreadsheet are not committed to the database. Also, the MIDB menu will only appear when the results adhere to the following two rules: namely, one database table is used in the AHQE query and the primary key elements of the table is contained in the report. Primary key information is required in the worksheet in order to make the appropriate database updates.

2.2.4 Conclusions

Over the course of this task and TOA Task 0008, several spreadsheet packages were considered for integration with MIDB. None provided the mechanisms for integration with a database application that NeXS does. The low cost and complete feature set make NeXS a logical choice for integration with MIDB. Certain NeXS reporting features should also be evaluated. The extensibility features of NeXS would also allow custom graphs and charts to be automatically created and printed from MIDB transparently.

SECTION 3 PROTOTYPE ENHANCEMENT DESCRIPTION

3.1 Ad Hoc Query Editor

The AHQE has undergone many revisions in support of new MIDB requirements, new SQL (the structured query language used by Sybase) features, and other prototypes developed on this task.

3.1.1 Overview

The AHQE is an adaptation of the Ad Hoc Query tool provided in Combat Intelligence System (CIS)—GOTS—and was developed using Perl Tk and the Sybperl toolkit (Sybase extensions to Perl). It provides MIDB analysts the ability to perform complex queries without the knowledge of SQL. It can also be used to familiarize MIDB analysts with SQL and MIDB by allowing the user to view the SQL generated by the AHQE, or browse the MIDB structure, respectively.

3.1.2 Enhancements

The AHQE was enhanced significantly during the course of this effort. Functionality was added to support the creation of subqueries, adding computations and string functions to a report or query criteria, a progress indicator, along with additional functions in support of the MIDB 2.0 Tie Table convention, and functions that utilize additional SQL-92 features implemented in Sybase (e.g., inner and outer joins). Furthermore, the AHQE was modified to export query results to the other prototypes developed on this effort.

3.1.2.1 Subqueries

The AHQE now supports subqueries. Subqueries provide a way to restrict the rows returned in a main query or to show different sets of aggregate data such as sums or counts in the same query. Subqueries can be correlated or uncorrelated. A correlated subquery is affected by rows returned in the main query, while an uncorrelated subquery is not affected by the main query.

In the AHQE, a subquery is added to a main query from the Load button on the main display. Upon execution of this function, the user is presented with an open file dialog box that allows the user to select a previously saved query to be added to the main query. After selection of a query on disk, the user is required to enter whether the query should be correlated or uncorrelated.

Subqueries can also be added to the report items. This allows the user to show different sets of aggregate data such as sums or counts in the same query. Subqueries added to a report must return one row only. A subquery is added to the report via the Report Options button. In the resulting dialog box, select the menu named "Add to Report...". Upon execution of Add Subquery... menu item, the user is presented with an open file dialog box that allows the user to select a previously saved query to be added to the report. After selection of a query on disk, the user is required to enter whether the query should be correlated or uncorrelated.

3.1.2.2 Computations and String Functions

Built in computations and string functions allow the format and contents of the results of a query to be modified from how they are stored in the database. The results returned by a query can also be affected through the use of string and numeric functions.

3.1.2.2.1 Computations and String Functions as Report Items

Computations and string functions are added to a report via the Report Functions button on the main AHQE display. Upon selection of the Report Functions button, a dialog box appears with a scrolling list containing the current report items and the Add to Report menu. The String Function... and Computation... menu items under this menu allow the analyst to add a computation and string function, respectively.

Upon selection of the String Function... menu item, a dialog appears with two scrolling lists: one contains a list of the string functions defined in Sybase; the other contains the columns that can be used with the string functions (i.e., columns of char or varchar datatype). The analyst specifies a function by highlighting a column in the column scrolling list and double-clicking on a string function. The function and column appear in the data entry area above the scrolling lists for manual editing (e.g., the analyst must manually enter the indices required by the substring function). Selecting the Add button adds the function to the report items of the query. Selecting the Cancel button closes the dialog box leaving the query unchanged.

Upon selection of the Computation... menu item, a dialog appears with a scrolling list containing the columns that can be used with the numeric operators (i.e., columns of int, binary, float, etc.). Next to the scrolling list appears a keypad and the numeric operators defined in Sybase (i.e., +, -, /, and *). The analyst specifies a function by double-clicking a column in the column scrolling list and adding numbers and operators from the keypad. The numbers, operators and columns appear in the data entry area above the scrolling lists. Selecting the Add button adds the function to the report items of the query. Selecting the Cancel button leaves the query unchanged. Note, columns are not required to be added to the numeric function. Also, more than one column can be added to the function.

3.1.2.2.2 Computations and String Functions as Query Criteria

Computation and string functions are added to the query criteria via the String Function and Computation buttons on the main AHQE display above the Query Criteria scrolling list. Adding functions to the query criteria restrict the results of a query based on some computation. The String Function and Computation buttons work exactly as documented above for the String Function... and Computation... menu items, respectively.

3.1.2.3 Progress Indicator

A progress indicator (i.e., thermometer) has been added to the AHQE on the bottom of the screen, similar to the progress indicators included on today's World Wide Web browsers such as Netscape. Upon execution of a query, the message "Query sent..." appears in the thermometer. While the Sybase SQL server executes the query, the AHQE appears and is unusable. After the Sybase server begins to return the results of the query, the thermometer is filled proportionally to the records being returned from the server. Once the server begins to return rows, the user can cancel the query at any time using the Cancel button. Canceling a query displays the results that have already been returned from the server and cancels the rest of the results.

3.1.2.4 MIDB 2.0 Tie Table Support

Previously with the AHQE, when "TIEing" two or more tables, the analyst was required to manually select the appropriate tie table, add the ASSOC data element to the report criteria, and specify the ASSOC value via the Add Value button. With the current version, the analyst is

simply prompted for an ASSOC value that is added to the query criteria when "TIEing" to tables.

For example, to tie the EQP table to the FAC table in MIDB, the analyst must specify a report item from the EQP table to add the EQP table to the query. Recall that initially, all MIDB tables appear in the tables scrolling list. Upon inclusion of the EQP table in the query, the EQP table will appear as the only table in the tables scrolling list. This indicates that the current table is the EQP table. And, since the EQP table has an associated tie table, EQP_TIE, the Show Ties button becomes selectable. The Show Ties button appears to the left of the Primary and Foreign buttons under the tables scrolling list on the main AHQE display. Upon selection of the Show Ties button, the tables scrolling list will contain all MIDB tables that have an associated tie table. Highlighting the FAC table will populate the elements scrolling list with the elements defined for the FAC table. Adding a FAC element to the report or query criteria adds the FAC table to the query and indicates that a tie should be made between the EQP and FAC tables. The analyst is then prompted to enter the ASSOC value for associating these tables in a standard data entry dialog box. The dialog box also contains an ASSOC Help button that provides a list of ASSOC values for the appropriate tables (in this case the EQP to FAC assoc codes and code descriptions would be displayed) to aid the analyst in specifying an appropriate ASSOC value. Upon specification of an ASSOC value, the query criteria is automatically updated.

NOTE: The Primary and Foreign buttons perform the same functions as the Primary Relations and Foreign Relations buttons did in previous releases of the AHQE. These buttons are selectable after a table has been added to the report. The Primary button updates the tables scrolling list with all tables related to the current table with a one-to-many relationship. The Foreign button updates the tables scrolling list with all tables related to the current table by a foreign key. A foreign key exists from table B to table A when the elements that comprise the primary key in table A are contained in table B.

3.1.2.5 Extended Support of Sybase SQL Features

The AHQE has been enhanced to utilize additional Sybase SQL features from previous releases. Inner and outer joins can now be specified, and *group by* and *having* clauses are now supported.

3.1.2.5.1 Inner and Outer Joins

Inner and outer joins are called left and right outer joins in Sybase. These clauses provide a powerful extension to the natural joins typically performed in complex queries. When tables (e.g., Table A and B) are joined in a relational database management system (RDBMS), what actually happens in memory is that a temporary table is created that contains all columns in each table and a permutation for each row in table A with table B. The resulting temporary table is then parsed for rows that match the query criteria (e.g., the join condition). For example, sample data in a typical SALES database that includes a SALESPERSON table and a SALES table where each salesperson has zero or many sales might look like the following:

SALESPERSON

ID	NAME
1	Joe
2	Sam
3	Sue

SALES

ID	SALEID	AMOUNT
1	1	50.00
1	2	100.00
3	1	200.00
3	2	100.00

A typical query on these tables may be to retrieve the total sales for each salesperson. The SQL follows:

```
select a.NAME, sum(b.AMOUNT)
from SALESPERSON a, SALES b
where a.ID = b.ID                    /* Note the "natural" join condition */
```

This query would yield the following results:

a.NAME	sum(b.AMOUNT)
Joe	150.00
Sue	300.00

The reason Sam was not included in the report is due to the fact that he did not have any sales, thus the query criteria excluded him from the report. What happened in memory as a result of performing this query is that a temporary table was created as described above. The resulting table follows:

TEMPORARY TABLE

a.ID	a.NAME	b.ID	b.SALEID	b.AMOUNT
1	*Joe*	*1*	*1*	*50.00*
1	*Joe*	*1*	*2*	*100.00*
1	Joe	3	1	200.00
1	Joe	3	2	100.00
2	Sam	1	1	50.00
2	Sam	1	2	100.00
2	Sam	3	1	200.00
2	Sam	3	2	100.00
3	Sue	1	1	50.00
3	Sue	1	2	100.00
3	*Sue*	*3*	*1*	*200.00*
3	*Sue*	*3*	*2*	*100.00*

The italicized rows are the rows that match the query criteria (i.e., the join condition a.ID = b.ID) and thus are the only rows used in calculating a salesperson's total sales.

To report on all salespeople, a left outer join must be specified to include all rows in the left table. The SQL follows (Sybase Syntax):

```
select a.NAME, sum(b.AMOUNT)
from SALESPERSON a, SALES b
where a.ID *= b.ID                   /* Note the left outer join condition */
```

This query would yield the desired results:

a.NAME	sum(b.AMOUNT)
Joe	150.00
Sam	0
Sue	300.00

Left and right outer joins are specified in the AHQE via the Table Options button on the main AHQE display. Upon selection of this button, a standard dialog box appears with a list of table pairs that will be joined in the current query. The tables are separated by the bar character (i.e., '|') that indicates a natural join will be performed upon execution of the query. A left outer join is specified by highlighting one of the entries in the scrolling list and selecting the Left Outer Join button. The current separation character will be replaced by the less than sign (i.e., '<') that indicates a left outer join will be performed upon execution of the query. A right outer join is specified by highlighting one of the entries in the scrolling list and selecting the Right Outer Join button. The current separation character will be replaced by the greater than sign (i.e., '>') that indicates a right outer join will be performed upon execution of the query.

3.1.2.5.2 Group By and Having Clauses

The *Group By* clause takes the results of the specified query and puts the rows into groups as having the same values for the grouped columns. Each group is reduced to a single row in the resulting report. The *Having* clause applies selection criteria to the resulting grouped rows. The order of the grouped columns in a query does not matter. For example, the group by and having clauses would be required in a query of MIDB to provide a count of subordinate UNITs for each UNIT in the UNIT table. The SQL follows:

```
select a.UNIT_ID, count(c.UNIT_ID)
from UNIT a, UNIT_TIE b, UNIT c
where a.UNIT_SK = b.TIE_FROM_SK
and b.ASSOC = "C"
and b.TIE_BOOL = 0
and b.TIE_TO_SK = c.UNIT_SK
group by a.UNIT_ID
```

This query can easily be specified with the AHQE. Upon selection of the UNIT table from the list of tables, add the UNIT_ID element to the report by double-clicking on it in the list of elements. Selecting the Show Ties button displays the MIDB tables that can be tied to the UNIT table. The UNIT table will be added to the report again by adding UNIT_ID to the report. The AHQE will then prompt for the ASSOC code to be used in the *tie* of the UNIT table to the UNIT table. An ASSOC of "C" stands for commands or controls (see Figure 1). The group by and having clauses can be added to an AHQE query through the Report Options button on the main AHQE display. Upon selection of the Report Options button, a standard dialog box appears with the current report items (i.e., a.UNIT_ID and c.UNIT_ID). The group by clause is added via the Group By button that appears at the bottom of the dialog box. Highlight the report item to be grouped (i.e., a.UNIT_ID) then select the Group By button to add this clause to the query (see Figure 2). The count function is added to the report via the Apply Function menu button (see Figure 3). Highlight the report entry to apply the count function to (i.e., c.UNIT_ID), then select the count item in the Apply Function menu button.

3.1.2.6 Integration with other Task 22 Prototype Software

The AHQE was enhanced to work as a front end to the other Task 22 prototypes. The Export To... menu button provides entries to export results of a query to TAS, the Org Chart software, and the NeXS Spreadsheet application. Please see the sections for the individual prototypes for a discussion on the use of the AHQE to support these applications.

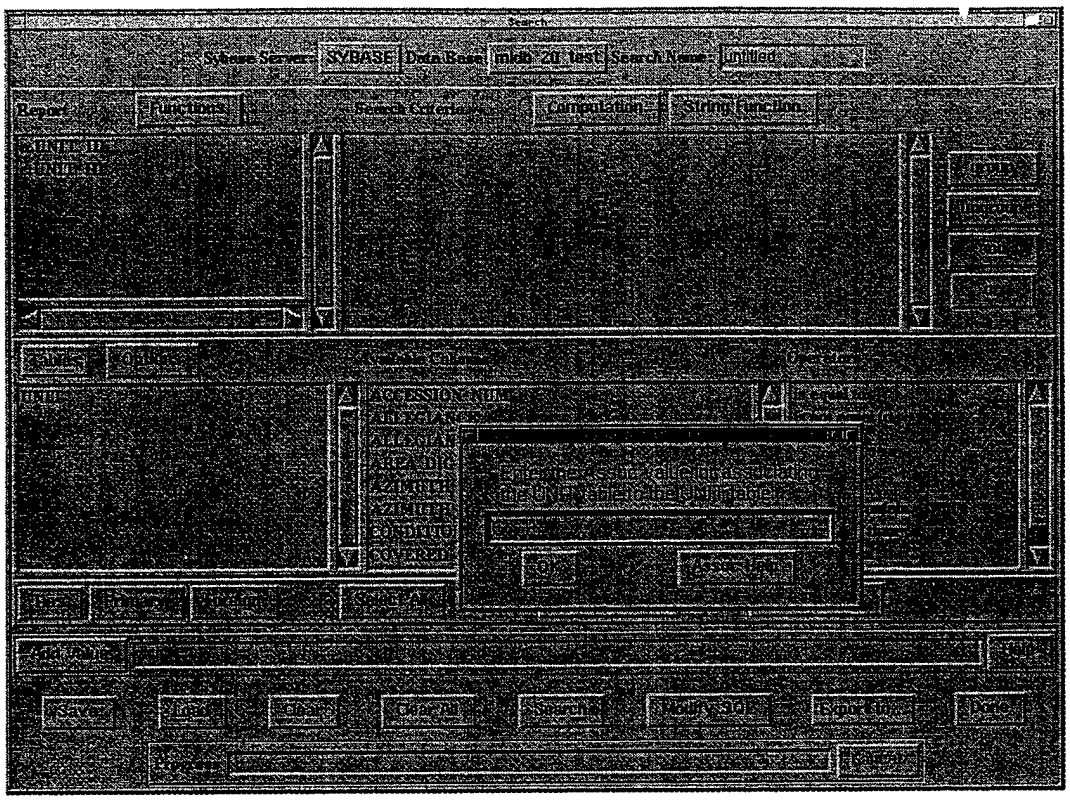


Figure 1: AHQE -- Prompt for ASSOC Value

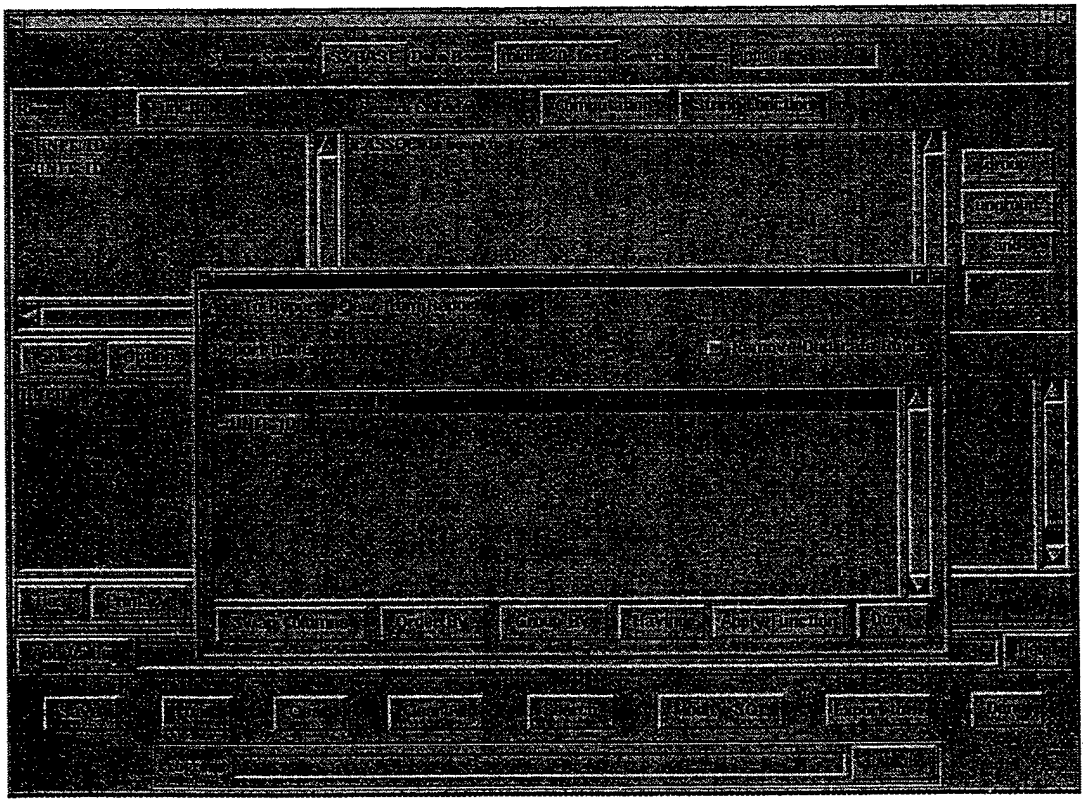


Figure 2: AHQE -- Specifying the "Group By" Clause

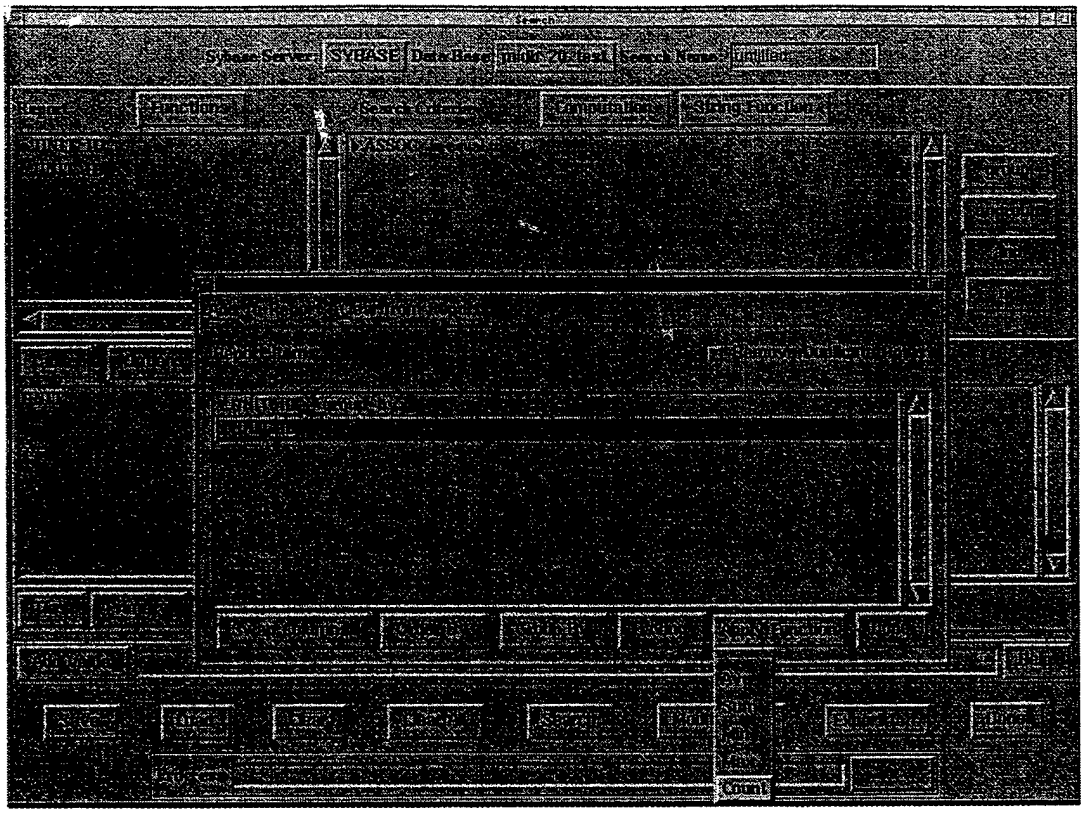


Figure 3: AHQE – Specifying the "Count" Function

3.2 Temporal Analysis System Evaluation

Previously in the Modernized Integrated Database (MIDB), historical data was used for administrative purposes only (e.g., IDBTF production). The historical data currently captured by MIDB (e.g., MIDB TRACK Table Structures) may prove to be useful to the MIDB analyst as well. Such historical information can be used to analyze Unit movement over time or analyze Event based information. The focus of this effort with respect to TAS, was to integrate the latest version of TAS (i.e., 4.0) and update the existing prototype software to utilize the TAS 4.0 enhancements.

3.2.1 TAS Description

TAS is a suite of tools consisting of six applications: Timelines, Maps, Chalkboards, Dictionary, a Domain Editor, and K-PASA (Knowledge-based Predictive Analysis and Situation Assessment). These tools provide assistance to the intelligence analyst in displaying and analyzing the results of time-based intelligence information. Timelines and maps aid the intelligence analyst by displaying the occurrence of events in a graphical format. The Domain editor allows the creation of custom event types. Previously, an unmodifiable set of events were delivered with TAS. K-PASA is a set of expert system tools that allows the analyst to build event models and apply imported event information against the models that provide insight into unfolding situations and expected future activity. A natural language explanation capability reports the likelihood that a set of events satisfies an event model.

A TAS event is an occurrence that is portrayed on a timeline or map by an icon or pattern combination. Each event has a date, time, and a duration associated with it. The TAS Timeline

application assists the intelligence analyst in building event timelines that can be used to identify repeated patterns of behavior, or to show the history of a sequence of events, whereas maps can be used to display a sequence of events, such as unit movement, geographically. These patterns of events can be used to perform situation assessment and predictive analysis.

3.2.2 Updating the TAS Timeline Prototype

As with the original prototype, the AHQE was used as the front end to TAS. The results of an MIDB query was imported into TAS and displayed in the TAS timeline display, such that each row returned by the query was displayed as an individual "general" Event. Once the data was imported into TAS, the analyst could access the full suite of analyst tools provided with TAS.

With TAS 4.0, an analyst can now create custom event types. A custom event type is built with the Domain Editor. Building a domain consists of creating the new domain and specifying its attributes through an easy to use GUI. Attributes are specified by a name and a datatype. Valid datatypes are similar to datatypes defined by Sybase (e.g., integer, text, etc.). Descriptive textual information can also be included.

The AHQE has been enhanced to recognize custom TAS event types defined by the analyst given the following requirement. The attributes defined in a custom event type must match MIDB element names for the AHQE to automatically export query results to a particular event type. For example, given a custom domain called "UNIT_TRACK" that is used to track UNIT movement and contains unit identification information (e.g., UNIT_ID, UNIT_NAME, OB_TYPE, and ECHELON) and unit location information (e.g., ILAT, ILON). For the TAS event type to be recognized by the AHQE, it would need to be specified as follows to be consistent with MIDB:

UNIT_TRACK
    UNIT_ID,
    UNIT_NAME,
    OB_TYPE,
    ECHELON,
    ILAT,
    ILON

The important thing to note is that each UNIT_TRACK attribute corresponds to an MIDB element name. The UNIT_TRACK domain is easily created with the TAS Domain Editor such that compatible datatypes are used as well.

To export data from MIDB into TAS via the AHQE, the analyst must specify a query using the AHQE, then select the Timeline Analysis option under the Export To... button menu on the main AHQE display. The AHQE will then try to match fields specified in the query report with attributes defined in the current event types specified in TAS. All event types compatible with the current query are presented to the analyst in a scrolling list. After the user selects the appropriate TAS event type, the AHQE exports the results of the query to a TAS Import file. The filename and location is specified by the analyst via standard create file dialog box. This file must manually be imported into TAS through the TAS interface. Under the Edit menu of either the Timeline or Map applications, the analyst can Import from Text file.... The imported events can be saved to the TAS database or displayed only.

3.3 Organization Chart Software

An organizational chart (org chart) graphically depicts the relationship between a commanding position and its subordinates, in this case military units. With the Org Chart software developed on this task, commanding units are displayed above and to the left of its subordinate units, with lines connecting the units. The resulting diagram is a directed graph where nodes indicate units and edges indicate lines of command. The Org Chart software delivered under Task 008 provided a dynamic interface such that sections of the Org Chart could be expanded and contracted for each commanding unit. The enhancements developed under this effort included the ability to identify a unit by its 2525 symbol code, a 'find' capability, balloon help, and allow the user to customize the number of levels of the org chart to be displayed upon execution. The AHQE was also enhanced to provide a front end to the Org Chart software.

3.3.1 2525 Symbol Code

Previously, units were identified in the org chart by a UNIT_ID or UNIT_NAME. This appeared next to a +, -, or 0 icon that indicated that the org chart was not displayed, displayed, or was a leaf node, respectively, from that point on in the org chart. This enhancement allows a 2525 symbol code to be used in place of the +, -, or 0. 2525 Symbol codes are generated by GOTS software called GSD (Geographic Situation Display) developed by TRW.

GSD creates a 2525 symbol in pixmap format and stores it to a working directory given a valid GSD symbol code. In MIDB, the GSD symbol code will be system generated and stored in a field called SYMBOL_CODE. The MIDB structure change required by this enhancement is documented in an MIDB change request.

The analyst can specify whether to use 2525 symbols upon execution of the Org Chart software in a standard configure dialog box.

3.3.2 Find

The Find function allows the analyst to search the entire org chart for occurrences of a specified string (e.g., substring of a UNIT_ID or UNIT_NAME). Upon finding a unit in the org chart that matches the search string, the org chart is expanded so the appropriate unit is displayed and the node in the org chart is highlighted (i.e., appears in reverse video).

The Find function is executed from the Edit menu and requires the user to enter the string to search for in a standard dialog box. The dialog box remains, allowing the analyst to perform multiple finds or find again, until the Done button is selected.

3.3.3 Balloon Help

Balloon help is activated via the Edit menu and toggles balloon help on and off. Balloon help is provided for each unit in the org chart. When balloon help is activated, additional identification information is provided for a unit when the cursor is placed over a unit in the org chart. The additional information includes the ASSOC value in MIDB that associates the unit to its commanding unit and the unit name or unit id.

3.3.4 Customize Levels

Previously, upon execution of the Org Chart software, the org chart would appear fully expanded. This enhancement allows the user to specify the number of levels the org chart

should be expanded to upon execution. Level One indicates that only the root nodes of the org chart be displayed, Level Two indicates that all root nodes and their immediate subordinates be displayed, and so on.

The analyst is presented with a dialog box upon execution of the Org Chart software that can be used to specify this information. The default is Level Two. There is also an option to Display All units on the org chart.

3.3.5 AHQE Export Update

For testing purposes, the AHQE was enhanced to create the appropriate Sybase temporary table used by the Org Chart software to display a unit hierarchy.

Any AHQE query that contains the UNIT_ID field can be used to drive the Org Chart software. The resulting set of UNIT_IDs are sent to a module that determines all subordinate units for each unit in the result set. The resulting set of units are entered into a the temporary table and displayed appropriately by the Org Chart software. The Org Chart software is also executed by the AHQE.

The export for the Org Chart software is executed via the Export To... menu button.

SECTION 4 FOLLOW-ON ACTIVITIES

The TAS, Ad Hoc Query, and Org Chart prototypes were highly regarded by the MIDB community and were easily integrated into the MIDB application. Possible enhancements to the existing prototypes include:

a. Updating the Org Chart software to display 2525a Symbology instead of 2525 Symbology

b. Updating the TAS prototype to work correctly with the most recent version of TAS (i.e., 4.2.1) to provide an automated way of importing event information through the use of UNIX sockets

c. Updating the Ad Hoc Query editor to support all SQL-92 features as implemented in Sybase

Further developments could be made to the NeXS spreadsheet application with respect to creating ad hoc graphs and charts.

SECTION 5 NOTES

This section lists and defines acronyms and abbreviations used throughout this document.

AFRL	Air Force Research Laboratory
AHQE	Ad Hoc Query Editor
API	Application Programmer's Interface
ASSOC	Association
ATD	Advanced Technologies Research and Development
CIS	Combat Intelligence System
COTS	Commercial-Off-The-Shelf
DBA	Database Administrator
DIA	Defense Intelligence Agency
DoD	Department of Defense
EQP	Equipment Table in MIDB
FAC	Facility Table in MIDB
FTR	Final Technical Report
GMI	General Military Intelligence
GOTS	Government-Off-The-Shelf
GQL	Graphical Query Language
GSD	Geographic Situation Display
GUI	Graphical User Interface
ID	Identification
IDBTF	Integrated Database Transaction Format
ILAT	Integer Latitude
ILON	Integer Longitude
K-PASA	Knowledge-based Predictive Analysis and Situation Assessment
MIDB	Modernized Integrated Database
MIIPS	Military Intelligence Information Processing System
Org	Organization
R&D	Research and Development
RDBMS	Relational Database Management System
SQL	Structured Query Language
TAS	Temporal Analysis System
TOA	Task Ordering Agreement
XIDB	eXtended Integrated Data Base

APPENDIX A

TRANSNETS, ENERGYNETS FINAL TECHNICAL REPORT

The attached Transnets, Energynets Final Technical Report was prepared by PRC Inc. under contract number F30602-94-D-0007, Task 022 (subcontract number 7326-PRC-008) for the Information Technology Division of Sterling Software, Rome Department.

TRANSNET Technical Efforts:

Requirements Analysis was conducted through a series of discussions with Government PMO and Analyst (user) representatives to design a production capability with a map-based GUI to facilitate rapid development and maintenance of Physical Network structures. TRANSNET allows the building and maintenance of various modes (types) of physical networks. These include Transportation networks such as Rail, Highway, Inland Waterway, and Ports (not currently implemented); and Energy networks such as Pipelines for petroleum or gas, Electrical Power Grids; and other networks such as Physical Communications and Manufacturing (Industrial Networks).

In cooperation with the MIDB program, we participated in redesign of the Networks data structures to enable a common structure to be used for both Physical and Functional Networks.

An initial data base load of Rail and Inland Waterway data was accomplished using spreadsheet data supplied from the UK. This data was delivered with the prototype software.

The ability to build, store, retrieve and maintain multiple data templates was provided. Templates contain sets of "default data" which can be selected and used to supply common data to the building of multiple records, such as the track gauge and normal operating speed for railroad links in a given area.

A set of standard query windows were developed to retrieve current Network records. Options to display these results in tabular format or to display them on a map were provided. Repetitive queries may be executed to build up a working set of retrieved data. Working sets may be stored, retrieved, or modified. They can be retrieved as an initial starting point or saved for future reference.

Using the CMTK map display, we added TRANSNET functions to allow rapid point and click building and maintenance of links, nodes and networks. The use of template data to be applied to new records speeded up the process and eliminated repetitive data entry on the part of the analyst.

Integration of TRANSNET maintenance functions with the MIDB Production capability and database was accomplished to provide full compliance with MIDB integrity and production rules. These rules are invoked whenever an attempt is made to add or update records to the database. MIDB records for facilities, equipment, and units may be associated to the networks as nodes.

Additional analyst support functions were added to the maintenance windows to allow calculation of capacities, transit times, distances, designation of shortest path, etc. Simple connectivity checks are also provided. Functions for combining or splitting links may be initiated using data selection methods and menu items on the map display. Additional functions for calculation of rate of advance have also been implemented.

TRANSNET Demonstrations:

The following demonstrations were conducted in support of the TRANSNET effort:

- Prototype demonstration and requirements/design working session with DIA RAI-1A, USTRANSCOM, AFRL, and MOD UK representatives at PRC, Fairview Heights
- Prototype demonstration and requirements/design working session at PRC, Fairview Heights

- Prototype demonstration and requirements/design working session at Scott AFB
- Installed software and presented a demonstration of TRANSNET capabilities at DIA
- Demonstrated the TRANSNET software to AFRL and AFIWC personnel at PRC Bellevue. Reviewed known shortfalls for inclusion in our list of recommendations.

Documentation and Support:

In conjunction with the TRANSNET software development, the following additional services were provided:

- Initial data for Rail and Inland Waterway loaded into the MIDB data structure
- Data Load Software and Runtime Instructions
- Draft Users Manual
- Installation/Integration Instructions
- Briefing on TRANSNET capabilities in conjunction with the MIDB Users Group (MUG) at DIA
- Telephone and informal documentation support to the MIDB development contractor.
- Documented the design approach and installation requirements and a list of "issues" or future requirements/recommendations which should be considered later on in the integration/enhancement period for TRANSNET. These were transmitted to the MIDB PMO as draft CRs.

Recommended CRs:

The following recommendations were made regarding future changes or current shortfalls to TRANSNET:

- Upgrade to Sybase Version 11.3.
  This is intended to prevent lockups encountered in Route creation software encountered during 8 October demo. These could not be duplicated at PRC Bellevue facility.
- Template Maintenance should be simplified.
  This should take the form of a maintenance screen image with the ability to "fill in" the blanks with desired default values.
- Integrate TRANSNET with MIDB query engine.
  Include geographic searches.
  Allow results lists to be added to the TRANSNET Working Set.
- Integrate TRANSNET with MIDB to include
  TRANSNET application launch.
  Print capability.
- Error notices are not helpful: Sybase returns error message - Displayed error message identifies database element names and does not identify current screen display field when data is missing or in error.

This will require either revision of the schema to incorporate the necessary fields within the Link table or the interception by the application of the raised data base error and a correlation with the display field name before putting out an error message.

- HELP: Add Help capabilities for functions as well as integrating valid values.
- HELP function on clearance or trait records varies by mode.
  This will require either revision of the schema to incorporate the necessary fields within the Link table or a change to the Help function to specifically correlate the clearance or trace element to the mode-specific context.
- Routes and Segment IDs - TRANSNET has not implemented sub-links which would allow redefinition of link records.
  The requirements for this should be analyzed for the various modes and implemented as necessary. The AKA feature provides only a limited capability.
- Cancel Query:
  Cancel Query should be implemented for long-running queries for the TRANSNET Query function. Currently, the analyst must be careful to limit his search and not ask for a full data base search which could take a very long time.
- Net Node maintenance does not allow Assocs.
  Currently, the only way to include Assocs is by first changing node from a generic Node to a specific Facility, Unit, or Equip.
  Requirements should be analyzed to determine if this generic Node capability is sufficient. The initial Node maintenance implementation was intended only to allow rapid construction of networks with the capability to refine the node definitions later by replacing them with more specific references.
- Need to establish production criteria and prodperms for Templates.
- PAL criteria for Networks need to be established.
- Duplicate Templates may be created. Need to review indexes used for Network-related entities to preclude this.
- SPP QC should be re-examined to address the new network entities.
  Updates of networks may need to be keyed to Net Link Type.
- Max density - Add to database for 3.0
- Move data from Clearance and Trait tables to Link table where the data is appropriately part of the link, e.g. Motive Power - move from trait to Link (rail).
  This will greatly facilitate the solutions to the database error notice and HELP data problems among others.
- WWY data - move data from clearance to FAC.
  This data was accommodated in the Network data structure as an initial solution to load the British inland waterway data.
- Map Color codes and symbology enhancements.
  RAI has requested numerous enhancements for the graphical representation of links to include such things as symbology to discriminate among various types of motive power, dual/single track, colors to indicate capacity, etc. This capability should be extended to link types other than rail. This involves requirements coordination with the CMTK replacement effort.

- Provide capability to save/initialize default template to/from user account.
- Display both the calculated and a maximum density for links where appropriate.
- Re-examine Net maintenance and implement more convenient method to add additional Nodes or Links to the Net record.

  Currently the plot function for a net plots only the Links. Nodes must be queried for, plotted, and selected separately. Additional links must also be added separately to the working set, plotted and selected to appear in the candidate set on the Net Maintenance screen.
- Shortest Path calculation (time):

  This does not currently take into account the Link Type and calculates only the distance/speed.

  Note: Transit time, which is stored in the Rail clearance record, is not used.

  Node impedance is not used in the calculation.
- Length should be mandatory in Link.

  Length is required for many calculations. Shortest path will not consider a link with null length. Zero length will be treated as a legal value. Default lengths could be computed based on straight-line or great circle distances.

Final Software Delivery:

The final TRANSNET prototype demonstration software was delivered via tapes sent to the following addresses on 1 Dec, 1997

DIA/RAI-1A
Attn: Mr. John Donaldson
Bolling AFB, Bldg 6000
Washington D. C. 20340-5100

AFIWC/DBIF
Attn: Mr. Barry Stevesoon
102 Hall Blvd., Suite 349
San Antonio, Texas 78243-7990

AFRL/IFEB
Attn: Mr. John Frank
75 Electronic Parkway, Bldg 119
Rome, New York 13441-4515

TRW, Inc
Mountain View Operations
Attn: Mr. Ray Covert
430 Ferguson Dr
Building 1
Mountain View, California 94043

USTRANSCOM - JCJ2-J
Attn: Ms Francine Billings
508 Scott Dr., Rm 339
Scott AFB, Illinois 62225

Equipment Disposition:

Hardware and COTS software and documentation acquired during the conduct of this effort was delivered and turned over to DIA/RAI and to USTRANSCOM - JCJ2-J.

Disposition of hardware to the DIA RAI Transportation Analysts and to the ENERGYNET Analysts was made following the final software demonstrations to allow continuing evaluation and exploitation of the TRANSNET capabilities and to allow formulation of requirements for improvements and extensions to the TRANSNET production functionality.