

AD-A100 847

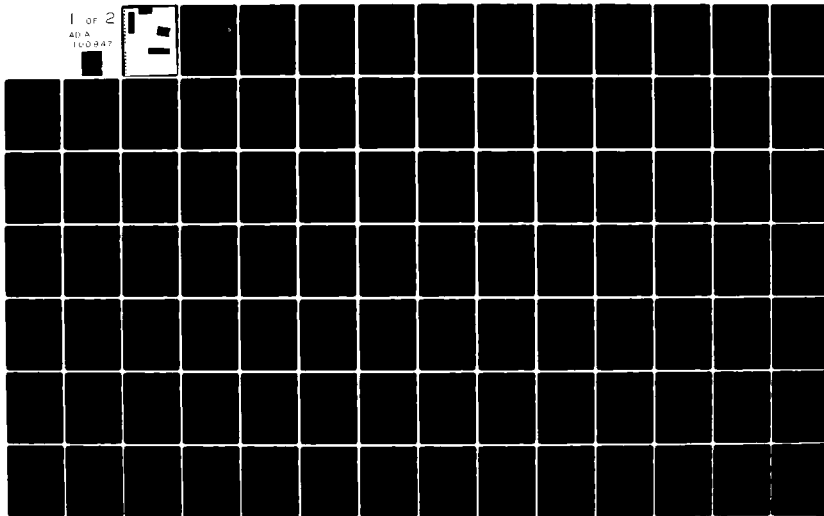
LOGICON INC LEXINGTON MA
AIRMICS LARE/CADSAT ANALYSIS.(U)
APR 81 K M TERRELL, L A JOHNSON
R81003(A)

F/6 9/2

DAHC26-80-C-0021
NL

UNCLASSIFIED

1 OF 2
AD A
100847



LEVEL

AD A100847

JUN 30 1981

C

DISTRIBUTION STATEMENT

Approved for public release
Distribution is unlimited

Report No. R81003(a)

AIRMICS LARE/CADSAT ANALYSIS

Final Report

by

Kim M. Terrell

Larry A. Johnson

30 Apr 1981

Prepared under Contract Number

DAH26-80-C-0021

for

Department of the Army
U.S. Army Computer System Selection and Acquisition Agency
2461 Eisenhower Avenue
Alexandria, VA 22331

18 Hartwell Avenue
Lexington, MA 02173

DISTRIBUTION STATEMENT
Approved for public release;
Distribution Unlimited

LOGICON

PREFACE

This study presents the results of our application of Logicon's Automated Requirements Engineering (LARE/CADSAT) to two existing U.S. Army Computer System Command (USACSC) Detailed Functional System Requirements Documents (DFSR). The effort was performed by Logicon under Contract DAHC26-80-C-0021.

Logicon wishes to acknowledge the support of Major John S. Davis and Mr. John Mitchell of Army Institute for Research Management Information and Computer Science (AIRMICS).

The authors wish to recognize Ms. Karen Favreau for her outstanding technical support on this project. We also thank Mr. Jeffrey Miller for his contributions, Ms. Deborah Queen for overseeing the final product, Ms. Eileen Fitzgibbon for her graphics and Ms. Jill Andersen for her tireless word processing support.

Accession For	
NTIS	GP&C
DTIC TAB	
Unannounced	
Justification	
By	<i>free for the</i>
Distribution/	
Availability Codes	
Dist	Special
<i>A</i>	

TABLE OF CONTENTS

		<u>Page</u>
	PREFACE	ii
	LIST OF FIGURES	v
	LIST OF TABLES	v
Section 1	Introduction	1
1.1	Goals and Objectives	1
1.2	Background	2
1.3	Summary of Conclusions and Recommendations	7
Section 2	Technical Approach	9
2.1	Introduction	9
2.2	Development of Requirements Engineering Plan	9
2.3	Identification of Functional Requirements	12
2.4	Analysis of Functional Requirements	12
2.5	Analysis of Control Requirements	13
2.6	Analysis of Data Requirements	14
2.7	Analysis of Requirements Traceability	15
2.8	Analysis of Requirements Consistency and Completeness	16
2.9	Loading of Requirements Text	18
2.10	Evaluation of Several Existing Requirements Analysis Methodologies	18
Section 3	Results	21
3.1	Introduction	21
3.2	Requirements Engineering Plan	21
3.3	Functional Requirements Identification	23
3.4	Functional Requirements Analysis	26
3.5	Control Requirements Analysis	31
3.6	Data Requirements Analysis	32
3.7	Requirements Traceability	43
3.8	Consistency and Completeness Analysis	45
3.9	Requirements Specification Generation	45
3.10	Evaluation of Alternative Requirements Analysis Methodologies	49
Section 4	Conclusions and Recommendations	61
4.1	Conclusions Specific to This Study	61
4.2	Conclusions Supported by This Study and Prior Experience	62
4.3	Recommendations	64
Appendix A	Requirements Engineering Methodology	65

TABLE OF CONTENTS (Cont'd.)

		<u>Page</u>
Appendix B	A Description of LARE	73
Appendix C	Proposed LARE Enhancements	96
Appendix D	Sample LARE Outputs	104
Appendix E	Bibliography	164

LIST OF FIGURES

		<u>Page</u>
Figure 1-1	Functional Overview of CADSAT	6
Figure 2-1	Task Overview of AIRMICS Study	10
Figure 3-1	Example of MOM Text Translated into LARE	24
3-2	Example of Specific Data Retrieval	25
3-3	Initial High-Level MOM Hierarchy	27
3-4	Revised MOM High-Level Hierarchy	28
3-5	Example Process Structure with Comments	30
3-6	Sample Discrepancy Report No. 1	33
3-7	Example Name List with Discrepancies	35
3-8	Sample Discrepancy Report No. 2	36
3-9	Sample Discrepancy Report No. 3	37
3-10	Sample Discrepancy Report No. 4	39
3-11	Example Consists Comparison Report, Part I	40
	Example Consists Comparison Report, Part II	41
	Example Consists Comparison Report, Part III	42
3-12	Example Requirements Traceability Report	48
3-13	Sample of LARE Generated Text - Format I	50
3-14	Sample of LARE Generated Text - Format II	51
3-15	Schematic Diagram of CADSAT-Defined MIS	52
3-16	REVS Functional Organization	54
3-17	Sample R_NET Structure in RSL and in Graphical Form	55
3-18	REVS Simulate Functional Components	56
3-19	Functional Illustration of IORL	57
3-20	Illustration of IORL Schematic Block Diagram	58

LIST OF TABLES

Table 2-1	Evaluation Criteria	20
Table 3-1	Discrepancy Reports Breakdown	46
3-2	Reports Offering Visibility on Discrepancy Reports	47
3-3	Evaluation Results	60

LOGICON

1. INTRODUCTION

1.1 Goals and Objectives

This study was conducted to determine the applicability of Logicon's Automated Requirements Engineering (LARE) methodology in the Army logistics support environment. Logicon developed LARE to provide a means to define, effectively analyze, and maintain system requirements. LARE includes a coherent set of procedures which allows analysts to define/analyze requirements using computerized tools. The Computer Aided Design and Specification Analysis Tool (CADSAT) was used on this project. CADSAT was designed to aid structured documentation and analysis of large processing systems. This project used the Air Force's standard CADSAT with the CADSAT extensions developed by Logicon.

The study focused on:

- illustrating the methodology
- defining LARE procedures
- identifying enhancements to LARE

Normally LARE/CADSAT is used to produce or validate specifications for a system and its components. Study objectives did not however dwell on reproducing a set of documents. Replication of system documentation would have prevented the effective evaluation and illustration of LARE technology.

LOGICON

Discussion of the technical feasibility of building the system was also excluded from the study. No functional or analytical simulations were performed to assess performance requirements with respect to timing and accuracy constraints. While these subjects are crucial to system design, they are outside the scope of this effort.

1.2 Background

Development of major systems is an increasingly complex problem. This problem manifests itself in varying degrees of successful system implementations but has its roots in differing perspectives. Users, analysts, developers, and managers concentrate on that portion of a system for which they are directly responsible -- a potentially expensive situation. Effective system implementation demands that someone comprehend the total system under development to allow effective management of its progress.

No completely automated way, at this time, can ensure the absolute success of a target system. This study addresses one approach to reducing the difficulties inherent in defining complex systems. LARE centralizes information storage in a set of databases. Using LARE to model existing specifications or develop an initial concept for an operational system will significantly improve the quality of the system and its documentation, thus easing the implementation, testing and operations/maintenance phases of the project.

LARE developed on these precepts:

- valid requirements definition is critical to successful system implementation,

LOGICON

- computerized information storage should be used to maintain all basic data about a system,
- selected retrieval of stored information in formats suitable to users and developers is essential, and
- traceability between levels of system documentation is mandatory.

Defining a valid set of requirements before development work starts minimizes system costs. It is generally recognized that one reason for the high cost of systems development is the delay in detecting errors, inconsistencies and omissions in the requirements/specifications.

LARE prevents some problems and detects others early. LARE language constructs force analysts to be precise. The output reports make errors, omissions and inconsistencies highly visible.

Storing the requirements and specifications in centralized databases facilitates:

- selective retrieval
- up-to-date documentation
- life-cycle maintenance
- standardization of terminology
- incremental information updates

LOGICON

Selective retrieval of centrally-stored information allows individuals on a project to get information pertinent to their jobs. Managers, analysts and programmers require different types of information to do their respective jobs.

Requirements tracing is the process of ensuring that all general requirements in a high-level specification have detailed counterparts in lower-level specification(s).

Every requirement must be traceable through all levels of system documentation to ensure that the system fully meets the user needs -- and that every stated need has been addressed in the allocated requirements and design.

In the last few years, both government and industry have begun to recognize the necessity of doing thorough requirement analyses. Managers responsible for the development of information systems find themselves plagued by poor quality documentation, unstructured requirements and specifications, and the inability to verify the consistency and completeness of system requirements and specifications.

These problems begin in the initial stages of system development and their impact increases in terms of time and cost over the system's life cycle. For example, inconsistent, incomplete requirements foster inconsistent, incomplete specifications, which translate into design errors. These flawed designs are turned over to developers who find them difficult and in some cases impossible to implement.

LOGICON

In 1971, the Problem Statement Language/Problem Statement Analyzer (PSL/PSA) was developed at the University of Michigan (Teichroew, Hershey and Sayani). PSL/PSA is a language for describing system requirements and design. The Air Force acquired a slightly modified version called the User Requirements Language/User Requirements Analyzer (URL/URA). URL/URA is also referred to as the Computer Aided Requirements Analyzer (CARA) and the Computer-Aided Design and Specification Analysis Tool (CADSAT).

All variations of the tool have three main parts and a common problem. The first part is a language to express system requirements. The second is a database system to store the requirements. The third is outputs/reports to depict the requirements. All tool variants lack a documented methodology for application.

Logicon's enhanced CADSAT is the result of an evaluation which found URL/URA lacking in the following areas:

- visibility into the database(s)
- traceability between databases
- documentation acceptable by Military Standards

CADSAT therefore differs from PSL/PSA in its output capabilities (Figure 1-1). CADSAT provides a systematic approach for building information systems, but it soon became apparent that an approach to effectively using it was essential. Logicon's Automated Requirements Engineering (LARE) methodology was developed by applying CADSAT over a period of five years (Applications include programs done for the Air

LOGICON

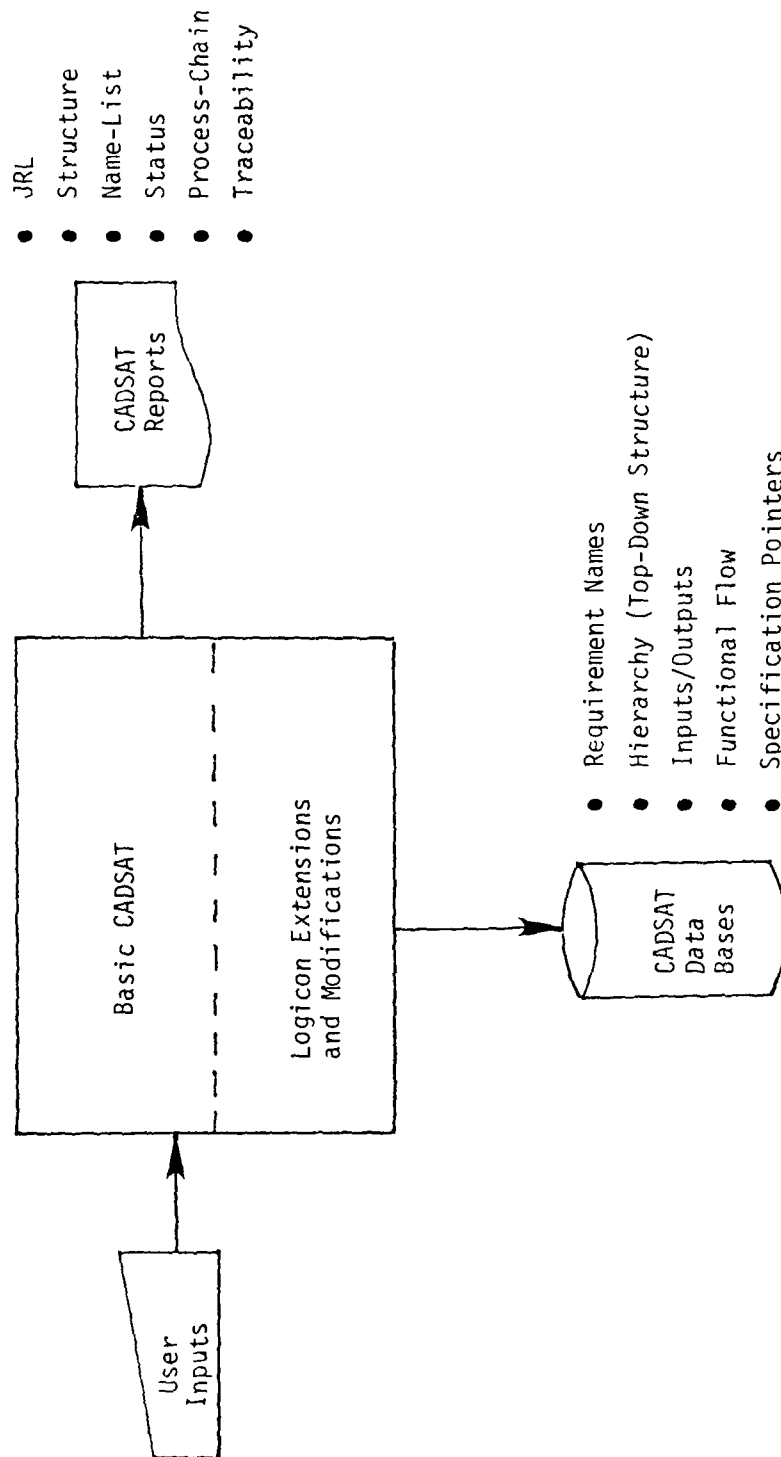


Figure 1-1 Functional Overview of CADSAT

LOGICON

Force, Army, Navy and other defense/Government agencies). Hereafter, the term LARE will refer to both the tool and the methodology used to apply it.

1.3 Summary of Conclusions and Recommendations

Under analysis were the Standard Army Maintenance System (SAMS) - Retail Level, Maintenance Operations Management (SAMS-1/MOM) and the Maintenance Program Operations Management (SAMS-2/MPOM) specifications, hereafter known as the MOM and the MPOM respectively. The MOM contains approximately 2,100 pages. About 85 pages of narrative discuss system functions and constraints. Design discussions and solutions also appear in the narrative. The balance of the MOM is contained in nine annexes which consist of input/output descriptions, flowcharts, decision tables, file descriptions, information elements, external interfaces, telecommunication requirements, and a glossary of terms. MPOM uses the same format and information breakdown, but is considerably smaller.

The study demonstrated that LARE could effectively describe both specifications. Details necessary to describe both systems were translatable into LARE and representable by LARE outputs. As analysts found information they could not justify or information felt to be inconsistent or incomplete, Discrepancy Reports were written. Table 3-2 and 3-3 in the Results Section summarize the types of discrepancies noted and the number of discrepancies formally written up.

LOGICON

Briefly, Logicon recommends the following:

- 1) Simplify LARE's man/machine interface by incorporating user prompts.
- 2) Improve LARE's ability to record and depict conditional control flow.
- 3) Augment LARE's ability to record and depict test requirements.
- 4) Enhance LARE's database management system.
- 5) Apply the enhanced LARE to the specification of an Army system from initial requirements definition through implementation.

LOGICON

2. TECHNICAL APPROACH

2.1 Introduction

This section describes the technical approach used in applying LARE to the example Army Detailed Functional System Requirements (DFSR).

Figure 2-1 shows the basic steps of the technical approach. Each of the steps is described within this section. In addition to describing the procedure, the discussion includes the rationale underlying decisions.

The emphasis of this application has been to demonstrate the applicability of LARE to the Army DFSRs, illustrate LARE'S strengths and weaknesses, and document the experience gained during the study.

2.2 Development of Requirements Engineering Plan

At the beginning of each project an individualized set of appropriate conventions are developed. This enables analysts working on the project to move uniformly toward a common goal. LARE provides a generalized methodology applicable to most system development cycles. Each project requires some tailoring of LARE. Details of the Requirements Engineering Plan developed for this effort appear in Appendix A.

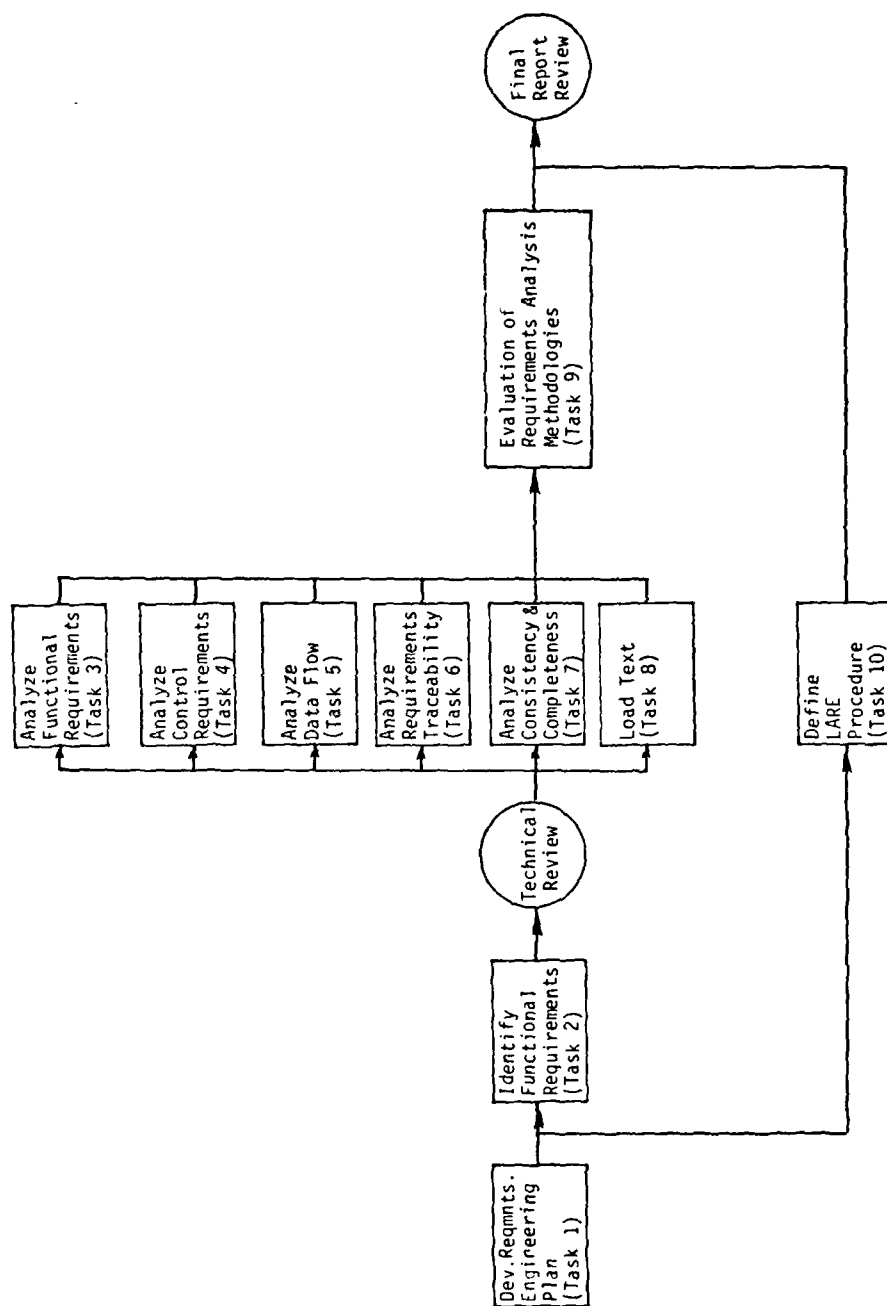


Figure 2-1. Task Overview for AIRMICS Study

LOGICON

The first step of the project was familiarization with the MOM and MPOM specifications. Analysis was isolated from the developmental effort responsible for the specifications. They were therefore represented based on several assumptions:

- Both systems consist of functional requirements
- These functions are connected or interrelated in various ways
- The main purpose of both systems is to process data
- Properties and values impact design

After reading the MOM and MPOM documents, we split tasks into major areas of concentration.

- Develop evaluation criteria. Evaluation criteria were set up to gauge LARE's ability to support requirements engineering (See Table 2-1). The criteria consider only the analysis of existing specifications.
- Refine standard Requirements Engineering Plan. The Requirements Engineering Plan given to analysts in the early stages of this application is included in Annex A. Specifics of the plan pertinent to this project include: generate a synonym algorithm, identify reports to be produced, and select CADSAT language to represent information in the specifications.
- Define and depict functional requirements. Sections 2.4 through 2.9 detail our approach to requirements definition.

LOGICON

- Evaluate alternative technologies. Section 2.10 and 3.10 discuss our brief evaluation of alternative requirements engineering aids.

2.3 Identification of Functional Requirements

Each specification was reviewed and the task of identifying functional requirements started. At this stage, analysts searched the documents paragraph by paragraph extracting and naming what they determined to be functional requirements of the system.

Conventions used for identifying and naming requirements are detailed in the Engineering Plan found in Appendix A, Section A.4.

2.4 Analysis of Functional Requirements

Once the functional requirements of the system had been identified, a high-level requirements hierarchy was built. The first hierarchy emphasized the input, internal, and output processing of the system. This approach was used since the target system is a Management Information System. The three categories, though strongly emphasized did not allow us to adequately decompose the system requirements. Because of this, a second hierarchy was built. The modified structure emphasized the type of data being processed (e.g., process-work-orders, process-task-performance-data, maint-parts-inventory). The input/output processing for each of these areas is represented at a lower level of the system hierarchy.

LOGICON

After finalizing the high-level hierarchy, the detailed functions to be performed were selected and incorporated into the process structure.

2.5 Analysis of Control Requirements

Data and system control flow are closely related. LARE defines control flow as the order in which system functions are sequenced. Data derived by functions early in a sequence often impact the next sequence of events. Data flow (i.e., the flow of information into, through and out of the system) may be interrupted by system functions temporarily suspended and awaiting the proper control sequence. Data inputs often cause spontaneous execution of specific processing.

Primary objectives of control flow analysis are to identify:

- timing relations among target system functions
- functions omitted by previous steps in the order
- missed requirements of a target system

Due to the complementary nature of LARE, control flow analysis may be performed in two ways. The first defines a sequence which initializes processing at the point in the processing when it crosses system boundaries. Possible subsequent sequences of steps or processes are recorded, analyzed and inserted in the database. The resultant chain shows all possible processes that may be performed, including

LOGICON

parallel or mutually exclusive ones. Mutually exclusive processes imply a decision made during performance. A second method of analysis starts at the output and traces back through the internal sequence which caused this end result. This process, like the first, yields parallel and mutually exclusive threads of process control. Identifying and following these threads through the system frequently reveals functions which were omitted or input requirements that had been overlooked.

2.6 Analysis of Data Requirements

The primary objectives of Data Analysis are:

- identify information crossing the external system boundaries
- determine the information deciding control flows
- determine the outputs requiring inputs

At the requirements definition stage of system development, the "real" data requirements are data crossing external system boundaries. These data items must be defined precisely and carefully controlled since they always impact the ability of the target system to interface with other systems.

LOGICON

Identification of information required to determine which internal functions the system will perform is closely tied to the control flow analysis. If this information is not provided in the system, it will not operate.

The third part of data requirements analysis determines the inputs necessary to produce required outputs. All too often, system data requirements are unclear or input/output data are identified as "nice to record" or "nice to know". As a result, requirements definition is vague. This vagueness impedes the development of an operational system. Required outputs should first be justified in terms of their utility to the user or other external system. An output should be specified in terms of frequency and accuracy constraints. Once an output is deemed required, an analysis must be performed to identify information elements necessary to produce the output. The algorithm used to transform inputs to output must also be determined. The final inputs identified must be shown to be available.

2.7 Analysis of Requirements Traceability

Requirements traceability is defined as the process of ensuring that upper-level requirements are allocated in the lower levels of documentation.

It was originally thought that a traceability analysis could be performed between the MPOM and MOM specifications. Closer examination of these specifications showed that they were distinct systems to support different Army Management levels, not different levels of specification of the same system. Although this analysis could not

LOGICON

be performed on the specifications used in this study, a discussion of the methodology is included in this report.

A typical analysis would trace through the following levels of documentation:

- GFSR to DFSR
- DFSR to actual system test requirements
- DFSR to the system design implemented to meet the requirements.

2.8 Analysis of Requirements Consistency and Completeness

The consistency and completeness of requirements has been mentioned throughout the above discussion. No analytical technique exists to assure the completeness of a set of requirements. The LARE approach cuts through the target system requirements from multiple perspectives and provides the analyst with a relatively small number of items to review. The analyst then assesses completeness.

The first type of completeness analysis is hierarchical analysis. If a system has 2,000 requirements defined, it is difficult to determine whether the requirements are complete or whether there should be 2,001 requirements. With hierarchical analysis, each requirement is decomposed into four to seven components. At each step of the decomposition

LOGICON

the question is asked: Does the sum of the sub-requirements completely describe the requirement at the next level up? If the answer is yes, the decomposition of that requirement stops; if no, the missing requirements are identified. This process continues until the analyst feels that the requirement has been accurately broken down. This approach looks superficially similar to that used in generating most requirements documents. Specification formats usually call for a hierarchical paragraph numbering scheme. The primary difference is that LARE requires the strict logical relationship of parts between each paragraph and its subparagraphs.

Next, data must be analyzed for consistency. At the requirements stage, the basic objective is to identify the type of data needed and derived by each process. This identification procedure helps to ensure that data is derived prior to being required. It also helps flag data derived by multiple functions. This data should not be described in detail at the *requirements definition* stage. Defining the detailed layout of internal data is a problem for system design and not for requirements specification.

Consistency analysis is predominately a problem of sorting through multiple statements of individual functional requirements. Consistency of requirements does not manifest itself until all statements are pulled together. LARE aids the analyst in assembling the information and storing it logically. Inconsistencies are not limited to functional requirements. They can appear in constraints (e.g., performance requirements), information content, control flow, data units (e.g., time to repair equipment in hours vs. days of a person's time).

2.9 Loading of requirements text

System specifications are frequently viewed as a necessary evil which must be generated at the beginning of system development only to be forgotten once system construction begins. This should not be the case. Up-to-date system specifications, are relevant throughout the system life cycle. Requirements generally undergo changes during system development. To minimize system life cycle costs, analyze the impact of requirements before developing system updates. Otherwise, implementing new system capabilities or changes will cause other system errors or problems.

LARE greatly enhances the ability to maintain up-to-date system specifications. The effort required to print change pages or entire specifications ready for publication is greatly reduced. In addition, LARE provides a single integrated database representation of the specification. This minimizes divergence of functional and data structures from the requirements specification text representation - likely occurrence if text is maintained manually or in a separate word processor.

2.10 Evaluation of Several Existing Requirements Analysis Methodologies

A cursory review was made of two additional computer-aided requirements analysis tools and methodologies. Input/Output Requirements Language (IORL), and Software Requirements Engineering Methodology (SREM) were chosen because both are being used by the Army and are the focus of independent technological demonstrations similar to this study. A comparative evaluation was made of these technologies and LARE.

LOGICON

Logicon developed LARE and greatly enhanced the computerized tool (CADSAT) used by the methodology. However, Logicon is committed to the evolution of computer-aided requirements analysis and specification generation independently of the specific tools used. The objective of the evaluation was to identify the strengths and weaknesses in capabilities of tools to represent system documentation. The evaluation Criteria are shown in Table 2-1.

Table 2-1. Evaluation Criteria

Ease of:

- database modification
- report generation
- report readability

Ability to:

- record/depict functional requirements
- record/depict constraint requirements
- record/depict control flow
- record/depict data flow
- record/depict functional hierarchies
- record/depict data hierarchies
- record/depict interfaces
- record/depict external documents
- record/depict test requirements
- record/depict project status

Ability to:

- aid in detection of incomplete data
- aid in detection of inconsistent data
- aid impact analysis

Ability to:

- generate system specifications
- generate test specifications
- generate design documentation

Ability to:

- simulate (general)
- trace requirements
- trace between levels of documentation
- provide diagnostics
- provide well defined methodology

LOGICON

3. RESULTS

3.1 Introduction

The results section presents the findings in the same order as the technical steps described in the previous section. Tables and report illustrations are included in this section. Longer example reports have been included in Appendix D. Reports produced by LARE, except the specification report, are analytical aids. These reports constitute a snapshot of the database.

3.2 Requirements Engineering Plan

The Requirements Engineering Plan (see Appendix A) developed for this project includes all the basic elements used by project personnel. It is however, simpler than one for a project which uses LARE as the primary tool to develop a complete system specification. A project using more people over a longer interval would include extensive discussion on requirements traceability, use more LARE reports, and discuss the tracking of requirements changes/discrepancies at length.

One key element of the plan is the identification of system boundaries. The project was provided the MOM and the MPOM specifications. These specifications detail a set of requirements for two systems, or possibly two subsystems, of a much larger system, the complete SAMS (wholesale, retail and all levels of user/management). Each of the

LOGICON

specifications refer to data processing functions (both manual and computerized), maintenance inspections, actual maintenance/repair, and maintenance planning.

It was decided to treat the MOM and MPOM as separate systems for analysis. Thus, communication between these two systems becomes external. Furthermore, the system boundary was defined to include only manual and automatic data processing functions.

Manual data processing functions were included, although they were not as well defined as the automatic functions, because the line of demarcation between manual and automatic processing functions might change in the future. Also, including manual functions enables clearer documentation of the requirements,

Discussions of maintenance, inspection, or maintenance system planning are considered descriptive (in terms of our definition of system boundaries) and are not addressed as functional requirements. This is not to imply that this material should be removed from the document; it frequently helps the reader understand the actual functional requirements. Another implication of our system boundary definition -- personnel operating the eventual system may be part of the system or an external interface. Operators become part of the system whenever they perform manual data processing functions; e.g., developing a list of non-operational equipment by going through a file containing equipment status for each item. Physically locating each piece of equipment and inspecting it to determine operational readiness is not a data processing function. The operator performing this task is not directly related to the system as defined. Entering the results of an inspection into a box on an inspection sheet, onto a form for keypunching, or directly keying the results into a computer terminal, all constitute the

LOGICON

"input" of data into the "system" from an external "interface". In this case, the operator is the external "interface".

3.3 Functional Requirements Identification

Functional requirements for both systems were broken down into discrete requirements. As with all specifications for which LARE has been applied, the requirements were found to be scattered throughout the documentation. Most requirements were identified from the text portion of the specifications. Analysis of appendices primarily identifies data inputs/outputs, i.e., data from or to external sources or external users.

In addition to functional requirements, other requirements were identified which do not "do" something but say something about how or when things are done. These requirements are called constraints, and were defined mainly in the MPOM database.

After identifying a functional requirement a descriptive LARE name was assigned, a synonym was defined, and a paragraph reference linked to the analyst assigned name. Figure 3-1 is an example of how functional requirements are extracted from paragraphs in the MOM specification and translated into LARE terminology. The figure also shows how the same function is repeated in several paragraphs. Figure 3-2 was generated by prompting the database for all names associated with paragraphs m-4.10.g, m-4.10.q, and m-4.10.x.5 (MOM). One of LARE's greatest assets is that it stores information once but allows that information to be accessed in numerous ways.

Process structure

parameters for: str

process indent=3 noindex

count level name

reference

```

1 4 enter-standard-vo-data
2 5 assign-equip-recall-reqmnt-vo
3 5 assign-intra-shop-vo-numbers
4 5 enter-assigned-work-order-number
5 5 enter-work-order-task-data
6 5 enter-vo-evacuation-data
7 5 process-maintenance-request
8 5 register-intra-shop-work-order
9 5 enter-work-order-parts-data
10 6 enter-vo-supplemental-parts-data
11 6 enter-wo-parts-reassignments
12 7 re-assign-vo-parts
13 6 xm-co-decbl-input-datacheck
14 6 xm-cs-decbl-input-datacheck
15 6 process-edit-subprocess
16 5 enter-registration-wo-data
17 6 xm-a-decbl-input-datacheck
18 6 xm-b-decbl-input-datacheck
19 6 process-vo-registration-data
20 5 enter-vo-requirements-data
21 6 process-vo-requirements-tasks

```

INTERFACE OR PROCESSOR

4-10g. The Shop Office Clerk keys for entering Task and MHR data entry (Document Identifier Code XMC(T)). When prompted by the processor, the data for the Task Requirements (12 03 KZ) are entered. Preparations are then made to enter the WO status into the WOPF.

4-10X(5). The Shop Office Clerk keys for entry of task and man-hour data (Document Identifier Code XMC(T) (12 03 KZ)) (ref para 4-10g).

4-10q. The Shop Office Clerk keys for Task Completion Data entry (Document Identifier Code XMD(T)). (If man-hour accounting is in effect, a Document Identifier Code XMD(L) will be used.) When prompted by the processor, the Task Completion Data (12 04 KZ) are entered.

level	count	level	count	level	count	level	count
1	0	2	0	3	0	4	1
6	9	7	1			5	10

Figure 3-1. Example of MOM Text Translated into LARE

parameters for: tps

formatted problem statement

```
file noindex print empty nopunch smarg=5 nmarg=20 amarg=10 bmarg=25 rnmarq=70 cmarg=1 hmarq=40
nodelign one-per-line define comment nonew-pade nonew-line noall-statements
complementary-statements line-numbers printend
```

```
1 define m=4.10.g
2 as a source;
3 applies to: enter-work-order-task-data,
4 enter-work-order-time-expanded,
5 enter-wo-completion-data;
6
7 define m=4.10.d
8 as a source;
9 applies to: documentation-identifier-code,
10 manhour-accounting-regent-opt,
11 enter-wo-completion-data,
12 enter-work-order-task-data,
13 enter-work-order-time-expanded;
14
15 define m=4.10.x.5
16 as a source;
17 applies to: enter-work-order-task-data,
18 enter-work-order-time-expanded;
19
20 eof eof eof eof
```

Figure 3-2. Example of Specific Data Retrieval

3.4 Functional Requirements Analysis

The primary objective of the functional requirements analysis is to take the discrete functions identified in the previous step and tie them into a consistent, understandable functional hierarchy. The hierarchical structures of both systems were made as similar as possible. These structures include much of the organization already contained in the specifications.

Functional requirements from the MOM were loaded into a LARE computerized database first. Management information systems, including maintenance information systems, are frequently organized around the concept of input, processing, and output. The initial high level MOM hierarchy is shown in Figure 3-3. This structure does not lend itself to an easy representation of the lower functional structure. One difficulty is that minimal internal processing is required. For the most part, data comes into the system, is stored, and is reprinted in various formats.

A second hierarchy, developed for the MOM specification, was organized closely to the original specification (see Figure 3-4 for the top-level hierarchy). In this structure the input, process, output sequence repeats within each major area. This structure handles the lower level functions more easily than the original. The structure requires continued analysis in order to meet the objective in the engineering plan: each functional requirement to be decomposed should yield 4 to 7 subfunctions.

Once the MOM high level hierarchy had been built, the MPOM functional requirements were organized into a structure which retained as much

PROCESS STRUCTURE

Parameters for: str

Process indent=1 noIndex

Count level name

```

1 1 sams-retail-won-system
2 2 input-processing
3 3 annotate-work-order
4 3 annotate-repair-parts-form
5 3 annotate-shon-supply-list
6 3 annotate-proper-forms
7 3 inspection-w-o-status-change
8 3 enter-pbf-change-data
9 3 enter-class-inspectn-form-data
10 3 enter-equipment-usage-data
11 3 record-received-parts
12 3 enter-maintenance-request
13 3 enter-shor-supply-data
14 3 determine-required-calibration
15 3 determine-required-maint-service
16 3 enter-work-order-data
17 2 internal-data-processing
18 3 request-v-o-redistribution-data
19 3 record-parts-status-on-ord
20 3 transfer-data-to-tnf
21 3 data-retrieval
22 3 work-order-update
23 3 transfer-w-o-data-computer-media
24 3 work-order-closeout
25 3 sort-repair-parts-forms-tnf
26 3 post-cancel-reconcil-resp-media
27 3 float-processing
28 3 process-lookup-table-data
29 3 parts-tracking-handling
30 2 output-processing
31 3 send-w-o-with-equipment
32 3 distribute-tnf-data
33 3 enter-personnel-data-changes
34 3 report-generation
35 3 generate-w-o-r-transfer-file
36 3 transfer-list-work-ords-data
37 3 output-magnetic-transfer-media

```

level count	level count	level count	level count	level count
1	1	2	3	3

Figure 3-3. Initial High-Level MOM Hierarchy

process structure

parameters for: str

process indent=3 noindex

count level name

1	1	sams-retail-non-svstem	35	3	record-parts-status-on-rpt
2	2	produce-support-plan	36	3	record-received-parts
3	3	maint-support-plan	37	3	generate-monthly-report
4	2	process-work-orders	38	3	print-error-exceptn-report
5	3	enter-initial-wo-data	39	3	generate-eom-transfer-file
6	3	reconcile-wo-parts	40	3	transmit-accptd-work-orders
7	3	enter-wo-update-data	41	3	transmit-tech-labor-rpt-form
8	3	retrieve-wo-data	42	3	parts-tracking-handling
9	3	report-wo-status	43	3	sort-repair-parts-forms-lpd
10	3	close-out-wo	44	3	transmit-manual-acquisition
11	3	process-lookup-table-data	45	3	transmit-wo-rep-parts-form
12	3	work-order-transfer	46	3	gen-parts-awaiting-diso-action
13	3	generate-work-order-data	47	3	gen-parts-status-detail-list
14	3	edit-transactions	48	3	gen-shoostock-zero-balance-rep
15	3	enter-parameter-data	49	3	gen-supply-activity-req-list
16	3	update-internal-wo-files	50	3	xm-tab-decchl-input-database
17	2	process-task-performance-data	51	2	maint-equipment-data
18	3	sample-task-performance-data	52	3	record-equip-inspec-data
19	3	report-task-performance	53	3	determine-required-maint
20	3	distribute-tot-data	54	3	process-equip-recall
21	3	enter-task-net-factor-adj	55	3	process-equip-evacuation
22	3	task-net-fact-adj-supp	56	3	control-float
23	2	maint-parts-inventory	57	3	enter-equipment-usage-data
24	3	inter-ent-annot-shop-sup-list	58	3	print-equipment-usage-data
25	3	annotate-repair-parts-form	59	3	process-equipment-usage-data
26	3	enter-shop-supply-data	60	3	proc-tbl-build-ecc-sta-sto-ing
27	3	print-part-number-mismatch-lst	61	2	maint-personnel-data
28	3	print-reconciltn-exceptn-report	62	3	enter-personnel-data-changes
29	3	print-shon-stock-list	63	3	print-work-center-summary
30	3	print-ssl-constrnt-list	64	3	outout-magnetic-transfer-media
31	3	print-ssl-locator-lst	65	3	print-labor-util-summary
32	3	print-subply-activity-reqmnts	66	3	print-workload-status-agg-rpt
33	3	print-subply-reports	67	3	change-personnel-assignments
34	3	print-zero-balance-stock-list	68	3	change-toe-tde-authorization
			69	3	gen-labor-utilization-summary
			70	3	generate-work-center-summary
			71	3	updt-labor-utilization-detail
			72	3	xm-l-decchl-inout-database
			73	3	process-wrk-ctr-lbr-ext-subr

Figure 3-4. Revised MOM High-Level Hierarchy

LOGICON

MOM structure and nomenclature as was consistent with the MPOM system requirements. Retaining this similarity improves the ability to describe inter-system communications (MOM to MPOM). Items found in the MOM but not in the MPOM are included and tagged with a "can't justify" memo.

The functional requirements hierarchy is the "backbone" of the LARE computerized databases for both requirement sets. It provides a concise view of the system functions and indicates groups of related functions. It aids communication between users and analysts. If a user's job is to manually process work orders and the analyst's job is to automate portions of the manual process, the user must communicate his/her job operations to the analysts. The analyst, after obtaining information about the process, develops a structure that reflects his/her understanding, then uses that structure to interact with the user. A user looking at the analyst's structure can readily see the misunderstandings or omissions. Figure 3-5 is a marked up structure report that resulted from a discussion between project personnel.

The functional structure is also useful for tying related requirements together. The MOM and MPOM specifications (like all specifications) spread related requirements throughout the document. Paging through a document manually to establish relationships between items scattered throughout multiple chapters and documents is time consuming and error prone. Maintaining these relationships manually in a dynamic environment is often overwhelming.

The LARE functional hierarchies of the MOM and MPOM (see Appendix D) represent our understanding of the two systems at a high level. The structure also assists communication among analysts on the project. Further it is the "skeleton" on which the system is built.

process structure

parameters for: str

process indent=3 noindex

count level name

```

1 3 enter-initial-wo-data
2 4 enter-standard-wo-data
3 5 assign-equip-recall-reqmnt-won
4 5 assign-intra-shop-wo-numbers
5 5 ent-assigned-work-order-number
6 5 enter-work-order-task-data
7 5 enter-wo-evacuation-data
8 5 process-maintenance-request
9 5 register-intra-shop-work-order
10 5 enter-work-order-parts-data
11 6 enter-wo-supplmntl-parts-data
12 6 enter-wo-parts-reassignments
13 7 re-assign-wo-parts
14 6 xm-co-dectbl-input-datcheck
15 6 xm-cs-dectbl-input-datcheck
16 6 process-edit-subprocess
17 5 enter-registration-wo-data
18 6 xm-a-dectbl-input-datcheck
19 6 xm-b-dectbl-input-datcheck
20 6 process-wo-registration-data
21 5 enter-wo-requirements-data
22 6 process-wo-requirements-tasks
23 4 enter-production-wo-data
24 5 ent-prod-program-mgmt-detrn
25 5 establish-maint-production-prg
26 5 xm-g-dectbl-input-datcheck
27 5 process-maint-prog-dat-inp-sub
28 4 enter-alt-sro-data

```

Where is information about
consumption data?

Do you want to set up
a separate category
with for consumption data?

What about Labor and Parts?
where are the device checks?

Figure 3-5. Example Process Structure with Comments

LOGICON

The LARE functional hierarchy remains the "backbone" and concisely states requirements (baseline) throughout the system development cycle.

The following analyses are based on the terminology developed in these hierarchies. An advantage of the computerized databases is that hierarchy can be readily changed to include results of further analysis. The systems analyst's reports or working papers can be reprinted to readily reflect any modifications.

3.5 Control Requirements Analysis

MPOM control flows were analyzed and represented in the LARE computerized database. The objective was to determine the sequence of functions in order to show functional consistency and to identify missing requirements. In a completely consistent system every functional requirement appears as a step in at least one control sequence, and every control sequence has at least one cause.

Control flow analysis of MPOM proved difficult due to numerous missing requirements. These gaps left the LARE representation incomplete. Given more time and access to users to clarify requirements, it would be possible to build a complete, consistent MPOM control flow.

We identified two major types of missing requirements: control sequence activation and system initialization. Many control sequences can be constructed, but the specification omits requirements or fails to state clearly how to start the processing. Potentially, processing

LOGICON

could be started by particular messages being input, by certain fields being present in input messages, by clock time, or by time relative to other processing.

All software needs procedures to load data and start operation. Information management systems like MPOM also need procedures either to build the initial database or to start operations using a database built externally. The MPOM specification lacks requirements addressing these subjects.

In some cases it was not possible to follow the control sequence through the system to an output. Conversely there were instances of outputs not tracing back to an input sequence. An example of the latter case appears in Annex H, Table 406 of the MOM. (See Figure 3-6) Sequence 2 prompts for "CRD-DSG." This element is not defined as an input to the Equipment Recall process (See Figure 3-6)

3.6 Data Requirements Analysis

MOM and MPOM data flows were analyzed and represented in LARE computerized databases. All data identified in the MPOM specification was named, structurally related to other data where appropriate, and tied to the computational processes. The MOM data structure is simpler, concentrating on the nature of the inputs and outputs. Our objectives were to identify redundancy, inconsistency and incompleteness in the data specification. In a completely consistent system every input datum should contribute to at least one output, and conversely, every output of the system should be derivable.

Originator: KJF
Verified by: KMT
Sent to AIRMICS:

DISCREPANCY REPORT

Source	Type of Discrepancy	Remarks
MOM ANNEX H DECISION TABLE 406 PG H-215 SEQUENCE 2	INCONSISTENT DATA	SEQUENCE 2 PROMPTS FOR 'CRD-DSG'. THIS ELEMENT IS NOT DEFINED AS AN INPUT TO THE EQUIPMENT RECALL PROCESS. (I2.30.KY) SHOULD IT BE 'CARD-DSG-CD-SAMS' (XME "B")?

Figure 3-6. Sample Discrepancy Report
No. 1

LOGICON

Inputs and outputs for both the MOM and MPOM are detailed in Annexes. After loading data into the databases in a specific sequence, the Name-List Report was used to flag inconsistent nomenclature. Figure 3-7 shows actual listing in which inconsistent nomenclature between input and output elements surfaced. Another example in Annex B has an output (03 22 1Y) element referenced as P-WON. The corresponding input element defined in Annex A (I3 11 40) is P-WON-ORG. (See Figure 3-8). This type of inconsistency makes it difficult to identify instances of data 'used' but not "derived". Although inconsistent naming functions are thought of as clerical mistakes, they create nightmarish debugging problems if an output module expects P-WON and the input module defines P-WON-ORG.

In attempting to tie internally defined data with the data defined as inputs and outputs, the discrepancies multiplied. Internal data are defined by the specifications, especially in the decision tables. In a great many cases, the internal data could not be associated with the inputs or outputs by name. For example, Annex H of the MOM Table Number 1368, sequences 2, 3 and 6 prompt for DIC-SUP-ACT; we were unable to locate an input so defined. (See Figure 3-9). Furthermore, the specifications did not identify specific processing steps that would have allowed us to link the data or show that multiple naming had occurred.

When the specifications identified data characteristics (e.g. legal values) these were recorded in the LARE databases as attributes. When a piece of data had attributes already entered, LARE would flag attempts to enter new attributes of the same type. From this the analysts knew to check for consistency. Inconsistency does not automatically imply incorrect data, it must be checked by analysts. For example, MPOM input element END-ITEM-COMP-IND-FLD (Annex A, I3 01 8W FLD 3) is defined as alphanumeric. The only

name list					
name	type	synonym	paragraph		
525 odt-and-shipping-time-manater	element	ost-cmpt	m-b.02.41.4v.fld.11		
526 old-part-number-field	element	oranshti ost-ngr old-prt-no-fld olpanofi	m-b.02.41.4v.fld.22 m-b.02.39.4m.fld.24 m-b.02.41.4v.fld.10 m-b.02.34.4v.fld.6 m-b.02.34.4v.fld.6		
527 olpanofi	*** undefined ***				
528 on-hand-qnty-ort	*** undefined ***				
529 on-hand-qnty-rep	*** undefined ***				
530 on-hand-quant-ope-read-float	element	onhand-qnty-ort onhaduore onhaduorefl onhadu	m-a.12.40.kv.fld.8 m-a.12.40.kv.fld.8 m-b.02.10.4v.fld.7 m-b.02.09.4w.fld.17 m-b.02.08.4m.fld.12 m-a.12.06.kv.fld.6 m-a.12.07.8m.fld.11		
531 on-hand-quant-ope-read-flt	element	onhand-qnty-rep	m-a.12.17.kv.fld.20		
532 on-hand-quantity	element	onhaduore	m-a.12.17.kv.fld.20		
533 on-hand-quantity-for-repair	element	onhaduore	m-b.02.30.4w.fld.5 m-a.12.98.fld.4 m-a.12.98.fld.4		
534 on-hand-quantity-repair-parts	element	onhaduore			
535 one-ort-days-orts-stat-det	element	onpadapast			
536 one-ort-days-orts-status-tetail	element	parm-da-ort-sta-one			
537 one-param-days-wo-age	element	onpadawoag	m-a.12.98.kv.fld.5		
538 one-param-days-field-wo	element	onpadafwo	m-a.12.98.kv.fld.5		
539 one-param-days-range-fld-wo-age	element	onpadarafi	m-b.02.42.4v.fld.29 m-b.02.33.4v.fld.6 m-b.02.41.4v.fld.26 m-b.02.39.4m.fld.13 m-b.02.42.4v.fld.27 m-b.02.42.4v.fld.25 m-b.02.42.4v.fld.23 m-b.02.42.4v.fld.21		
540 onhand-qnty-rep-part	element	onpadarafi			
541 onhand-quantity-ort	element	onquor			
542 on-readiness-float-item-desc	entity	opreflitde	m-4.13		
543 open-supply-transactns-report	output	opsuttre	m-b.02.36.4w		
544 open-work-orders-report	output	opwoorre	m-b.02.06.4w		
545 optional	attribute-value	oot			
546 ord-date-stat-chg	*** undefined ***				

Figure 3-7. Example Name-List

Originator: JAM
Verified by: KJF
Sent to AIRMICS:

DISCREPANCY REPORT

Source	Type of Discrepancy	Remarks
MPOM	INCONSISTENT DATA	
03.23.1Y XMJ ANNEX B PG B-59 13.11.4D XMJ INPUT PG A-49	FIELD #2 IS OUTPUT AS P-WON BUT INPUT AS P-WON-ORG.	ARE P-WON-REFERENCES SYNONYMOUS?

Figure 3-8. Sample Discrepancy Report
No. 2

Originator: KMT
Verified by:
Sent to AIRMICS:

DISCREPANCY REPORT

Source	Type of Discrepancy	Remarks
MOM ANNEX H TABLE NO. 1368 PG H-640 SEQUENCES NO. 2,3,6	QUESTION	WHY IS THE "DIC-SUP-ACT" BEING PROMPTED? WE WERE UNABLE TO LOCATE THAT ELEMENT. SHOULD THIS BE "SUPPLY SUPPORT ACTIVITY NUMBER" (SUP-SPT-ACT-NO) M-0010-01?

Figure 3-9. Sample Discrepancy Report
No. 3

LOGICON

legal values are alphas. This is inconsistent but not necessarily illegal as far as the system is concerned. (See Figure 3-10).

Data analysis identified other problems worthy of attention. One is redundancy. Instances of duplication were found where different users defined the same input in several ways. Other data structures which appeared dissimilar on casual inspection were shown by LARE reports (Consist/Compare Report) to consist of identical components. (See Figures 3-11). Inputs and outputs that appear similar could be identical if data components with slightly different names are in fact identical. Reducing redundancy cuts storage and processing costs, and increases target system efficiency.

The Consists Comparison Report provides visibility into the impact of certain changes. For instance changing the "part-number-field" from a numeric to an alphanumeric would impact, at a minimum, the circled items on the matrix report. (See Figures 3-11).

Another problem is the specification of design approaches - not requirements - which imply inefficient when implementation. For example, the Supply Status File for MPOM is sorted on each update and sorted again for each use. Sorting is a time consuming process which should be minimized to increase processor availability. This may be accomplished in the example case by carefully choosing the occasion for sorting and perhaps by trading off some storage in which to save secondary sort keys. Further, sorting of new updates, followed by a merge with previous data would reduce processing time.

Originator: JAM
Verified by: KJF
Sent to AIRMICS:

DISCREPANCY REPORT

Source	Type of Discrepancy	Remarks
MPOM ANNEX A I3.Ø1.8W.WOR FLD 3 PG A-14	INCONSISTENT DATA CONFLICTING FIELD TYPE AND VALUES FOR END-ITEM-COMP-IND-FLD.	FIELD TYPE IS ALPHA- NUMERIC, BUT THE ONLY LEGAL VALUES ARE ALPHA (E OR C). SUGGEST FIELD TYPE OF "ALPHA" FOR THIS FIELD.

Figure 3-10. Sample Discrepancy Report
No. 4

consists comparison report

basic contents matrix

the rows are the given input names.

the columns are the lowest level objects which are contained in the rows, with intermediate groups ignored.

if any columns are group names, then the definition is incomplete.

if any columns are ambiguous names, they are possible elements.

row names	column names
alt-sto-requirements	1 unit-id-code-support-unit element
1 bench-stock-data-record-xm-pd	2 unit-id-code-customer-unit element
2 bench-stock-data-record-xm-pe	3 modification-number-field element
3 bench-stock-data-record-xm-pf	4 modification-wrk-ord-ord-prio-cod element
4 bench-stock-data-record-xm-pq	5 ltno1tno1 element
5 bench-stock-data-record-xm-pq	6 ltno1tno1 element
6 calibration-reg-by-customer	7 part-number-field element
7 calibration-required-by-item	8 equip-serial-lcl-con-no-fld element
8 calibration-work-order	9 registration-serial-number element
9 cross-reference-transaction	10 recall-avail-date-ordinal element
10 coredelta	11 procurement-reg-ord-number element
11 equipment-recall-new-item-a	12 documentation-identifier-code element
12 equipment-recall-new-item-b	13 file-input-action-code element
13 equipment-recall-require	14 card-designator-code-sams element
14 first-file-adjustment	15 activity-address-code element
15 maint-prog-data-record	16 funds-available-designator element
16 maint-prog-reqmts-data-record	17 issue-priority-designator element
17 param-workload-backlog-age	18 account-processing-field element
18 parameter-duty-hours-rec	19 project-code element
19 parameter-norms-norm-rec	20 condition-desig-conus-location element
20 parameter-parts-status-detail	21 identifying-number-code element
21 parameter-previous-cycle-rec	22 item-noun element
22 parameter-report-control	23 routing-identifier-code element
23 part-number-change-data-rec	24 signal-code element
24 parts-receipt-data-record	

Figure 3-11. Example Consists Comparison Report - Part I

consists comparison report

basic contents matrix

	1	1111111112	2222222223	3333333334	4444444445	5555555556	6666666667	7777777778	8888888889	9999999990
1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1	1	1
9	1	1	1	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1	1	1	1
11	1	1	1	1	1	1	1	1	1	1
12	1	1	1	1	1	1	1	1	1	1
13	1	1	1	1	1	1	1	1	1	1
14	1	1	1	1	1	1	1	1	1	1
15	1	1	1	1	1	1	1	1	1	1
16	1	1	1	1	1	1	1	1	1	1
17	1	1	1	1	1	1	1	1	1	1
18	1	1	1	1	1	1	1	1	1	1
19	1	1	1	1	1	1	1	1	1	1
20	1	1	1	1	1	1	1	1	1	1
21	1	1	1	1	1	1	1	1	1	1
22	1	1	1	1	1	1	1	1	1	1
23	1	1	1	1	1	1	1	1	1	1
24	1	1	1	1	1	1	1	1	1	1
25	1	1	1	1	1	1	1	1	1	1

Figure 3-11. Example Consists Comparison Report - Part II

consists comparison report

contents similarity summary

rows name	rows name
4 bench-stock-data-record-xm-pf	is a subset of 2 bench-stock-data-record-xm-pd
89 xm-zb-dectbl-input	is a subset of 2 bench-stock-data-record-xm-pd
3 bench-stock-data-record-xm-pe	is equivalent to 31 shop-stock-adj-data-record-b
89 xm-zb-dectbl-input	is a subset of 3 bench-stock-data-record-xm-pe
4 bench-stock-data-record-xm-pf	is a subset of 30 shop-stock-adj-data-record-a
89 xm-zb-dectbl-input	is a subset of 4 bench-stock-data-record-xm-pf
89 xm-zb-dectbl-input	is a subset of 5 bench-stock-data-record-xm-pa
7 calibration-required-by-item	is a subset of 6 calibration-reg-by-customer
89 xm-zb-dectbl-input	is a subset of 9 cross-reference-transaction
38 table-build-data-rec-xm-ya	is a subset of 11 equipment-recall-new-item-a
89 xm-zb-dectbl-input	is a subset of 11 equipment-recall-new-item-a
89 xm-zb-dectbl-input	is a subset of 12 equipment-recall-new-item-b

Figure 3-11. Example Consists Comparison Report - Part III

LOGICON

3.7 Requirement Traceability

Prior to a detailed review of MOM and MPOM specifications, it was thought that the MOM specification was a more detailed discussion of the MPOM system. Close examination showed that although many processing functions are the same, MOM and MPOM support different levels of Army maintenance management. These systems transfer information back and forth and must therefore be capable of interfacing. But requirements traceability as such was not possible. It is discussed here in the abstract because it is felt to be a key issue in the development and maintenance of complete/consistent specifications.

One objective of this study was to demonstrate the applicability of LARE to Army system specifications. What follows is a discussion of how requirements traceability is done. Several sample reports have also been included.

Most large systems have a hierarchy of documentation or specifications. Generally, the following types of information can be found:

- User notes, letters, concept papers -- proposals to support the initial systems engineering
- User requirements document -- statement of the system requirements from the user perspective
- System functional requirements document -- the general functional system requirements (GFSR) in Army nomenclature

LOGICON

- Detailed functional requirements document -- the detailed functional system requirements (DFSR) in Army nomenclature
- System test requirements -- the set of requirements for system testing prior to the government agency accepting delivery.
- System design specification -- detailed description of the system architecture (hardware and software) which will be built to meet the system required capability
- Hardware layouts and prints, or software listings

Many problems of system development and maintenance can be avoided by clearly documenting system requirements traceability. Development is improved because all requirements end up in hardware or software implementation and all key requirements get tested prior to system acceptance. Following the requirements from the top level all the way down to the hardware prints or software listings achieves traceability.

System maintenance (changes to the system as requirements change) can cost less. All too frequently, system failure results when the obvious changes are made in software modules. Good requirements traceability documentation clearly aids identification of all necessary changes, not just the obvious ones. LARE helps by listing closely related requirements, the analyst determines the necessary changes.

LOGICON

An example of a typical requirements traceability report is shown in Figure 3-12. The report is printed showing traceability from a higher-level to a lower-level specification (traceability between any two separate databases). The report automatically prints the traceability from the lower level to the higher level. It also lists requirements which exist in the specifications but which are not found in the databases and lists requirements for which there is no traceability in either direction.

3.8 Consistency and Completeness Analysis

Inconsistency or incompleteness of requirements is found by performing the tasks described in the preceding paragraphs. Rather than scatter the discussion a separate subsection summarizes and discusses the problems. While this analysis identified numerous problems, the analysis has been kept superficial to demonstrate the applicability of the technology rather than to exhaustively identify problems.

Tables 3-1 and 3-2 summarize:

- types of discrepancies
- LARE reports which aided discrepancy identification
- tally of discrepancies formally written-up

3.9 Requirements Specification Generation

A sample of text printed by LARE is shown in Figure 3-13. The statements or paragraphs of individual requirements are much shorter

INCONSISTENT DATA		INCOMPLETE DATA	QUESTIONS/OBSERVATIONS	TYPOGRAPHICAL ERRORS
<u>Inconsistent:</u> dic code field label references format element names mnemonic field length field name definition field type title name media	<u>Duplicated:</u> names subprocesses sources numbers pages Conflicting: title page/page content output values Missing: outputs inputs references cross references source numbers field checks alphabetical sequence	Missing Information: Fields: references significance field checks xiii code elements Design: customer priority processing soft criteria information entry rules value change checks error code numbers initialization data General: fields titles contract sheets prompts numbers numbers pages tables element names	Regarding: Design: legality range value max pgs loading transactions system limits p-won constructs flow charts Files: temporary files file name changes Duplication: duplicate inputs multiple field use duplicate elements Process: total number days error codes prompts reference calculations deletions sort field number change Reports: print reports name conventions report distribution element differs	Typographical errors though present were not noted. (Some typographical errors, were written up as samples.)

Table 3-1. Discrepancy Reports Breakdown

<u>Report Names</u>	<u>Inconsistent Data</u>	<u>Incomplete Data</u>	<u>Total</u>
Namelist Report	20	11	31
Structure Report	19	10	29
Process Chain Report	6	6	12
Consist-Compare Report	10	7	17
Content Report	20	5	25
Name Generation Report	15	6	21
Formatted Problem Statement Report	18	5	23
Process Chain Report	7	2	9
	115	52	167

*There were also 79 Discrepancy Reports labelled Observations/Questions and 4 labelled typographical errors, making a total of 250 Discrepancy Reports.

Table 3-2. LARE Reports Offering Most Visibility into Problems that were Recorded on Discrepancy Reports.

DATA BASES TRACED:
e3amsiss TO radar

DB1-REF	REQUIREMENT NAME	DB2-REF	REQUIREMENT NAME
m-3.1	dedicated-air-operation	r-1.2.a	air-surveillance-type-11
m-3.1	dedicated-air-operation	r-3.1.1.1.2.1	py-ms-surveillance-volume
m-3.1	dedicated-maritime-operation	r-3.1.1.1.2.2.5.1.b	ms-detect-maritime-targets
m-3.1	dedicated-maritime-operation	r-3.1.1.1.2.2.5.1.b	maritime-surveillance-type-viii
m-3.1	dedicated-maritime-operation	r-1.2.b	maritime-surveillance-type-viii
m-3.1	dedicated-maritime-operation	r-3.1.1.1.2.2.5.1	py-as-non-ecm-environment
m-3.1.1.1.1	py-clear-and-ecm-environments	r-3.1.1.1.1.e	py-as-ecm-environment
m-3.1.1.1.1	py-clear-and-ecm-environments	r-3.1.1.1.1.f	py-as-non-ecm-environment
m-3.1.1.1.1	py-clear-and-ecm-environments	r-3.1.1.1.1.f	py-as-ecm-environment
m-3.1.1.1.1	dedicated-air-operation	r-1.2.a	air-surveillance-type-11
m-3.1.1.1.1.a	dedicated-air-operation	r-3.1.1.1.2.1	py-ms-surveillance-volume
m-3.1.1.1.1.c	dedicated-maritime-operation	r-3.1.1.1.2.2.5.1.b	ms-detect-maritime-targets
m-3.1.1.1.1.c	dedicated-maritime-operation	r-3.1.1.1.2.2.5.1.b	maritime-surveillance-type-viii
m-3.1.1.1.1.c	dedicated-maritime-operation	r-1.2.b	maritime-surveillance-type-viii
m-3.1.1.1.1.c	dedicated-maritime-operation	r-3.1.1.1.2.2.5.1	as-resolve-range-ambiguities
m-3.1.1.1.1.2	process-radar-returns	r-3.1.1.2.1.1	detect-and-report-air-targets
m-3.1.1.1.1.2	target-detection	r-3.1.1.1	as-detect-range
m-3.1.1.1.1.2	target-detection	r-3.1.1.1	as-report-range
m-3.1.1.1.1.2	target-detection	r-3.1.1.1	as-detect-azimuth
m-3.1.1.1.1.2	target-detection	r-3.1.1.1	as-report-azimuth
m-3.1.1.1.1.2	target-detection	r-3.1.1.1	as-detect-elevation-angle
m-3.1.1.1.1.2	target-detection	r-3.1.1.1	as-report-elevation-angle
m-3.1.1.1.1.2	target-detection	r-3.1.1.1	as-detect-ecm
m-3.1.1.1.1.2	target-detection	r-3.1.1.1	as-minimize-effect-on-serv-vol
m-3.1.1.1.1.2.a.1	py-air-surveillance-volume	r-3.1.1.1.1.1.c	py-as-surveillance-volume
m-3.1.1.1.1.2.a.1	position-antenna-beam	r-3.1.1.1	detect-and-report-air-targets
m-3.1.1.1.1.2.a.1	position-antenna-beam	r-3.1.1.1	as-detect-range
m-3.1.1.1.1.2.a.1	position-antenna-beam	r-3.1.1.1	as-report-range
m-3.1.1.1.1.2.a.1	position-antenna-beam	r-3.1.1.1	as-detect-azimuth
m-3.1.1.1.1.2.a.1	position-antenna-beam	r-3.1.1.1	as-report-azimuth
m-3.1.1.1.1.2.a.1	position-antenna-beam	r-3.1.1.1	as-detect-elevation-angle
m-3.1.1.1.1.2.a.1	position-antenna-beam	r-3.1.1.1	as-report-elevation-angle
m-3.1.1.1.1.2.a.1	position-antenna-beam	r-3.1.1.1	as-detect-ecm

Figure 3-12. Example Requirements Traceability

LOGICON

than in the original specification. This is consistent with the recommended LARE methodology; each numbered paragraph must impose a requirement. Most paragraphs in the original specification contained multiple requirements.

Additional elements of the methodology, not used in this effort, provide editing of the text statements, automatically record changes in requirements and create a history file of previous requirements statements. The facility assists system configuration control and flags requirements which have changed since the analyst's last reading of the document. An example format used by Logicon for the Air Force's Joint Surveillance System (which required this capability) is shown in Figure 3-14.

3.10 Evaluation of Alternative Methodologies

Three requirements analysis methodologies were selected for a cursory comparative evaluation: LARE, SREM, and IORL. A brief overview is included to provide the reader a context for the comparative discussion.

LARE is a methodology built around Logicon's extended CADSAT. As stated earlier, LARE evolved from the University of Michigan's Problem Statement Language (PSL)/ Problem Statement Analyzer (PSA). In addition, LARE includes the Functional System Simulation Data Processing System (FSDPS) which assists system feasibility analysis and performance estimation. An overview of the LARE computerized support is shown in Figure 3-15.

2.1.1 For the MOH: Functional collector -- see data below.

2.1.1.1 For the MOH: For each item in Equipment Recall Requirements, a WO# shall be automatically assigned.

2.1.1.2 For the MOH: The Shop Office Clerk shall register an additional Intra-Shop WO, using the next alpha I/S code to the WO#. The WO# sequence number remains constant, keeping Intra-shop WO's together.

2.1.1.3 For the MOH: Maintenance Control Numbers with a start date within 90 days shall enter an assigned work order number on the TPR and WO#.

2.1.1.4 For the MOH: The Shop Office Clerk shall key for entering task data entry. When processor prompts, task requirements data shall entered.

2.1.1.5 For the MOH: The Shop Office Clerk shall enter Evacuation Data on the WO for higher level maintenance.

2.1.1.6 For the MOH: The Shop Office Clerk shall process customer maintenance requests.

2.1.1.7 For the MOH: The Shop Office Clerk shall register additional Intra shop WO's and assign the next alpha Intra shop code to the WO#.

2. For the MOH: The Shop Office Clerk shall enter work order parts data.

2.1.1.8.1 For the MOH: The Shop Office Clerk shall enter supplemental work order parts data when the Repair Parts form requires supplemental parts that are on hand. Copy 3 of the work Order Supplemental Parts Request shall be used for the entry.

2.1.1.8.2 For the MOH: The Shop Office Clerk shall enter WO parts reassignments, causing transfer to the TPR file.

2.1.1.8.2.1 For the MOH: There shall be reassignment of parts from one WO to another.

2.1.1.9 For the MOH: The registration of work order data is accomplished by the input of DIC X4A and DIC X4S (work Order Registration Data).

2.1.1.10 For the MOH: Text to be determined.

2.1.1.1 For the MOH: There shall be entry of WO requirements data for task, parts, or supplemental parts data in real-time to the TPR.

REQ-NO	USERS	STATEMENT OF REQUIREMENTS	NREV CREF SOURCE	SEGMENT SPEC
JRpa-780	A,C,US	Preserve at the simulation console positions the display and operator interaction capability available at standard operating positions.	0 a-jrpa-7 0-5.3.2.2.12	r-3.7.1.2.10.1.6
JRpa-790	A,C,US	Provide the capability for simulation operators to start, stop, advance and monitor external simulated data/inputs.	0 a-jrpa-9 0-5.3.2.2.12	r-3.7.1.2.9.2.3.6 r-3.7.1.2.9.2.3.b r-3.7.1.2.9.2.3.d
JRpa-800	A,C,US	Provide the capability for the simulation control consoles to input and alter specific Mode 2, 3, 4 and C responses simulated tracks.	0 a-jrpa-10 0-5.3.2.2.12	r-3.7.1.2.9.2.3.c
JRpa-810	A,C,US	Provide the capability for the simulation control consoles to enter simulated height on SIM flights for which mode C and/or previous simulated height is not available.	0 a-jrpa-11 0-5.3.2.2.12	r-3.7.1.2.5.4.3
JRpa-820	A,C,US	Provide the capability during a SIM/LIVE (mixed) environment to limit or exclude live data to specified console positions.	0 a-jrpa-13 buc-x	r-3.7.1.2.10.1.b
JRpa-830	A,C,US	The capability during the conduct of live exercises to ensure the safety of all aircraft through the use of Positive Target Control (PTC).	0 a-jrpa-14 0-5.3.2.2.12.4 c-57	r-3.7.1.2.9.3.a
JRpa-840	A,C,US	Provide the capability to assign a PTC team with the following data routed to the consoles: IFF data with Predesignated Mode 3 exercise codes, exercise flight plans, exercise tracks and, track data on tracks initiated by target monitors.	0 a-jrpa-15 0-5.3.2.2.12.4 c-57	r-3.7.1.2.9.3.d
JRpa-850	A,C,US	Provide the capability to start (simulated) data (flight paths) by specifying: the SKN, initial coordinates, heading, speed, altitude, and the specified type of sensor/strobe data.	0 a-jrpa-17 a-jrpa-21 0-5.3.2.2.12 buc-x	r-3.7.1.2.9.2.1.a
JRpa-860	A,C,US	The real time generated simulated flight path shall have the performance characteristics of the type aircraft being simulated.	0 a-jrpa-17 a-jrpa-21 0-5.3.2.2.12 buc-x	r-3.7.1.2.9.2.1.a
JRpa-870	A,C,US	Generate data for simulating input from and output to specific external agencies.	0 a-jrpa-18 c-48	r-3.2.1.9.1.4.e

Figure 3-14. Sample of LARE Generated Text - Format II

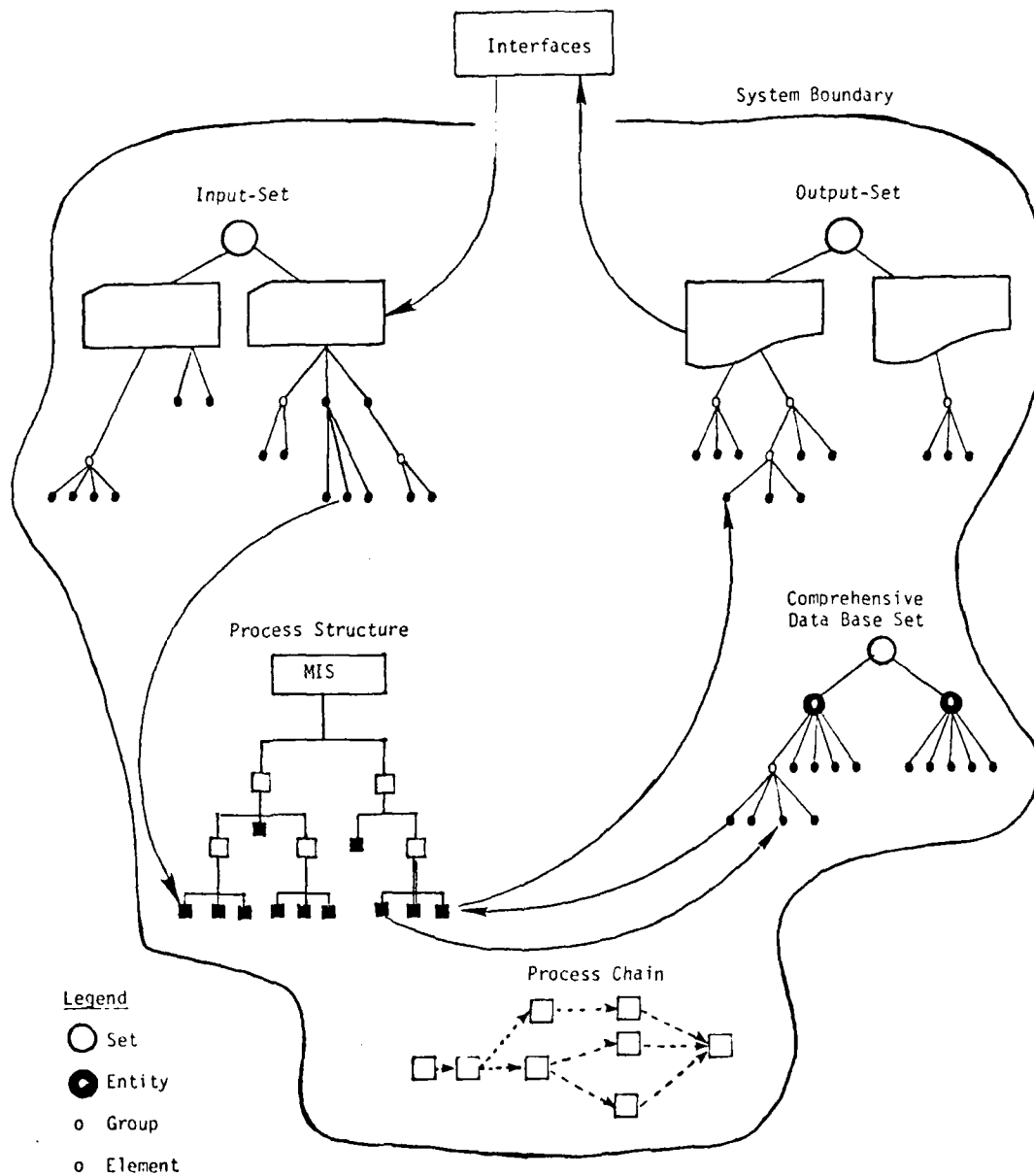


Figure 3-15. Schematic Diagram of CADSAT-Defined MIS, Structures, Control and Data Flow

LOGICON

SREM is a methodology built by TRW which uses computer assistance: Requirements Statement Language (RSL) and the Requirements Evaluation System (REVS). This tool also evolved from PSL/PSA. However, REVS moved much more radically from the standard PSL/PSA concepts. The only thing that remains is the Fortran data base management system (DBMS) at the heart of REVS. The input processors, analysis routines, and report generators have all been rewritten in Pascal. An overview of the functions and capabilities of REVS is shown in Figure 3-16. The REVS database containing the system description (defined by RSL statements) is called the Abstract, similar to the CADSAT or PSL languages but includes several enhancements (especially for representing functional flows - see Figure 3-17). One of the advantages of RSL is the ability to define or modify language constructs.

REVS appears capable of producing whatever outputs the user desires from any reasonable inputs. The user can perform either functional or analytical simulations. (This is the only tool reviewed which has an analytical simulation capability.) The difficulty for the user of REVS is that every module simulated must be described in Pascal statements and the user must write a driver in Pascal which simulates the external system environment. REVS provides three basic capabilities to support simulation: an executive controller, a set of simulation utilities, and automatically generated Pascal data descriptions for variables used by each module. The structure of the REVS simulator is shown in Figure 3-18.

Input Output Requirements Language (IORL) is supported by a system developed by Teledyne Brown Engineering. The language is not proprietary but the computer system processing the language is. The language may be described as a graphics language for describing either a set of system requirements or the actual system design. Functional interrelations are illustrated in Figure 3-19 and 3-20.

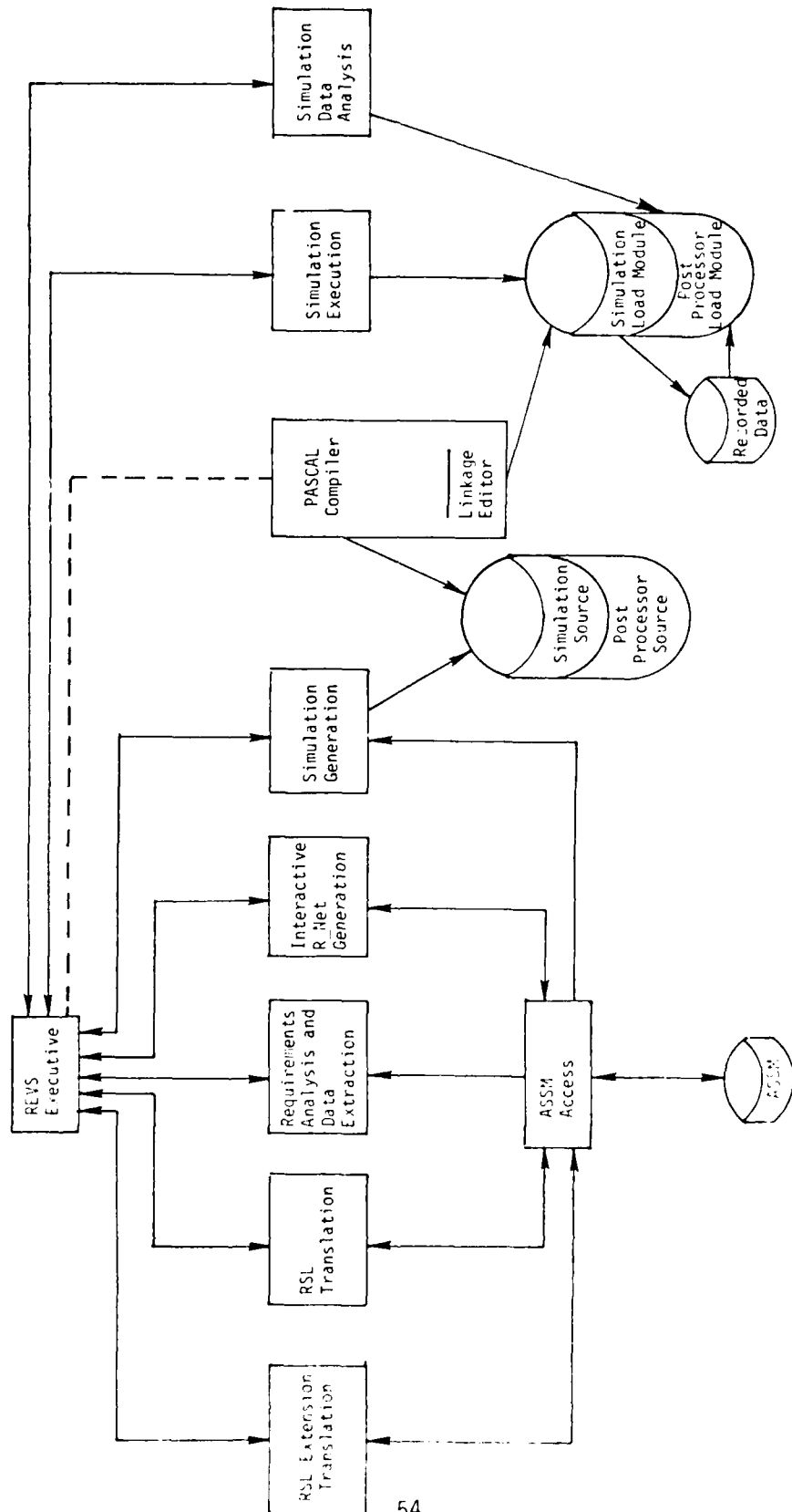


Figure 3-16. REVS Functional Organization

```

R_NET: SAMPLE.
STRUCTURE:
  INPUT_INTERFACE I1
  VALIDATION_POINT V1
  ALPHA A
  SELECT ENTITY_CLASS IMAGE SUCH THAT (Y < Z)
  DO
    ALPHA B
    FOR EACH FILE HISTORY RECORD
      DO SUBNET C END
    AND
    ALPHA D
    CONSIDER DATA STATUS
    IF (READY)
      ALPHA E
    OR (NOT_READY)
      ALPHA F
    END
  END
  IF (X > 5.0)
    ALPHA G
    VALIDATION_POINT V2
    OUTPUT_INTERFACE O1
    OR (X = 5.0)
    DO
      ALPHA H
      OUTPUT_INTERFACE O2
      AND ALPHA J
      ALPHA J.
      TERMINATE
    OTHERWISE
      EVENT Q
      TERMINATE
    END
  END.
  
```

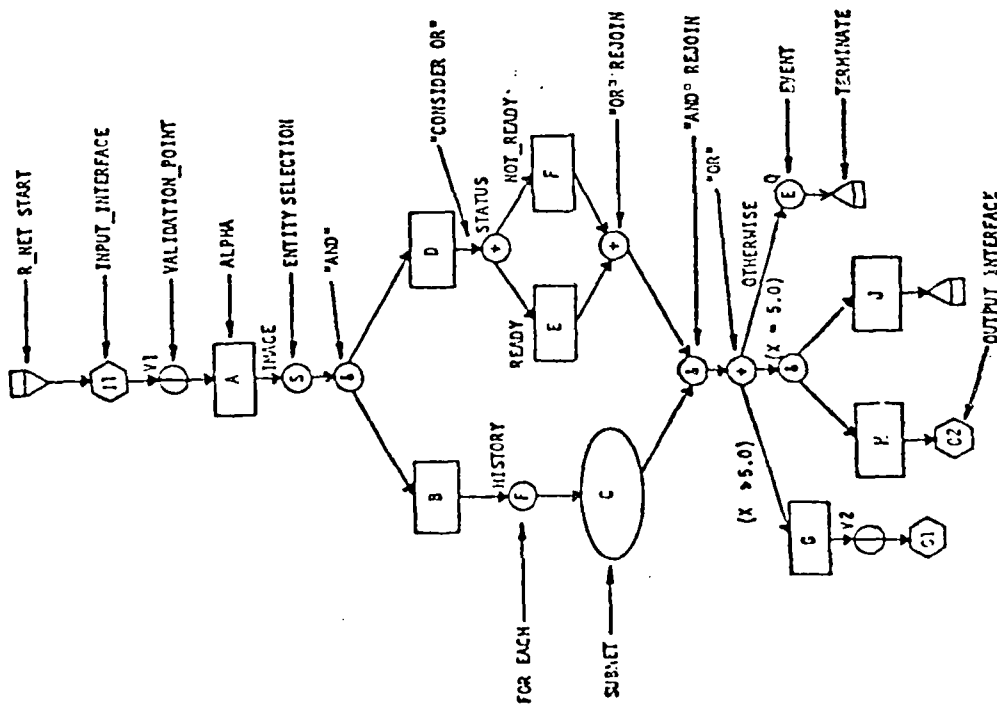


Figure 3-17. Sample R_NET Structure in RSL and in Graphical Form

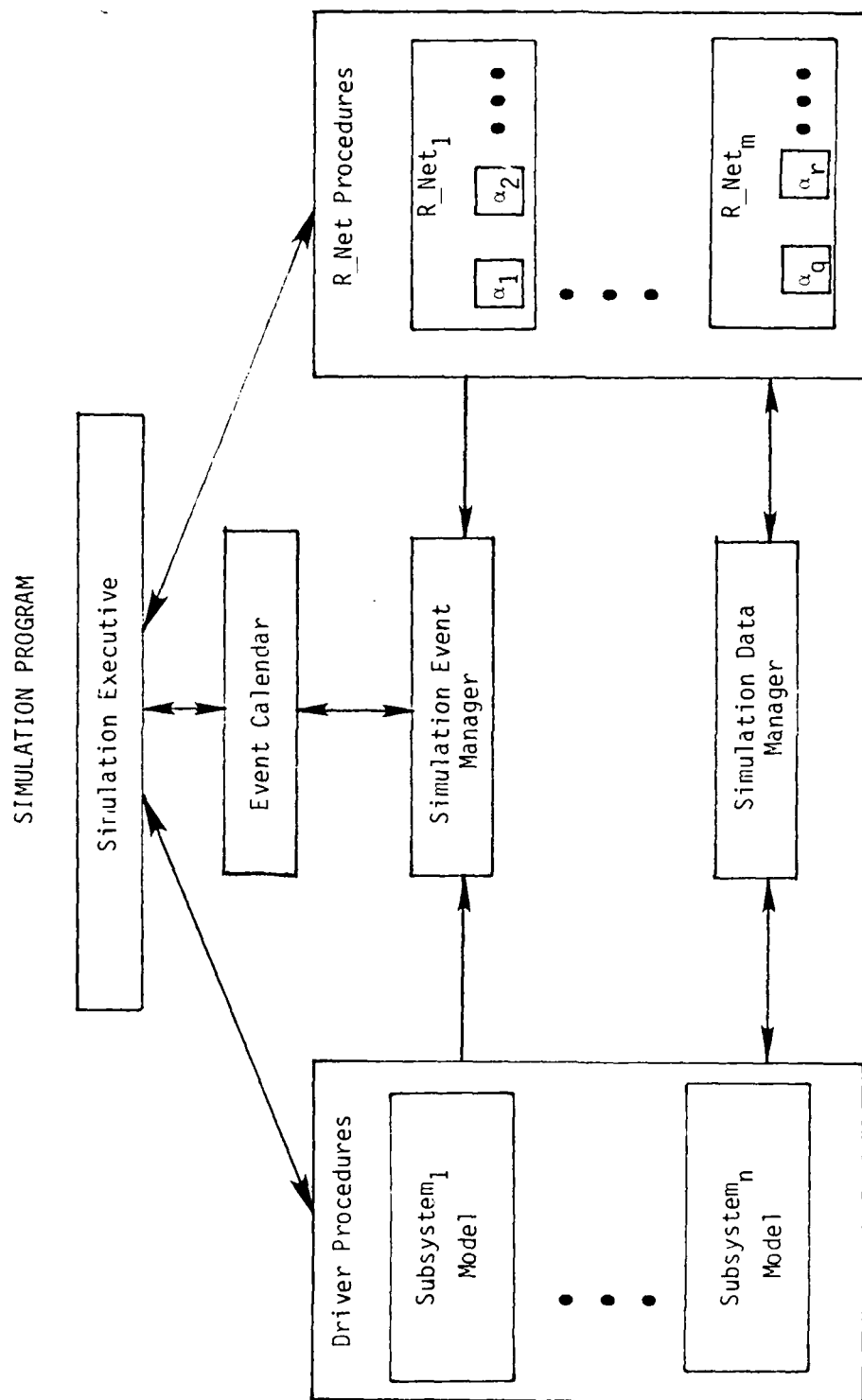


Figure 3-18. REVS Simulator Functional Components

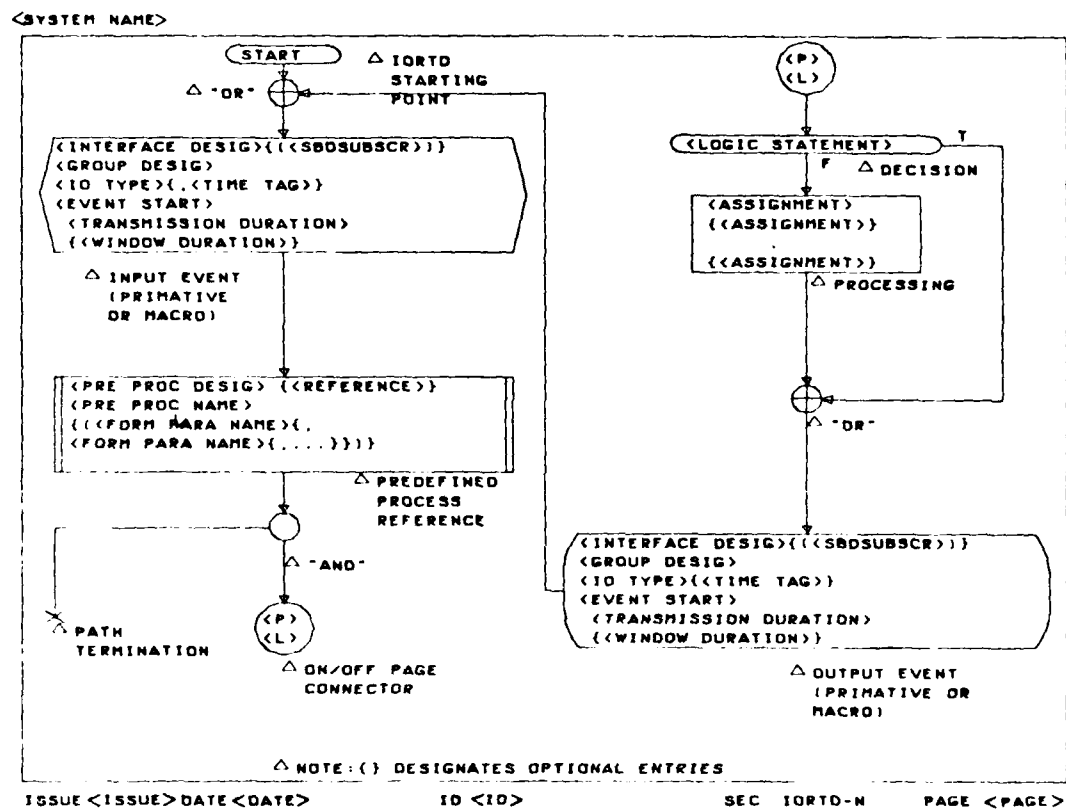
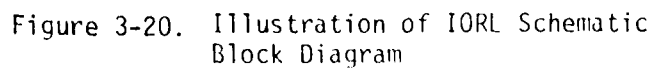


Figure 3-19. Functional Illustration of IORL



LOGICON

Table 3-3 shows the individual ratings assigned to LARE, SREM and IORL with respect to the twenty-four categories listed. Categories were weighted equally. The overall rating is a simple average.

No distinction was made between tool performance in the various categories and the degree of human expertise required to achieve results.

While we recognize that this is an overly simplified approach, a more comprehensive evaluation was not within the scope of the project.

Recommendations to reduce LARE's cited weaknesses are discussed in Appendix C. All enhancements involve improvements in the presentation or display of information.

LOGICON

		LARE	SREM	IORL
	<u>CRITERIA</u>		<u>RATING</u>	
Ease of:	database modification	2	2	2
	report generation	3	3	3
	report readability	2	2	2
Ability to:	record/depict functional requirements	3	3	3
	record/depict constraint requirements	3	1	1
	record/depict control flow	2	3	3
	record/depict data flow	2	2	3
	record/depict functional hierarchies	3	1	2
	record/depict data hierarchies	3	1	1
	record/depict interfaces	3	3	2
	record/depict external documents	3	2	1
	record/depict test requirements	2	1	1
	record/depict project status	3	1	1
Ability to:	aid in detection of incomplete data	3	2	1
	aid in detection of inconsistent data	3	2	1
	aid impact analysis	3	2	2
Ability to:	generate system specifications	2	1	1
	generate test specifications	2	1	1
	generate design documentation	2	2	2
Ability to:	simulate (general)	2	1	0
	trace requirements	3	1	1
	trace between levels of documentation	3	1	1
	provide well-defined methodology	3	3	1
	provide diagnostics	2	2	2
Totals		62	43	37
Overall Rating		2.58	1.79	1.54

Table 3-3. Evaluation Results

LOGICON

4. Conclusions and Recommendations

4.1 Conclusions Specific to this study

- This test program demonstrated that LARE can be applied to AIRMICS requirements. No technical barriers emerged, but some operational guidelines appear desirable. (These suggestions appear in the recommendations.)
- LARE's analytical effectiveness came through clearly on this project. Over 250 discrepancies were found in the specifications. They were found by several people in a short period of time (6 months) who had no familiarity with Army specifications or with maintenance systems. In addition, there was no contact with the system users although some clarifications were requested from AIRMICS. The problems identified are presumably a small sample of those existing. Problem identification was only one objective. Much more of the effort was spent documenting the methodology and illustrating how the methodology can be applied.
- Review of alternative tools identified three key performance differences:
 - LARE provides the best overall requirements engineering support.
 - SREM offers the best functional flow presentation but focuses on describing system design rather than requirements.

LOGICON

- IORL displays the best graphics representation of requirements.

The cursory review permitted in a short time limits the conclusions to these first impressions. Comparison was further hampered by insufficient internal documentation and inadequate experience in applying these other tools.

4.2 Conclusions Supported by this Study and Prior Experience

4.2.1 Basis of the Conclusions. General conclusions are based on the results of this study and Logicon's experience over the past six years with computer aided requirements engineering and development of software tools to support analysis. Some LARE capabilities could not be demonstrated by this project, including: configuration control assistance, feedback to analysts, defining the system, and assistance in analyzing impacts of requirements changes.

Since LARE was shown to be effective and applicable to AIRMICS, Logicon's prior experience suggests that LARE could also effectively handle communications and weapons systems.

4.2.2 Assistance to Configuration Control. Configuration control of system requirements includes many things. One element is control of the statement of requirements and the republication of documentation. LARE assists this process. LARE enables changes in requirements to be flagged as either insignificant (correction of typos) or significant (change of the actual requirement or constraint). In addition, the technology provides an ability to move obsolete requirements to history files and to retrieve these requirements for analysis. If a particular

LOGICON

area of requirements changes too frequently, a more general analysis and change is probably warranted. Demonstration of this capability would have required a longer time period with an iterative interface with the user.

4.2.3 Feedback to Analysts. The general experience of human and computerized communications is that people think communication is taking place when it is not. A technique used to improve this situation is to provide feedback to the speaker or sender. In the case of human communications, it is not sufficient for the analyst to write a textual document of the requirements and ask the user to answer yes or no as to whether it accurately represents user requirements. Several approaches exist for improving this communication prior to the discovery that a system has been developed that fails to meet major "requirements". One technique currently being explored by AIRMICS is "system sketching" which builds rudimentary system capability to enable the user to operate on samples of real data to verify the desired capability. The viability of this approach depends on the ability to develop "quick and dirty" solutions considerably cheaper than the final production system. A second approach involves the use of LARE. The requirements are expanded in levels and the implied control and data relationships are explicitly identified so that the user can see more clearly how the analyst is interpreting the user's statement of requirements (verbal or written).

4.2.4 Simulation of Systems. System simulation is frequently necessary to show the feasibility of system development and to bound both costs and technology risks. A recent Logicon study (Integration of CADSAT (LARE) with General Purpose System Functional Simulation Technology; Contract F04701-77-C-0069) illustrated LARE's capabilities in this field (referred to as the Functional System Simulation Data Processing System (FSDPS)). It is a completely generalized simulator. Current internal Logicon efforts include research into driving this simulator directly from a LARE database.

LOGICON

4.2.5 Time of Application. This demonstration applied LARE to an existing set of requirements specifications. While it was said earlier that the application was successful, the results could have been even more beneficial had LARE been used from the beginning of the effort (during the development of the first drafts of the requirements specifications, including the General Functional System Requirements documents). While it is advantageous to apply LARE as early as possible, the reader should not conclude that there is ever a time for which the application is too late. Logicon has had application experience on other projects in which the systems had already been built and implemented prior to using LARE. In terms of the system life cycle, the majority of the system life is during the operations and maintenance phase. During this phase, considerable effort is expended handling proposed requirements changes and attempting to determine the technical ramifications of these changes (what other requirements must change and how do these impact the implemented design).

4.2.6 Initial Planning of LARE Application. Another consideration is time. A month or two is needed for a single/couple of analyst(s) to sort through the requirements and develop a reasonable draft to the Requirements Engineering Plan. Analysts applied too soon in an uncontrolled manner contribute to poor results and a waste of time and money.

4.2.7 Time Required for Application. The effort should be more than six months of calendar time. The requirements definition phase generally takes place over a year or two and in some cases, several years. The issue is not one of putting enough people on the project but of having enough time for the analysts to think through the structural relationships of the requirements and get the appropriate reviews and feedback with the system user community.

LOGICON

4.2.8 Time Needed to Reorganize Database. Periodically, the requirements database needs to be reorganized. In the case of a requirements definition effort, the requirements expanding or the analysts' understanding of the requirements changing results in a need to modify the database. In the case of the analysis of existing specification, it becomes apparent that the functional breakout of the requirements is awkward and obscures relationships to the analyst and especially to the user. Time is needed for this reorganization. The ability to do this in a reasonably short period is one of the major advantages of having a computerized requirements specification. It should not be viewed by the analyst or management as a poor or inadequate job. The poor job is the failure to put effort into the reorganization when the methodology indicates it. The need for doing this type of iterative restructuring emphasizes the earlier point of having more than six months for requirements engineering.

4.2.9 Updating Capabilities. Many benefits of LARE were not realized or illustrated by this study because of the limited time period and scope. Much of the major effort of requirements engineering involves analysis of the problem and the loading of the initial databases. Once loaded, they provide an inexpensive and powerful capability to attack problems during both the system development or operations/maintenance phases of the system life cycle. Also, as additional use is made of the databases, the analyst discovers requirement relationships not previously perceived. These new relationships are frequently the realization of the interdependency (if one requirement changes, the other is likely to change) between requirements rather than identification of new requirements. As these new relationships are discovered, they should be loaded into the database. Thus, the LARE database provides a "corporate memory" of all analytical experience with a particular system. This is especially advantageous to deal with system personnel changes. A set of manual notebooks maintained by individual analysts could not be integrated nor expressed in as uniform a fashion.

LOGICON

4.2.10 Analysis of Proposed Requirements Changes. Proposed requirements change impact analysis is a major area, often overlooked.

People Logicon have spoken to have made the assumption that sufficient effort was going to be applied up front in order to eliminate the need for significant changes to the system. Our experience is that it is very unusual for the system to be implemented prior to major requirements changes. Two basic LARE capabilities lend themselves to this type of analysis: recording the interrequirements relationships within the computerized databases and reporting requirements traceability tracing the requirements from the top level requirements down to the individual hardware components, software modules, or personnel procedures.

4.3 Recommendations

- Improve LARE by incorporating the R_Net capability of SREM and augmenting the existing graphics capability with the use of IORL.
- Augment LARE's specification generation, data representation/display and information reporting capabilities in the areas detailed in Annex C.
- Apply the enhanced LARE to specify an Army system from initial requirements through implementation, using the following guidelines:
 1. Adequate time for initial requirements definition
 2. Develop a realistic Requirements Engineering Plan
 3. Require adequate "feedback" sessions between users and analysts

LOGICON

4. Allow time for reorganization of databases with respect to incorporating change
5. Simulate the proposed system

APPENDIX A
REQUIREMENTS ENGINEERING METHODOLOGY

A. INTRODUCTION

This Appendix details the requirements engineering plan used to analyze two Army Detailed Functional Specification Requirements (DFSR) specifications.

Each project must make decisions about which of the numerous LARE language features will best suit their individual needs. The main consideration in developing the AIRMICS specific plan was the determination of which LARE reports would best represent the information in the MOM and MPOM; and which reports would be used by analysts to detect problems. All of these decisions must be based on the goals and objectives of the project.

A.1 LARE Language Features. The specific language features chosen for this project are discussed at length in Appendix B. LARE is a language for describing system requirements and design. It is not a procedural programming language. It provides a capability for naming objects and providing textual descriptions of objects which play a role in the system. More importantly, it has the capability to define relationships among the objects and store/generate text associated with these objects.

Object names used in this application were: PROCESS, MEMO, INTERFACE, ATTRIBUTE, SOURCE, INPUT, OUTPUT, SET, GROUP, ENTITY, and ELEMENT. The main relationships used to describe the various aspects of the MOM and MPOM specifications were: RECEIVES, GENERATES (data across system boundaries), PART OF (system structure) CONTAINED IN, CONSISTS OF (data definition).

SYNONYMS, SOURCES, ATTRIBUTES and DESCRIPTIONS were used to reflect user-defined values and properties. Emphasis was placed on establishing a high level functional description of the system as well as the contents of the inputs and outputs used by the system.

A.2 LARE Report Features. There are three categories of reports/programs: application, update and utility. The application reports aiding Logicon analysts throughout this project are discussed briefly below and in detail in Appendix B. The update and utility reports are discussed only in this appendix.

LOGICON

A.2.1 Application Reports. The following reports were chosen to aid analysts in building the MOM and MPOM data bases:

DB Status (DBS)
Formatted Problem Statement (FPS)
Structure Report (STR)
Contents Report (CONT)
Name Generation (NG)
Name List (NL)

A.2.2 Database Update Programs/Reports. These programs enable the user to update the databases. The update programs commonly used on LARE-aided projects are:

- DELETE-PSL (DPSL)
Deletes specific LARE relationships previously established in the database. A permanent record of the change is generated in the form of the deleted - LARE Report.
- INPUT-PSL (IP)
Adds information to records in the database. A permanent record of the change can be generated in the form of the AS-IS Source Listing and Cross Reference.
- RENAME (REN)
Changes a name or list of names in the database. The Rename Report establishes a permanent record of the change.
- DELETE (DEL)
Deletes a name or a list of names from the database. When a name is deleted all of its connections to other names are deleted as well. A permanent record of the change is also generated in the form of the Deletion Report.
- PUNCH-COMMENT-ENTRY (PCOM)
Produces a punch file used as an input file to RCOM and DCOM. It is used for changing and deleting textual database entries.
- REPLACE-COMMENT-ENTRY (RCOM)
Replaces, for a given name, specific comment entries associated with the name. The Replaced Comment Entries Report records the change(s).

LOGICON

- DELETE-COMMENT-ENTRY (DCOM)
Deletes, for a given name in the database, the specified textual entry associated with the name. The Deleted Comment Entries Report records the change(s).

A.2.3 Utility Programs. These programs are used by LARE software maintenance personnel to backup, maintain databases (errors are occasionally experienced by system crashes or sudden communications terminations).

- DB DUMP
Dumps the contents of a database to the users' terminal showing pertinent information required for debugging any database problems. It provides a formatted dump of the internal database structure.
- DUMP
Dumps a database, in a sequential file format for input to RESTORE.
- RESTORE
To restore a previously dumped database. A database must be initialized before it can be restored.
- BCK
Used to back up databases from disk to tape.
- RST
Used to restore databases from tape to disk (previously backed by BCK).

A.3 Requirements Engineering Procedures. This section addresses why specific choices were made. Analysis, by definition, is an iterative process. Time constraints precluded multiple iterations. Decisions on how to best represent the data were made at the onset of the project. It was decided, for instance not to address the decision tables immediately. When they were addressed, it became apparent that the sheer volume and level of detail contained in them could be translated into LARE terms by example only. This is not to suggest that LARE cannot handle massive decision tables but that considerably more time would have been needed to make the conversion.

LOGICON

We decided to first concentrate on establishing a functional hierarchy of requirements for both the MOM and MPOM. This approach helps analysts working with an unfamiliar system towards an understanding of what the system is supposed to do. In addition, the functions provide the basic blocks or items, the objects for which the analyst must determine the interrelationships.

After agreeing on a structure suitable to both the MOM and MPOM, we diverged our emphasis. The MOM database was built to reflect inputs and outputs and the detailed elements contained in each. The emphasis was on Annexes A, B, and C of the MOM. We also loaded short textual statements that described the requirements as we determined them.

The MPOM database was taken several steps further. We established relationships in order to illustrate how LARE depicts control and data flows. This was in addition to information described with respect to the MOM database.

The Requirements Engineering Plan is dynamic in nature. As unique problems, specific to a project arise, they are evaluated and decisions about how to handle them are made. The plan must be updated accordingly to be an effective tool for the analysts.

Section A.4 is the plan given to project team members at the beginning of this study. It assumed, at a minimum, an existing understanding of LARE.

A.4 Procedures to be Followed on the Airmics Study Initial work on the Army specifications will concentrate on:

- Reviewing Source Documentation

- Identifying System Functions

- Organizing Functions into a Hierarchical Structure

A.4.1 Reviewing Source Documentation. Two requirements databases will be initiated and developed, one for each specification. Read the main sections of the source documentation. Make notes about your concerns and questions. If further reading or referencing various annexes does not address your problem(s), then write up a Discrepancy Report detailing the problem and/or question.

LOGICON

Highlight what you consider to be system functions and other pertinent system information.

A.4.2 Identifying Functional Requirements Use PROCESS (PRC) to identify system functions in both the MOM and MPOM. The following procedures are to be conformed to when deriving the name of a function:

- no more than 30 alphanumeric characters in a name
- no abbreviations unless it is to stay under 30 characters
- no special characters in lead field of name
- no embedded blanks (separate the words in names with hyphens)
- whenever possible, start the name of a functional requirement with an action verb

The object is to concisely name the abstracted function. Examples of action verbs used in the past are:

annotate	forward
inspect	transfer
enter	sort
record	distribute
generate	retrieve
submit	notify
determine	

Whenever possible, avoid using LARE-reserved words as the first word in an analyst-assigned name.

Use SOURCE (SRC) to identify where in the MOM or MPOM specifications you are extracting your information. If the information comes from the MOM, use the prefix m- and the appropriate paragraph number. For source paragraphs from the MPOM, use the prefix p-. If annexes are being referenced, use the appropriate lead prefix followed by the annex letter. For example:

m-a-i2.03.4D
p-p-03.04.8W

The first example indicates that the information was found in the MOM, Annex A. The second example indicates the reference is in the MPOM, Annex B. Paragraph references rather than page numbers were chosen for the source references because they are less apt to change during any later expansions of requirements and provide more accurate references to the specific requirements.

LOGICON

A.4.3 Organizing-Functions Into a Hierarchical Structure As you are defining functions, logical groupings will begin to surface. Use the part of (PART) convention to identify PART/SUBPART relationships. Remember:

- a hierarchy of functions should provide a concise overview of what the system does (is to do)
- a hierarchy of functions is independent of system flow
- a hierarchy, in order to make sense, may have to include high-level collectors not specifically addressed in the user documentation (collectors are analyst invented names for groupings of required functions. Collectors aid organization of requirements)
- a hierarchy should contain no more than 4 to 7 discrete requirements under any given aggregate function

A.4.4 Analyze Control Requirements. Use the TRIGGERS (TRGS)/TRIGGERED BY (TRGD) language feature to establish the flow of control. TRIGGERS should be used to specify processing functions that necessarily follow one another.

Do not use the UTILIZES/UTILIZED By feature. Time and volume preclude accurately defining primary and secondary functions. Remember constraints:

- Primary function - a function which is part of the major mission, objective, or goal of the system. e.g., provide listing of all available equipment.
- Secondary functions - a function which by itself is not part of the mission of the system. e.g., provide building space to house personnel and equipment required for the maintenance of a data processing system.

A.4.5 Analyze Data Requirements. The input and output requirements of both the MOM and MPOM are detailed in Annexes A and B. Use the INPUT (INP)/OUTPUT (OUT) constructs to identify inputs and outputs. Whenever possible, use the names already assigned. Define the inputs and outputs in terms of the ELEMENTS that they contain.

LOGICON

Use the SET construct to collect inputs and outputs with multiple parts.

Load INPUTS and OUTPUTS identified in the text before loading the annexes. This will facilitate locating requirements called out in the text but not detailed in the input/output sections and vice versa. It will also help point to inconsistent naming conventions.

Employ the USES/USED BY construct to indicate how data is used within the system. RECEIVES/GENERATES should be used to indicate how the data enters the system and who/what receives the OUTPUTS. Define the who/what by the INTERFACE construct.

A.4.6 Analyze Requirements for Consistency and Completeness. This should be done at each step of the project. System completeness will be virtually impossible to determine. Concentrate on consistency checks. Use the Discrepancy Reports to detail instances of inconsistencies and information that you determine to be incomplete.

A.4.7 Load Text. This will be done after the hierarchy has been built. Generate a structure report with assigned references. Build a file using the DESCRIPTION (DESC) construct. Use the specification text that best describes the requirement. If multiple requirements exist in one paragraph, extract text which suitably and independently describes each function you have named. Do not load the entire paragraph every time it is addressed.

A.4.8 Use of the Computer Center. These procedures are to be followed by AIRMICS project personnel:

- 1) Each analyst is responsible for building his/her own input files. All input file names are to end in ".INP".
- 2) Proof your own files and submit to a second party for proofing before database updates are requested.
- 3) Databases are to be updated by the Project Manager only. If specific sequencing is required for inputs, specify a numeric order in the name. (i.e., xxx1.INP, xxx2.INP) Once a series of files is ready for loading, leave mail in the PM's directory indicating such.
- 4) Request report updates before running them. This will ensure that we stay within budget.

LOGICON

- 5) Databases will be backed up daily, so there is no need to clutter directories with unnecessary files. On-line storage is expensive!
- 6) Your AIRMICS' directories are to be used for project-related work only.

LOGICON

APPENDIX B A DESCRIPTION OF THE LARE/CADSAT APPLICATION TO AIRMICS

B. INTRODUCTION. This appendix describes how Logicon has chosen to apply LARE to two Army DFSRs. Certain LARE terms (such as PROCESS, MEMO, TRIGGERS, etc.) which take on specialized meanings in this application, are discussed. More detailed information about LARE can be found in the documentation supplied by the University of Michigan (URL User's Manual, and URA User's Manual, Report No. ESD-TR-78-127, Volumes I and II).

B.1 Definition of LARE Terms

LARE terms used in this application are defined and explained in this section. Where abbreviations for terms are accepted by the LARE Programs, these abbreviations appear in parentheses following the names of the terms. A single sheet summary of the definitions and explanations is given in Table B-1, to assist the reader in recalling this information.

B.1.1 LARE Terms LARE names contain up to 30 characters with no embedded blanks; they represent objects in the CADSAT data base, such as PROCESSES, MEMOS or data aggregates. In this application, a name is commonly made up of several English words or abbreviations, separated by hyphens, to suggest the meaning of the object it represents.

B.1.2 PROCESS (PRC). A requirement is named as a process if it is a functional requirement, that is if it denotes an action which must be taken. Thus, for example, "process-work-orders" is named as a PROCESS.

PROCESSES are also used to "collect" non functional information represented in a specification. For example, chapters 1, 2, and 3 of the MOM and MPOM contain information about assumptions, benefits to be achieved, statutory and other regulatory requirements, etc. Paragraph reference numbers (sources) containing this type of information have been attached to "collectors". Examples of non-functional collectors are: header-only, non-functional-requirements, gfsr-assumptions, specific-dfsr-assumptions and safeguard-personal-data.

B.1.3 MEMO. A requirement is named as a memo if it can be stated as a constraint (such as sizing or timing). Whenever appropriate, a single

LOGICON

Table B-1. Summary of LARE Definitions

CADSAT Names:	Character strings up to 30 characters in length, English words or abbreviations from Table A-2 suggesting meaning, separated by hyphens
PROCESS:	Object naming a functional requirement - e.g., an action which must be taken
MEMO:	Object placing a constraint on a PROCESS or providing a description. MEMOS always apply to PROCESSES and must be so indicated on INPUT
SYNONYMS:	Given to all PROCESS, MEMO, INPUT, OUTPUT, ELEMENT
SOURCES:	Given to all MEMOS and PROCESSES Use alphabetic prefix for specification, numeric paragraph number separated by periods, multiple sources allowed.
DESCRIPTIONS:	Contains information relating to LARE object document text or description of contents of data items.
TRACE KEYS:	Placed in higher level specification database to provide traceability to a lower level specification; same format as SOURCES.
ATTRIBUTES:	Three types used - frequency, file length, file type, LODE (Annex C ref) media, location and status
PART, SUBPART:	Places processes.
APPLIES, SEE MEMO:	Associates MEMO with PROCESS.
TRIGGERS, TRIGGERED BY:	Defines executive flow of PROCESSES.
UTILIZES, UTILIZED BY:	Defines use of other PROCESSES as subroutines.
CONSISTS OF, CONTAINED BY	Defines data structural relationships.
USES, USED BY:	PROCESS uses data if it operates on it but does not change it.
DERIVES, DERIVED BY:	Applies to data aggregates generated by a PROCESS.
UPDATES, UPDATED by:	Applies to data that is changed.

LOGICON

memo may be applied to more than one process. Memos are also used to call attention to special cases and are helpful as a way of communicating between analysts. For example, a memo tag attached to a series of output requirements stating "no-inputs found" remind the analyst that he/she has unresolved problems. If the problem is resolved the memo is disassociated from the output requirements that have been satisfied and left associated with unresolved problems.

B.1.4 SYNONYMS (SYN). All LARE PROCESS, and MEMO, INPUT, OUTPUT and ELEMENT names were given SYNONYMS.

SYNONYM generation must be consistent if it is to be effective. Once a SYNONYM has been assigned to a 30-character name that name can then be referenced by the shorter synonym string. This becomes helpful when loading massive updates to a database. It was also effective, in catching instances of inconsistent name assignment of INPUT and OUTPUT elements in both the MOM and MPOM.

SYNONYMS were derived by using the first two letters of an analyst assigned LARE name. For example, the SYNONYM for "generate-wo-status-age-listing" is "gewostagli." In the case of input and output elements, two synonyms were assigned. One was a Logicon-generated SYNONYM for the ELEMENT name. The other is the Army-assigned mnemonic for data elements. For example, the Logicon SYNONYM for the ELEMENT "identifying-number-code-old" is idnucool". The Army SYNONYM is "ident-no-cd-old".

B.1.5 SOURCE (SRC). A source identifies the specification and the paragraph number from which a requirement is taken. A SOURCE is associated with a PROCESS or MEMO in a database in order to provide traceability from the database back to the governing specification. Each AIRMICS source has an alphabetic prefix identifying the specification, for example, "m-" denotes the MOM specification. Then, the specification paragraph number follows. For example, MOM Specification paragraph number 3-7a(1)(a) is m-3.7.a.1.a as a LARE source. "p-" denotes a MPOM paragraph number.

LOGICON

Source references extracted from the various specification annexes were coded as follows:

m-a-i2.40.ky fld.5

where

m = the specification itself
a = the Annex reference
remaining fields - represent references from either the input,
output,
flowchart or decision tables

Thus m-a-i2.40.ky.fld.5 indicates that the reference is from Annex A of the Maintenance Operations Management Specification, Float File Adjustment Input, 12.40.KY element found in field five (5).

Multiple sources are allowed for a single object name. These may occur because the same requirement is described from different points of view in two or more different sections. Multiple sources may also occur because a single requirement statement spans several subparagraphs.

B.1.6 DESCRIPTIONS (DESC). A description field is used to contain the text of the document paragraph from which the requirement was extracted. A description may be attached as a comment to a PROCESS, MEMO, INPUT, OUTPUT or ELEMENT. A description contains a maximum of 60 lines of text, with at most 72 character per line.

Descriptions associated with data items contain input events, output events, input and output controls; (if required) and in some cases, user preparation procedures.

B.1.7 Attributes (attr). Attributes were used to specify properties of a given section. Attributes assigned to input/output elements are: field length, field type and the Lode (the corresponding Annex C reference). Attributes assigned to OUTPUTS are frequency and media.

LOGICON

B.1.8 DATA AGGREGATES. Five LARE types of data aggregates were used to model the MPOM. They are SETS, INPUTS, OUTPUTS, ENTITIES and GROUPS.

Three LARE types of data aggregates were used to model the MOM. They are SETS, INPUTS and OUTPUTS.

B.1.8.1 SET. SETS can be defined as physical or logical views of the data as seen by the user, designer, and/or programmer.

B.1.8.2 INPUT (INP). An INPUT is used to describe a collection of information produced external to the target system. An INPUT shows the flow of data from the outside world into the system. Hence, it crosses the system boundary. The INPUT section is also used to uniquely identify each system input.

B.1.8.3 OUTPUT (OUT). An OUTPUT is used to describe a collection of information produced by the target system, then used external to that system. The OUTPUT section is used to show the flow of data from the system to the outside world. Hence, it crosses the system boundary. It can also be used to locate and uniquely identify each system OUTPUT.

B.1.8.4 ENTITIES (ENT). An entity is a logical, usable collection of data that serves a unique purpose within the system. An entity is information used by the target system that represents an object or concept internal to the system. It is required by the system for information processing purposes.

B.1.8.5 GROUP (GR). A GROUP is a logical collection of data elements and/or other GROUPS. A GROUP is a collection of information which can be contained in larger collections of information. INPUTS, OUTPUTS and ENTITIES. For example, a work order number could be defined as a GROUP containing supporting unit, intra-shop code, year and sequence. It was not; it was instead defined as an element. The ELEMENT (ELE) is the smallest item of data that can be referred to within the system and still maintain its unique properties.

B.1.9 LARE RELATIONSHIPS. As the LARE objects described above are the "nouns" in the syntax of the User Requirements Language, LARE relationships are the "verbs". The definitions of LARE relationships as they are used in the AIRMICS application appear in the following paragraphs.

LOGICON

B.1.9.1 PART, SUBPART (PART, SUBP). These relationships define the position of a PROCESS in the process hierarchy. For example, process-work-orders has the subparts enter-initial-wo-data, reconcile-wo-parts, enter-wo-status, close-out-wo, process-lookup-table-data, work-order-transfer, generate-work-order-data, edit-transactions, enter-parameter-data, update-internal-wo-files. Conversely, it can be said that the subparts are part of process-work-orders. These relationships apply to PROCESSES only. Each PROCESS may be part of only one higher-level PROCESS, but it may have multiple SUBPARTS.

B.1.9.2 APPLIES, SEE-MEMO (APPL, SM). These relationships define the connections between MEMOS and PROCESSES. MEMOS are said to apply to PROCESSES, conversely, PROCESSES are related to MEMOS via the SEE-MEMO relationship.

B.1.9.3 TRIGGERS, TRIGGERED by (TRGS, TRGS). These relationships define the flowchart structure in the execution of PROCESS. If one PROCESS TRIGGERS another, this means that the second PROCESS is executed after the first; conversely one may write that the second PROCESS is TRIGGERED by the first.

B.1.9.4 UTILIZES, UTILIZED by (UTIS, UTLD). A PROCESS that can be looked upon as a "subroutine" of another PROCESS and in this sense subordinate to that PROCESS, handled in one of the following two ways. If the "subroutine" PROCESS is called upon only once, then this PROCESS is TRIGGERED first and it in turn TRIGGERS the major PROCESS. If there are several of these "subroutine" PROCESSES, they are TRIGGERED in turn. However, when the "subroutine" PROCESS is called upon more than once, then the main PROCESS UTILIZES the "subroutine" PROCESS although it may TRIGGER other PROCESSES itself.

B.1.9.5 INTERFACE (INTF). The INTERFACE is an object, organization or system outside the boundaries of the target system that interacts with the system being described. It identifies the origin and destination of system products.

B.1.10 Data Relationships The relationships described in the following paragraph are used for data aggregates.

B.1.10.1 CONSISTS of, CONTAINED in (CSTS, CNTD). These relationships define how data items are organized in a structural hierarchy. ENTITIES may consist of GROUPS. GROUPS may consist of other GROUPS or ELEMENTS. ELEMENTS may not consist of anything else. Conversely, GROUPS may be contained in ENTITIES. Other GROUPS or ELEMENTS may be contained in GROUPS. INPUTS and OUTPUTS may contain GROUPS and/or ELEMENTS and be contained within SETS. Nothing else may be contained in ELEMENTS.

LOGICON

B.1.11 PROCESS/DATA RELATIONSHIPS. The following relationships define actions which processes perform on data aggregates.

B.1.11.2 DERIVES, DERIVED BY (DRVS, DRVD). A PROCESS derives a data aggregate (conversely the data aggregate is derived by the PROCESS) when it completes operation on data it has obtained and puts out a changed data aggregate. If the data internal organization is changed by a PROCESS, a new data aggregate (with a different name) is considered to be derived by the PROCESS. In this case one data aggregate is used by the PROCESS and another derived.

B.1.11.3 UPDATES, UPDATED BY (UPDS, UPDD). A PROCESS UPDATES a data aggregate when it changes, expands or deletes information in that data aggregate without changing its basic nature. The name of the data aggregate remains the same and no new data aggregates are created.

B.2 LARE REPORTS

This section gives introductory descriptions of the LARE reports which have been most commonly used by analysts for this AIRMICS applications. These reports are:

- Formatted Problem Statement (FPS)
- Structure Report (STR)
- Contents Report (CONT)
- Name Generation (NG)
- Name List (NL)
- Process Chain (PC)
- Data Process (DP)

In addition to the report descriptions, the formats of input files and modification files needed to build and maintain the databases are discussed.

It is worthwhile to emphasize the fact that since the content of any report is simply a reflection of the information which has been prepared by the analyst, considerable attention should be paid to preparation of input files. Subsequently, fewer errors will be encountered during input and reports will contain correct and useful information.

B.2.1 FORMATTED PROBLEM STATEMENT (FPS). This report makes available all names and relationships associated with a name (or with a list of names in a file) specified in the command line which generates the report. The format

LOGICON

of the command is:

fps n = process-work-orders where n = individual data base name

or

fps f = wo.inp where f = file name containing a
collection of names

where wo.inp is the name of the file segment containing a list of database names for which formatted problem statements are to be generated.

Figures B-1 and B-2 provide examples of FPS. Figure B-1 was produced by entering the command:

fps n = process-work-orders

Figure B-2 was produced by entering the command:

fps f = wo.inp

where the file name wo.inp contained the names of the five ELEMENTS shown.

Figure B-3 shows an FPS and the input file that generated the listing.

B.2.2 STRUCTURE REPORT (STR). This report gives an hierarchical structure of the PROCESS names in the specified database. The user has the option of including in the report MEMOS, SOURCES and/or relationships by which other names are associated with the PROCESS names present in the structure. Frequently, the analyst is interested in only a subset of the information in the database; the capability to produce a structure report on such subsets exists. The LARE command is:

str

At this point, the program requests the user to enter NAME, DEPTH, and OPTIONS. The NAME parameter can be any process name in the database whose SUBPARTS, to a specified DEPTH, will be reported in structure format. The DEPTH parameter, therefore, requires an integer representing the number of levels of SUBPARTS desired. Entry of a zero (0) for the NAME parameter implies that a full structure (i.e., all process names included) is required and entry of a zero for DEPTH will give all levels of SUBPARTS. Values for the OPTIONS parameter

LOGICON LEXINGTON VAX SYSTEM

formatted problem statement

parameters for: fps

```
name=process-work-orders noindex print empty notunch start=5 nmar=20 amarg=10 bmarq=25
rmar=70 char=1 npar=40 notestname one-per-line define comment none=page none=1-line
noall-statements complementary-statements line-numbers printeof
```

```
1 process                                process=work-orders;
2   synonyms are:  prwor;
3   subparts are:  enter-initial-no-data,
4                  reconcile-work-parts,
5                  enter-work-update-data,
6                  retrieve-work-data,
7                  recsit-work-status,
8                  close-out-work,
9                  process-lookup-table-data,
10                 work-order-transfer,
11                 generate-work-order-data,
12                 edit-transactions,
13                 enter-parameter-data,
14                 update-internal-work-files;
15   part of:      sars-retail-work-system;
16
17 eof eof eof eof eof
```

Figure B-1. Sample Formatted Problem Statement

LOGICUS LEXICON VAX SYSTEM

for "after problem" statement

parameters for: fos

```
title noindex print empty nonlunch status=5 nmarj=20 arearq=10 nmarj=25 rnmard=70 cmarq=1 hmarj=40  
nodesignate one-per-line define comment nonew-page nonew-line nodll-statements  
no-complementary-statements line-numbers printed
```

estimated-delivery-date:

```

1 element
2   synonyms are: eid,
3   esmeta;
4   attributes are:
5     field-length 13,
6     field-type   numeric,
7     loc-for      m-037-05;
8   contained in: supply-recn-13ta-record-an-x;
9   source is:    m-a.12.10.00.114.1);
10
11 element
12   estimated-delivery-date-order;

```

estimated-repair-parts-cost;

```

27 element
28
29 synonyms are:   estrefaco,      estimated=
30                  est-ref-part-cost;
31
32 attributes are:
33   field-length  L9,
34   field-type    numeric,
35   loc-den       =0027=07;
36
37 contained in:   parts-status-tape-record,
38
39               xt-rectel-inout,
40               xt-rs-rectel-inout;
41
42 used by:
43   xt-rs-rectel-inout-data-check;
44   source is:
45     aa-12.1342.f1.11;

```

Figure B-2. Sample Formatted Problem Statement Output

LOGICOR LEXILGION VAX SYSTEM

formatted problem statement

parameters for: fns

```

file nolnpx print empty notunch start=20 amarg=10 hmarq=25 rnmarg=70 emarg=1 hmarq=40
noteonline one-ccr-line define comment none=page none=+line noall-statements
complementary-statements line-numbers printed

```

```

1 process
2 description:
3 For the work: Reconciliation Parts Data shall be entered in real time
4 using a PIC of A.M.;
5
6 process
7 description:
8 For the work: The Shop Office Clerk shall enter supplemental "0
9 parts data when the Repair Parts form requires supplemental parts that
10 are on hand. Copy 3 of the work order Supplemental Parts Request
11 shall be used for entry;
12
13 process
14 syncs are: deell1;
15 description:
16 For the work: After All/DMU Requirements record is processed
17 Error Listing shall be generated (if error message present);;
18 Source is: m-5.13.c;
19
20 process
21 description:
22 For the work: Discrepances between reconciliation Subprocess and
23 the Log File shall be printed on the reconciliation Exception
24 report;;
25
26 process
27 description:
28 For the work: Manual requisition shall be generated for parts needing
29 lead time;;
30
31 process
32 description:
33 For the work: The Parts Port Data shall be generated consistent to MRK
34 reports, enabling the data at current date;;
35

```

TOP PARTIAL OF EXACT FILE
 WAS USED TO GENERATE
 LINE'S 1 THRU 35 OF
 THE FORMATTED PROBLEM
 STATEMENT

LINE	FILE
100	enter-reconciliation-parts-data
200	enter-wo-parts-supplemental-paa
300	gen-error-listing
400	gen-reconciliation-exception
500	generate-manual-requisition
600	inspect-parts-port-data
700	print-error-exception-rpt
800	proc-wo-parts-sup-parts-ed-sub
900	process-wo-consumption-parts
1000	request-work-order-requisition
1100	submit-dic-am
1200	xm-dt-decbl-input-datchek

Figure B-3. Example Formatted Problem Statement and Input File

LOGICON

are one or more of the following:

- r (include source references)
- m (include memos)
- a (include all associated names)
- p (none of the above; include process names only).

Two examples of valid responses to the program's prompt are:

```
00 r m
```

or

```
your-process 4 ra
```

See Figures B-4, B-5, B-6. Figure B-4 shows the structure of all levels under enter-initial-owo-data printed with the "p" option. Notice that one of its subparts is enter-standard-wo-data which has eight subparts itself. Figure B-5 is a structure generated with this name and the "a" option. It therefore includes all names related to each of the eight subparts.

Figure B-4 was produced by:

```
str
enter-initial-wo-data 0 p
```

Figure B-5 was produced by:

```
str
enter-standard-wo-data 0 a
```

Figure B-6 was produced by:

```
str
enter-initial-wo-data 0 r
```

Figure B-3 shows an FPS and the input file that generated the listing. And as shown, MOM paragraph references are included, as well as all SUBPARTS (or subprocesses) enter-initial-wo-data and enter-standard-wo-data.

process structure

parameters for: str

process indent=1 noindex

count level name

```

1 3 enter-initial-ao-data
2 4 enter-standard-ao-data
3 5 assign-equip-recall-reqmnt-von
4 5 assign-intra-shop-ao-numbers
5 5 ent=assign-work-order-number
6 5 enter-ao-order-task-data
7 5 enter-ao-evacuation-data
8 5 process-maintenance-request
9 5 register-intra-shop-work-order
10 5 enter-work-order-parts-data
11 6 enter-ao-supplntal-parts-data
12 6 enter-ao-parts-reassignments
13 7 re-assign-ao-parts
14 6 x=cp-decbl-input-datcheck
15 6 x=cs-decbl-input-datcheck
16 6 process-ait-subprocess
17 5 enter-registration-ao-data
18 6 x=a-decbl-input-datcheck
19 6 x=t-decbl-input-datcheck
20 6 process-ao-registration-outa
21 5 enter-ao-requirements-data
22 5 process-ao-requirements-tasks
23 4 enter-production-ao-data
24 5 est-crod-program-ait-getrm
25 5 establish-ait-production-prq
26 5 x=a-decbl-input-datcheck
27 5 process-ait-program-ait-sub
28 4 enter-ait-sro-data

```

Figure B-4. Sample Process Structure with the "p" Option

LOGICAL DEXALING: VAX SYSTEM

PROCESS STRUCTURE

Parameters for: STR

Process Index=1 noindex

count (level or relationship) names

```

1 4 enter-standard-xx-data
2 5 assign-equip-repell-repell-xxn
3 5 assign-intra-shop-xx-numbers
4 5 enter-assign-work-order-number
   uses work-order-number
5 5 enter-work-order-task-data
6 5 enter-work-order-xx-data
7 5 process-maintenance-request
   trans determine-repair-calibration
   trans determine-repair-maint-service
8 5 register-intra-shop-work-order
9 5 enter-work-order-parts-data
10 5 enter-xx-supplemental-parts-data
11 5 enter-xx-parts-reassignments
12 7 reassign-xx-parts
13 6 xx-connect-input-tatacheck
   uses file-input-action-code
   uses task-sequence-field
   uses identifying-number-code
   uses item-number
   uses failure-code
   uses routine-identifier-code
   uses signal-code
   uses advice-code
   uses unit-of-issue
   uses data-code
   uses repair-parts-nois-designator
   uses refl-source-code
   uses conf-designator-requis-action
   uses transaction-quantity-issue
   uses transaction-control-number
14 6 xx-connect-input-tatacheck
   uses file-input-action-code
   uses task-sequence-field
   uses identifying-number-code
   uses item-number
   uses failure-code
   uses routine-identifier-code
   uses signal-code
   uses advice-code
   uses unit-of-issue
   uses data-code
   uses repair-parts-nois-designator
   uses refl-source-code
   uses conf-designator-requis-action
   uses transaction-quantity-issue
   uses transaction-control-number
   uses maintenance-level-unit-code
   uses recall-interval-code
   uses condition-designator-warranty
   uses failure-detection-lurmo-code
   uses condition-code
   uses equipment-utilization-code
   uses weapon-system-designator-code
   uses task-sequence-field
   uses item-number
   uses identifying-number-code
   uses file-input-action-code
   uses transaction-quantity-issue
   uses document-control-number
   uses repair-parts-nois-designator
15 6 process-edit-subprocess
   uses xx-connect-input
   uses registration-xx-data
16 5 enter-registration-xx-data
17 6 xx-a-decrl-input-tatacheck
   uses documentation-identifier-code
   uses file-input-action-code
   uses unit-xx-code-support-unit
   uses unit-xx-code-customer-unit
   uses intra-shop-code
   uses end-item-component-indic-field
   uses identifying-number-code
   uses part-number-field
   uses conf-design-repairs-cust
   uses materiel-radi-reporting-oesid
   uses item-number-iter-noun
   uses sequence-number
18 6 xx-b-decrl-input-tatacheck
   uses file-input-action-code
   uses supplemental-data-code
   uses quantity-to-be-repaired
   uses type-maint-request-or-rep-code
   uses miles-xx-at-submis-of-xx-reu
   uses rounds-xx-at-submis-of-xx-reu
   uses author-xx-at-sprs-of-xx-reu
   uses maintenance-repair-code
   uses maintenance-level-unit-code
   uses recall-interval-code
   uses condition-designator-warranty
   uses failure-detection-lurmo-code
   uses condition-code
   uses equipment-utilization-code
   uses weapon-system-designator-code

```

Figure B-5. Sample Process Structure with the "a" Option

process structure

parameters for: str

process indent=3 noindex

count level name

reference

```

1 4 enter-standard-ao-data      m-4.14.b
2 5 assign-ao-call-recall-ao    m-4.10.x.2
3 5 assign-intra-shop-ao-nuops  m-4.15.c
4 5 enter-ao-order-number       m-4.17.d
5 5 enter-work-order-task-data  m-4.10.d
6 5 enter-ao-evacuation-data    m-4.10.a
7 5 process-maintenance-request m-4.14.e
8 5 register-intra-shop-work-order m-4.10.x.2
9 5 enter-ao-order-parts-data   m-4.12.g
10 5 enter-ao-supplemental-parts-data m-4.12.h
11 6 enter-ao-parts-reassignments m-4.12.k
12 7 re-assign-ao-parts        m-5.5.a.0
13 6 x-process-ao-input-data-check
14 6 x-process-ao-input-data-check
15 6 process-ao-input-data-check
16 5 enter-registration-ao-data m-5.4.a
17 6 x-process-ao-input-data-check
18 6 x-process-ao-input-data-check
19 6 process-ao-registration-data
20 5 enter-ao-requirements-data m-5.4.b
21 6 process-ao-requirements-data m-5.10.c
22 6 process-ao-requirements-data m-5.10.c

```

```

level count level count level count level count level count
1 5 2 0 3 0 4 1 5 10
6 5 7 1

```

Figure B-6. Sample Process Structure with "r" Option Only

AD-A100 847

LOGICON INC LEXINGTON MA
AIRMICS LARE/CADSAT ANALYSIS.(U)
APR 81 K M TERRELL, L A JOHNSON
R81003(A)

F/6 9/2

UNCLASSIFIED

DAHC26-80-C-0021
NL

2 OF 2

AD A
100 847

END
DATE
FILMED
7-81
DTIC

LOGICON

and as shown, MOM paragraph references are included, as well as all subparts (or subprocesses) enter-initial-wo-data and enter-standard-wo-data.

B.2.3 Contents Report (CONT). The Contents Report gives a structure of data names (as contrasted with process names). The INPUTS to "contents" must be ENTITY or GROUP names because these are the objects which consist of other GROUPS or of ELEMENTS. An example of a Contents Report appears in Appendix D, Figure D-4.

```
cont n = mpom-internal-data
```

or

```
cont f = your-file
```

are examples of valid commands. The file your.file would usually contain several entity/group names.

B.2.4 Name-List (NL). Name List shows alphabetically every name in the database. An example NAME-LIST appears in Appendix D, Figure D-8.

B.2.5 Name-Generation (NG). Name Generation is a useful method of creating file(s) containing database names, to be used as inputs to other report generating programs. The user controls the types of names to be included by specifying values of a selection parameter. The selected names will be extracted from the database and put into a file which is usually used as input to other commands. The format of the command is:

```
ng s = "some boolean expression"
```

or

```
ng s = "some boolean expression" punch = your.file
```

Inclusion of the punch file parameter is optional. When it is omitted, a default file name is used.. The selection parameter is designated by the "s".

LOGICON

The general forms for the boolean expression are:

"operand operator operand"

or simply

"operand"

Operands are legal data base names types (process, keyword, etc.) and operators can be & (AND) and (OR). The following three examples of possible command lines should clarify this description.

```
ng s = "process" punch = proc.inp
ng s = "entity group"
ng s = "process & keyword = m-5.8.c " punch = your.inp
```

The first command will put all PROCESS names into a user defined file named proc.inp. The second command will extract all of the database names which have been defined as either GROUP or ENTITY and put the list into a default file in the users directory. This command also illustrates the method of using the output file from ng as input to another command. If the user were to enter:

cont

following execution of the second ng example, a Contents Report would be produced giving a structured list of the contents of each GROUP or ENTITY name in NG's output file.

Following execution of the third NG example, the names of only those processes with the associated keyword m-5.8.c will be extracted from the database and put into your.file.

```
ng s = some process-inp
```

or

```
pc f = some.inp
```

B.2.6 Data Process (DP). The output of Data Process depicts, in matrix format, the relationships between data and processes. A data item is an ENTITY, GROUP or ELEMENT, a relationship is USES, USED BY, DERIVES, DERIVED BY, etc. The report also includes a brief analysis of each matrix and states any inconsistencies in the data flow.

When generating this report, the user has the option of specifying either data items or PROCESSES as INPUT. The report may be produced for a single name is necessary.

LOGICON

To generate the report, enter one of the following commands. Note, however, that when the input file is not specified in the command line, the Data Process will assume the existence of Names-Gens's default file in the user's directory.

```
dp d
dp d n = name-of-data-type
dp d f = your.file
```

The 'd' in the above commands implies that data item names are being used as input so the program will then search the data base for the related PROCESSES to generate the matrices.

When the user enters on of the following commands

```
dp p
dp n = process-name
dp p = your.file
```

The program expects to receive process names from the input and will then find An example Data Process report appears in Appendix D, Figure D-5.

B.2.7 Input Files (IP). The content of input files will vary depending upon whether the analyst is in the early stages or later stages of data base development. Typical early stage input consists of:

```
PROCESS names
SOURCES
MEMOS
DESCRIPTION statements.
```

An example will best illustrate the required format of input files.

Assume that several requirements are known, some of which are sub-requirements of others. Name them process-1a, process-1b, process-2a, process 2-b, etc. Now let their respective B5 paragraph references be k-1a, k-1b, k-2a, etc. It may also be necessary to input text which describes some of the requirements, such as MEMOS. All of this information must then be entered into the database base in a structured manner. Examine the following input (IP) file. This general format is required for correct data entry.

Later stage input involves using all of the CADSAT name types and relationships which are valid for AIRMICS application. Analysts set up relationships (such as TRIGGERS) between PROCESSES allowing data flows and process chains to be defined.

LOGICON

B.2.8 Modification Files. There are four types of database modification files:

```
DPSL  (Delete Problem Statement Language)
DEL   (Delete)
REN   (Rename)
CT    (Change Type)
```

B.2.9. DPSL Files. The format of DPSL files is identical to that of ip files but are used to "disconnect" relationships between names. Consider the following dpsl file:

```
prc   process-1a,
key   k-1a,
subp  process-x,
eof,
```

After DPSL processes the above input, the keyword k-1a is no longer associated with process-1a and process-x is no longer a SUBPART of process-1a. However, all three names still exist in the database. The form of the command is

```
dpsl  f = your.inp
```

B.2.10 DEL Files. The format of the Delete command is one of the following:

```
del   n = some name
el    f = some.file
```

This command is used to actually delete names from the database entirely (and therefore the relationships to other names). The form of the input file is simply a list of data basenames to be deleted:

```
some-process-1a
some-keyword
some-process-b
some-data name
```

B.2.11 Rename Files. The format of the Rename command is one of the following:

```
ren  p = old name      n = new name
ren  f = some.file
```

LOGICON

This command is used to rename objects in the database (you may also want to change their synonyms). The input file format is:

```
old-name-a    new-name-a
old-name-b    new-name-b
.
.
.
```

B.2.12 Change Type Files. The change type command is either

```
ct n = some.name    t = new type
```

or

```
ct f = some.file    t = new type
```

Change Type is used to correct the "type" of database objects. For example, if a number of objects were mistakenly entered as GROUP names and should have been ELEMENTS the following command should be entered:

```
ct f = some.file    t = element
```

where some.file contains:

```
element-a
element-b
element-c
.
.
.
```

APPENDIX C
PROPOSED LARE ENHANCEMENTS

C. INTRODUCTION

This study and other previous applications have found LARE an effective methodology/tool for analyzing and defining system requirements. Even though Logicon has been successful, with a number of LARE enhancements could make the technology even more effective and easier to use. Proposed enhancements have been grouped into three categories: specification generation, data representation/display, and information reporting.

C.1 Specification Generation

One of the chief advantages of LARE is the ability to generate text specifications directly from the computerized databases. Enhancing the capability to provide the following is desirable:

C.1.1 Generate Automatically Maintained Table of Contents

The ability exists to generate text paragraph numbers automatically based on functional hierarchical structure or generate the text in an arbitrary order based on predefined paragraph numbers. The missing capability is to generate the table of contents including the key text phrase and the appropriate page number.

C.1.2 Develop a Generalized Specifications Generation Language

Logicon has generated text for system specifications in several different formats. Changing the format requires software adaptation of the specification generator. What is desired is a simple specification generation language which would permit the user to define the specification format, and enable the use of specialized symbols embedded in the text to control printing format. The capability should include the option to indicate specific terms for inclusion in an index.

LOGICON

C.2 Data Representation/Display

The presentation of information could be improved to aid the analyst in understanding relationships among requirements the inconsistencies or incompleteness of the requirements. The following presentation improvements have been identified:

C.2.1 Upgrade Process-Chain Report to Provide Relational Control Flow

The Process-Chain Report should include an option to allow display of control flow relations, whether explicit or implicit, at any level of a functional hierarchy. This capability will allow the analyst to identify loops and logic errors. With the example in Figure C-1, the user could specify "Level =1" and have the Process-Chain depict control flow at the highest level of the functional hierarchy (X TRIGGERS Y, in the example) even though the relationship is implicit. This would allow the analyst to look at any level of the functional hierarchy without losing information.

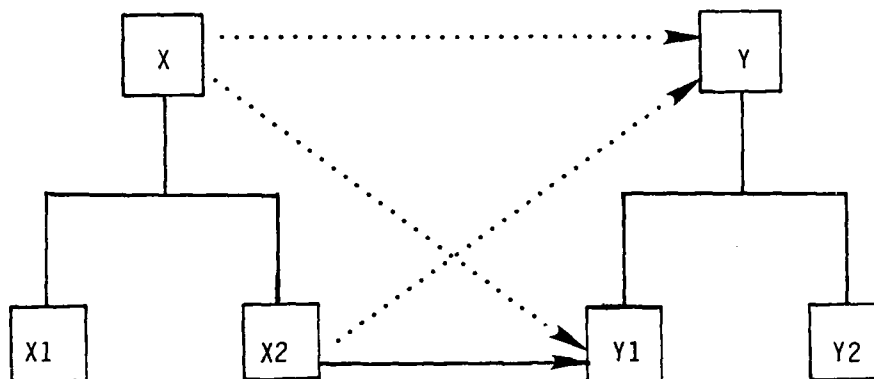
C.2.2 Improve Reports to Provide Relational Data Flow

This modification would impact the Data Process, Extended Picture, and Process Input/Output Reports. The most natural technique for implementing this modification would infer existence of data flow relationships, which are specified at any level of functional hierarchy, into all higher-level functions. Thus, with the example in Figure C-2, the USES/DERIVES relationship would be inferred by LARE to apply to functions X and Y. (in addition to their subparts). This modification improves the visibility of data flow relationships.

C.2.3 Augment Reports to Provide Relational Data Structure

This modification would affect the Data Process, Extended Picture, and Process Input/Output Reports. Whenever a data name is specified with a data flow relationship, all lower levels of the data name's hierarchy should be included in the relationship.

LOGICON



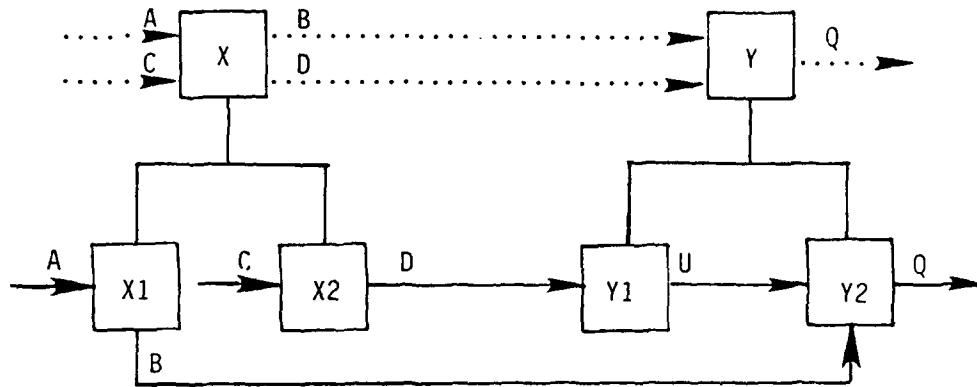
PROCESS X;
SUBPARTS X1, X2;

PROCESS Y;
SUBPARTS Y1, Y2;

PROCESS X2;
TRIGGERS Y1;

_____ = control flow depicted by Process-Chain
..... = control flow not depicted by any report

Figure C-1. Example of Control Flow Anomaly



PROCESS X
SUBPARTS X1, X2;

PROCESS Y
SUBPARTS Y1, Y2;

PROCESS X1,
USES A;
DERIVES B;

PROCESS Y2
USES B, U;
DERIVES Q;

PROCESS X2;
USES C;
DERIVES D;

PROCESS Y1;
USES D;
DERIVES U;

_____ = explicit data flow
..... = implicit data flow

Figure C-2. Example of Data Flow Anomaly

LOGICON

C.2.4 Add Condensed Listing

A condensed listing option would allow the user to format pictorial reports in listing form. This, modification in conjunction with a pictorial report, could guide the analyst in identifying links across page boundaries or it could be used independently. Unlike long pictorial reports, the condensed listing report is easy to follow. Figure C-3 shows one possible format. Its corresponding system diagram is shown in Figure C-4.

C.3 Information Reporting

There are several circumstances in which the handling of relevant information is awkward and the information not readily available. The following recommended enhancements are of this type:

C.3.1 Enhanced to NAME-GEN to Provide Source References

NAME-GEN should be extended to generate a list of all reference numbers contained in the database within a given interval (i.e., 3.2.1 - 3.2.7). This would simplify completeness checking during specification analysis and generation. In general, it would be helpful if users could supply alphabetic or numeric ranges of objects to be selected by NAME-GEN for use by additional report generations.

C.3.2 Expand Contents Reprots to Provide Source References for Data Types

The LARE Contents Report should be extended to allow data structure reports similar to the Logicon Extended Structure Report. The Contents Report should present SOURCE references for each data type, in addition to the relationships appropriate for data (DERIVED BY, UPDATED BY, etc.). This extension would consolidate information now contained in separate LARE reports.

C.3.3 Expand Triggers Relation to Allow Boolean Conditions

LARE should be augmented to handle conditional control flow and Boolean conditions. This feature would increase LARE's ability to accurately record specification information.

LOGICON

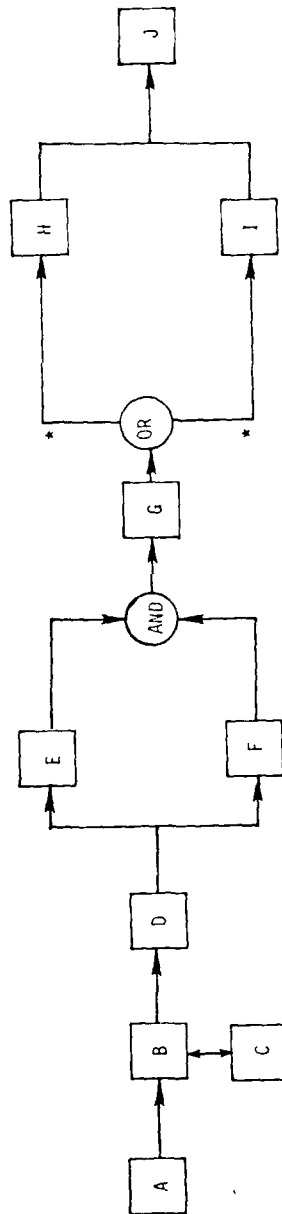
Line No*	Count+	References +			
1	5	Function-A			s-3.7.2
2	15	Function-B	UTIL	Function-C	s-3.7.2.1
3	7	Function-D			s-3.7.2.1
4	9	Function-E			s-3.7.2.2
5	2	Function-F			s-3.7.2.2
6	10	Function-G	TRGD WHEN	E&F complete	s-3.7.2.1
7	14	Function-H	WHEN	condition-name	s-3.7.2.2
8	15	Function-I	WHEN	condition-name	s-3.7.2.2
9	12	Function-J			s-3.7.2.3

Notes:

- * - This is a sequential number for easy reference to a specific line in the report.
- + - These correspond to the hierarchical level of number of the INTF, PROC, and I/O types in their respective structure reports.
- + - These are source document paragraph numbers for the adjacent INTF, PROC, or I/O types.

Figure C-3. Example Control Flow Report Using the Indented Format

SERIES: B is performed
after A



UTILIZES: B utilizes C
to perform its
activities

AND: E & F must
be performed
before G

OR: H or I will
result in J;
* the conditions
upon which alternative
paths are selected

Figure C-4. Control Flow Diagram

LOGICON

This feature increases CADSAT's ability to accurately record specification information.

C.3.4 Add Source-relationship Tags

Neither relationships (USES, DERIVES, etc.) nor conditions can have SOURCES attached to them. Develop an ATTACH statement which attaches a SOURCE name to anything in the database. No strong need for this modification has been uncovered by this study, but the feature would have been used if it had been available.

C.3.5 DBSTATUS Short Form

The DBSTATUS report needs a short form which includes only the sources, names and index value. This report could produce the "table of contents" for the structure report at less expense than the current, full DBSTATUS report.

APPENDIX D

Sample LARE Outputs

D. INTRODUCTION

Eight of the typical LARE reports used in the performance of the AIRMICS study have been included:

- STRUCTURE - structure report showing the functional hierarchy of the MOM and MPOM. (Figures D-1 and D-2)
- CONSISTS MATRIX - matrix and listing detailing those elements contained in specific inputs in the MPOM. (Figure D-3)
- CONTENTS - report showing data structure relations in the MPOM. (Figure D-4)
- DATA PROCESS - this report depicts in matrix format, the relationship between data and processes. (Figure D-5)
- DATABASE STATUS - report showing the status of each requirement and the types of relations defined. (Figure D-6)
- FORMATTED PROBLEM STATEMENT - report depicting all of the information in the database for the specified items. (Figure D-7)

LOGICON

- NAME LIST
 - report showing all names in the database.
(Figure D-8)

process structure

parameters for: str

process indent=3 noindex

count level name

reference

1	1	sams-retell-mom-svstem	
2	2	produce-support-plan	
3	3	maint-sudoort-plan	
4	4	process-cross-ref-upd-sub	m-n-table-no1150
5	2	process-work-orders	
6	3	enter-initial-wo-data	
7	4	enter-standard-wo-data	
8	5	assign-equip-recall-reqmnt-won	m-4.14.b
9	5	assign-intra-shop-wo-numbers	m-4.10.x.2
10	5	ent-assigned-work-order-number	m-4.15.b
			m-4.17.a
			m-4.10.g
			m-4.10.x.5
			m-4.10.g
			m-4.10.r
			m-4.14.e
			m-4.10.x.2
			m-4.12.d
			m-4.12.h
			m-4.12.k
			m-4.10.ss
			m-5.8.o
12	5	enter-wo-evacuation-data	
13	5	process-maintenance-request	
14	5	register-intra-shop-work-order	
15	5	enter-work-order-parts-data	
16	6	enter-wo-submntal-parts-data	
17	6	enter-wo-parts-reassignment	
18	7	re-assign-wo-parts	
19	6	xm-cp-dectbl-input-datacheck	
20	6	xm-cs-dectbl-input-datacheck	
21	6	process-ent-subprocess	m-n-table-no231
22	5	enter-redistribution-wo-data	m-5.8.a
23	5	xm-a-dectbl-input-datacheck	
24	6	xm-p-dectbl-input-datacheck	
25	6	process-wo-registraton-data	
26	5	enter-wo-requirements-data	m-5.8.b
27	6	process-wo-requirements-tasks	m-5.10.b
28	4	enter-production-wo-data	m-n-table-no201
29	5	ent-prod-program-mnt-detrn	m-4.17.j

Figure D-1. LARE Process Structure of the MOM

process structure

count level name	reference
30 5 establish-maint-production-prq	m-5.9.f
31 5 xm-d-ecdtbl-input-datacheck	
32 5 process-maint-ord-dat-inp-sub	m-h-table-no551
33 4 enter-alt-sro-data	
34 3 reconcile-wo-parts	
35 4 forward-wo-rep-burts-frm	m-4.10.y.2
36 4 transfer-data-to-rbf	m-4.12.c
37 4 enter-reconciliation-parts-data	m-5.8.k
	m-5.10.c.1
38 4 den-cust-o-reconciliation-p1	n-5.9.k
39 4 den-cust-o-reconciliation-p2	n-5.9.k
40 4 den-reconciliation-exception	m-5.10.f
41 4 den-reconciliation-response	a-5.10.f
42 3 enter-wo-update-data	
43 4 annotate-wo	
44 5 annotate-work-order	
	m-4.12.g
	n-4.13.d
	m-4.10.11
	m-4.10.0.1
	n-4.10.0.3
	m-4.10.1f
45 4 key-wo-change-data	
46 5 process-wo-carts-adj-sub	m-h-table-no1101
	m-h-table-no1
47 4 enter-wo-completion-data	m-4.10.q
	m-4.10.q
48 5 xm-d-ecdtbl-input-datacheck	
49 5 xm-d-ecdtbl-input-datacheck	
50 5 process-wo-consumption	m-h-table-no300
51 5 process-wo-consump-parts	
52 4 enter-work-order-status-update	
	m-4.10.w.3
	m-4.10.mm
	m-4.10.pp
	m-4.10.aa
53 5 xm-s-dectbl-input-datacheck	
54 5 process-wo-status-data-supp	m-h-table-no1001
55 4 enter-work-order-time-expended	m-4.10.q
	m-4.10.x.5
56 5 xm-dl-dectbl-input-datacheck	
57 5 process-wo-consumption-mannrs	m-h-table-no371

Figure D-1. LARE Process Structure of the MOM (Cont'd.)

Process structure

count level name	reference
58 4 enter-class-inspectn-form-data	m-4.10.o.5
59 4 inspectn-vo-status-change	
60 4 reenter-work-order-copies	m-4.10.z
61 4 transfer-wo-data-computr-media	m-4.10.q
62 4 work-order-update	
63 5 real-time-process-work-orders	m-4.14.i
64 4 change-maint-proc-prgm-data	m-5.8.f
65 4 change-work-request-status	m-5.8.i
66 4 enter-task-completion-data	m-5.8.c
67 5 xm-ct-dectbl-input-datcheck	
68 5 xm-dt-dectbl-input-datcheck	
69 5 xm-st-dectbl-input-datcheck	
70 5 process-task-compl-edit-supp	m-h-table-no301
71 4 enter-labor-usage-data	m-5.8.c.1
72 4 xm-w-dectbl-input-datcheck	
73 4 xm-x-dectbl-input-datcheck	
74 3 retrieve-wo-data	m-5.8.g
75 4 retrieve-annotated-work-orders	m-4.13.e
76 4 retrieve-work-order-jacket	m-4.20.n
77 4 request-wo-registration-data	m-4.10.x.3
78 4 retrieve-tpc-data	m-5.8.g
79 4 xm-h-dectbl-input-datcheck	
80 4 process-inquiry	m-h-table-no601
81 3 report-wo-status	
82 4 print-standard-wo-reports	
83 5 gen-monthly-work-order-report	m-4.11.hh
84 5 print-error-exception-rpt-wo	m-4.11.i
	m-4.12.a.1
	m-4.12.u
	m-4.11.h
	m-4.10.d
	m-5.9.a
85 5 print-open-status-work-orders	
86 5 find-open-work-orders	m-4.11.h
87 5 prepare-work-order-info	m-4.10.d
88 6 generate-wo-summary	m-5.9.a
89 5 generate-wo-testister-status-rp	m-5.9.i
90 5 generate-wo-status-age-listing	m-5.9.i
91 5 generate-workload-summary	m-5.9.c
92 4 print-production-reports	
93 5 gen-maint-program-control-doc	m-5.15.a
94 5 gen-maint-program-status-rp	m-5.15.b

Figure D-1. LARE Process Structure of the MOM (Cont'd.)

count level name	process structure	reference
95	4 print-alt-sro-reports	
96	5 gen-alt-sro-application-report	m-5.13.d
97	5 gen-alt-sro-schedule	m-5.13.c
98	4 notify-customer	m-4.10.t
		m-4.10.u
		m-4.10.v
		m-5.9.i
		m-5.9.n
		m-5.9.r
99	4 generate-transfer-file	
100	4 gen-joc-register-open-supply	m-4.10.o
101	4 transfer-data-to-rpm	m-4.10.m
102	3 close-out-xo	m-5.9.j
103	4 print-xo-registr-closed	m-5.9.o
104	4 work-order-closeout	
105	4 gen-xo-register-close-1-listing	m-4.10.tt.2
106	5 gen-doc-register-closed-supply	m-5.8.d
107	3 process-lookup-table-data	
108	4 insert-lookup-table-data	
109	4 delete-lookup-table-data	
110	4 enter-lookup-table-data	
111	4 xw-y-decbl-input-datatype	
112	3 work-order-transfer	
113	4 send-xo-with-equipment	m-4.20.b
114	3 generate-work-order-data	
115	4 generate-xo-number	m-5.8.a
116	3 edit-transactions	m-5.8.p
117	3 enter-parameter-data	m-5.8.r
118	4 xz-zo-decbl-input-datatype	
119	4 xz-zh-decbl-input-datatype	
120	4 xz-zc-decbl-input-datatype	
121	4 xz-zd-decbl-input-datatype	
122	4 xz-ze-decbl-input-datatype	
123	4 xz-zf-decbl-input-datatype	
124	4 xz-zg-decbl-input-datatype	
125	4 xz-zh-decbl-input-datatype	
126	4 process-param-duty-hours	m-h-table-nol1295
127	4 process-param-norm-norm-data	m-h-table-nol1280
128	4 process-param-prev-cyc-dat	m-h-table-nol1285
129	4 process-param-repts-control	x-h-table-nol1274
130	4 process-param-work-age	m-h-table-nol1213
131	4 process-param-workld-back-age	m-h-table-nol1233
132	4 process-parameter-subo	m-h-table-nol1200

Figure D-1. LARE Process Structure of the MOM (Cont'd.)

count level name	process structure	reference
133 4 process-param-pts-sta-det	m-h-table-no1255	
134 3 update-internal-no-files		
135 4 gen-error-exception-listing	m-5.9.n m-5.9.m m-5.12.c	
136 4 generate-contingency-files		
137 4 update-daos-file		
138 2 process-task-performance-data		
139 3 sample-task-performance-data		
140 3 report-task-performance		
141 3 distribute-tpf-data	m-4.18.b m-5.8.m	
142 3 enter-task-perf-factor-adj		
143 3 task-perf-fact-adj-subp		
144 2 maint-parts-inventory		
145 3 inter-annot-shop-sup-list	m-4.12.r.6	
146 3 annotate-repair-parts-form	m-4.12.f m-4.12.g m-4.12.h m-4.12.i m-4.12.n m-4.12.r.7 m-4.10.rr	
147 3 enter-shop-supply-data		
148 4 enter-part-number-change		
149 5 xm-n-dectbl-input-datacheck		
150 4 enter-repair-parts-form-data	m-4.12.o m-4.17.c	
151 4 generate-manual-requisition	m-4.12.r.1	
152 4 submit-dlc-aml	m-4.12.e	
153 4 enter-da-form-data	m-4.12.a	
154 4 enter-bslf-change-data	m-4.12.w.2	
155 4 post-cancel-reconcil-resp-meda	m-5.8.j m-5.10.a.1 m-5.10.d.2 m-5.10.e	
156 4 change-bench-stock-list		
157 5 process-bench-st-adj-require		
158 4 change-part-number	m-5.8.i	
159 4 change-shop-stock-list	m-5.8.j m-5.10.a m-5.10.d.2 m-5.10.e	
160 5 xm-pa-dectbl-input-datacheck		
161 5 xm-pb-dectbl-input-datacheck		

Figure D-1. LARE Process Structure of the MOM (Cont'd.)

count	level name	process structure	reference
162	5 xm-pd-decchl-input-datcheck		
163	5 xm-pc-decchl-input-datcheck		
164	5 process-ssl-adjustment-subp	m-h-table-no900	
165	5 process-ssl-head-data-edit-sub	m-n-table-no955	
166	4 enter-patts-receiot-data	m-5.8.k	
		m-5.10.c.1	
167	4 enter-patts-status-data	m-5.8.x	
		m-5.10.c.2	
168	4 enter-patts-usage-data	m-5.8.c.2	
169	4 update-shop-stock-req-status	m-5.10.d.1	
170	3 print-part-number-mismatch-list	m-4.12.r.1	
171	3 print-reconcilth-excptn-report	m-4.12.v.1	
172	3 print-shop-stock-list	m-4.12.u	
173	3 print-ssl-constnt-list	m-4.12.u	
174	4 generate-constraint-report	m-5.10.d.3	
175	3 print-ssl-locator-list	m-4.12.r.2	
		m-4.12.u	
176	3 print-supply-activity-reqmnts	m-4.12.r.10	
177	3 print-supply-reports		
178	4 process-prt-no-chy-da-subp		
179	3 print-zero-balance-stock-list	m-h-table-no1300	
180	3 record-patts-status-on-rpf	m-4.12.r.4	
181	3 record-received-patts	m-4.12.b	
182	3 generate-monthly-report		
183	3 print-error-excptn-report	m-4.12.u	
184	3 generate-rpon-transfer-file	m-4.11.o	
185	3 transmit-accptd-work-orders	m-4.10.e	
186	3 parts-trackino-handling	m-4.10.ff	
187	4 prepare-manual-requisition	m-4.12	
188	3 sort-repair-patts-forms-lbd	m-4.12.1	
189	3 transmit-manual-requisition	m-4.12.a	
190	3 transmit-ko-rep-patts-form	m-4.12.1	
191	3 gen-patts-awaiting-disc-action	m-4.12.b	
192	3 gen-patts-status-detail-list	m-5.10.d.4	
193	3 gen-shopstock-zero-balance-ren	m-5.9.m	
194	3 gen-supply-activity-req-list	m-5.10.d.3	
195	3 xm-rab-decchl-input-datcheck	m-5.10.d.4	
196	2 maint-equipment-data		
197			
198			

Figure D-1. LARE Process Structure of the MOM (Cont'd.)

count level none	process structure	reference
199 3 record-equip-inspec-data		m-4.7.o
200 4 inspect-inob-report		m-4.7.k
201 4 print-nots-norm-data		m-4.11.1f
202 4 annotate-proner-forms		m-4.10.1
203 4 gen-inoperative-equip-file		m-5.9.f
204 4 gen-nots-norm-data-listing		m-5.9.s
205 4 gen-nots-requirements-list		m-5.9.e
206 3 determine-required-maint		
207 4 determine-required-calibration		m-4.14.e
208 4 determine-required-maint-service		m-4.14.e
209 3 process-equip-recall		
210 4 gen-equip-recall-delinq-list		m-4.14.m
211 5 gen-equip-recall-delinquency-list		m-5.12.b
212 4 gen-equip-recall-schedule		m-4.7.v
		m-4.14.c
		m-5.12.a
		m-4.7.v.8
213 4 print-equip-recall-delinq		m-4.14.m
		m-4.14.e
		m-5.8.d
		m-h-table-no400
214 4 retrieve-equip-recall-schedule		
215 4 change-equip-recall-data		
216 4 x-e-declbl-input-datacheck		
217 4 process-equip-recall-subproc		
218 3 process-equip-evacuation		
219 3 control-float		
220 4 enter-float-adjustments		m-4.13.1
221 5 xm-f-declbl-inout-datacheck		
222 5 process-float-file-adjustment		m-h-table-no501
223 4 indicate-rejected-float-candid		
224 4 issue-float-exchange-notice		m-4.13.g
225 4 assign-ort-transaction-code		m-4.13.1
226 4 print-float-candidate-report		m-4.13.k
227 4 print-float-status-report		m-4.13.b
228 4 retrv-prev-float-candidate-rep		m-4.13.1
229 4 compare-float-calc-params		m-5.8.e
230 4 change-float-authorization		m-5.11.f
231 4 enter-float-changes		m-5.11.b
232 4 gen-float-candidate-rep		m-5.11.c
		m-5.11.d

Figure D-1. LARE Process Structure of the MOM (Cont'd.)

count	level name	process structure	reference
233	4 generate-float-status-rep		m-5.11.e
234	3 enter-equipment-usage-data		m-5.11.a
235	4 enter-usage-data		m-5.11.e
236	4 gen-usage-data-survey		m-4.10.jj
237	4 gen-usage-exception-list		m-5.8.n
238	4 xm-u-xmv-decrl-input-datchek		m-5.14.b
239	4 process-usage-subproc-subp		m-5.14.c
240	3 print-equipment-usage-data		m-h-table-no1051
241	3 process-equipment-usage-data		
242	3 proc-trj-build-ecc-sta-sto-ing		m-h-table-no1171
243	2 maint-personnel-data		
244	3 enter-personnel-data-changes		m-4.10.qq
245	3 print-work-center-summary		m-4.11.b
246	3 output-magnetic-transfer-media		m-4.11.dd
247	4 load-labor-util-detail-tape		m-4.11.dd
248	4 load-transfer-data-file-tape		m-4.11.gg
249	3 print-labor-util-summary		m-4.11.dd
250	3 print-workload-status-aqrpt		m-4.11.4
251	3 change-personnel-assignments		m-5.8.n
252	3 change-toe-tde-authorization		m-5.8.p
253	3 gen-labor-utilization-summary		m-5.9.p
254	3 generate-work-center-summary		m-5.9.p
255	3 updt-labor-utilization-detail		m-5.9.g
256	3 xm-l-decrl-input-datchek		m-5.9.h
257	3 process-wrk-ctr-lbr-edt-subp		m-h-table-no801

level count	level count	level count	level count	level count
1	2	3	4	5
6	11	6	120	52

Figure D-1. LARE Process Structure of the MOM (Cont'd.)

Process structure

Parameters for: str

Process [indent=] no[index

count (level or relationship) names

```

1 1 same-retail-mmom-system
2 2 manage-support-plan
3 3 develop-support-plan
4 4 determine-workload-histroy
5 3 maintain-support-plan
6 4 maintain-workload-data
7 4 monitor-data-for-reports
8 4 print-workload-reports
9 3 accept-key-entries
10 4 accept-key-report-generation
11 4 accept-key-date-info
    rcs key-entrv-innnts-mmom
    unds report-end-date-ordinal
    unds report-start-date-ordinal
12 2 process-work-order-data
13 3 enter-initial-no-data
    memo cant-justify-dtm
14 3 reconcile-no-parts
    memo cant-justify-mmof
15 3 enter-no-data-update
16 4 edit-input-transactions
    memo cv-resolve-errors-offline
17 4 accept-mmom-vor-torf
    utlm ident-input-errors
18 4 update-mmom-files
19 3 retrieve-no-data
20 4 monitor-inhs-awaiting-parts
21 4 monitor-open-work-orders
22 4 search-mmop-files
23 4 search-cvorf-data-for-rept
    rcs key-entrv-innnts-mmop
    trns compute-turnaround-time-index
    uses report-end-date-ordinal
    uses completed-work-order-hist-file

```

Figure D-2. LARE Process Structure of the MPOM

process structure

```

count (level of relationship) names
uses report-start-date-ordinal
drvs evaluation-hold-file
trgm mainin-valid-transaction-file

3 report-no-status
4 control-report-generation
4 generate-monthly-reports
4 generate-weekly-reports
3 close-out-wo
  memo cant-justify-nrof
3 process-lookup-table-data
3 work-order-transfer
  memo cant-justify-nrof
2 maintain-parts-inventory
3 monitor-repair-parts
3 print-critical-share-parts-list
2 maintain-equipment-data
3 record-equip-inspec-data
3 determine-required-maint
3 process-equip-recall
3 process-equip-evaluation
3 control-float
3 enter-equipment-usage-data
3 print-equipment-usage-data
4 print-inob-equip-recort
  memo pv-display-by-criteria
4 generate-inob-equip-card
  memo pf-run-daily
  trgm sort-with-ulc-acc
3 process-equipment-usage-data
4 monitor-incremental-equip
4 update-inob-equipment-file
4 maintain-inob-equipment-data
2 process-task-performance-data
3 sample-task-performance-data
  memo cant-justify-nrof
3 report-task-performance
4 print-maint-cost-summary
  memo sup-maint-cost-summary-afp
  trgm record-maint-cost
4 print-support-cost-summary

```

Figure D-2. LARE Process Structure of the MPOM (Cont'd.)

process structure

```

count (level of relationship) names
gens support-maint-cost-sum-ecc
trsd record-support-maint-cost
trsd process-sort-with-cchrf
53 4 print-maint-evaluation-reports
54 3 distribute-tnf-data
55 3 provide-cost-data
56 4 record-support-maint-cost
57 4 record-maint-cost
trsd print-maint-cost-summary
trsd calc-support-maint-cost
58 3 perform-maintenance-evaluation
59 4 compute-turnaround-time-index
trsd sort-data-by-criteria
trsd print-suc-maint-turnover-unit
uses evaluation-hold-file
trsd search-cwrf-data-for-rent
uses evaluation-hold-file
60 4 sort-data-by-criteria
trsd print-suc-maint-turnover-ecc
uses evaluation-hold-file
uses ulc-unit-name-cross-ref-file
trsd ecc-turnaround-time-data
trsd compute-turnaround-time-index
61 4 calc-support-maint-cost
trsd record-maint-cost
trsd lead-data-to-label-tables
62 2 maintain-personnel-data
63 3 monitor-man-hours-applied
64 4 monitor-unit-activity-manhours
trsd print-unit-meantime-repair-mhrs
65 4 monitor-ecc-meantime-to-repair
trsd print-ecc-meantime-repair-rtt
66 3 report-man-hours-applied
67 4 print-unit-meantime-repair-mhrs
gens supt-maint-man-act-rent-unit
trsd monitor-unit-activity-manhours
68 4 print-ecc-meantime-repair-rtt
gens supt-maint-man-rent-ecc
trsd monitor-ecc-meantime-to-repair

```

Figure D-2. LARE Process Structure of the MPOM (Cont'd.)

process structure

```

count (level of relationship) names
69 3 calculate-labor-costs
70 4 load-data-to-labor-tables
    tras calc-support-maint-cost
    uses valid-transaction-file
    uses equipment-category-code
    uses hourly-rate
    uses labor-overhead-rate-fla
    uses other-costs
71 4 process-sort-with-cxort
    tras print-support-cost-summary
    uses completed-work-order-list-file
    uses report-start-date-ordinal
    
```

level count	level count	level count	level count	level count
1	2	3	29	4
1	2	3	29	35

Figure D-2. LARE Process Structure of the MPOM (Cont'd.)

CADSAT version 3.2R2

LOGICON LEXINGTON VAX SYSTEM

JAN 26, 1981 09:56:07

page 39

consists matrix report

workload-status-age-rep-out
work-order-summary-rep
work-order-sheet-dist
work-order-repair-parts-form
equipment-recall-schedule-out
work-order-registr-closed-out
work-order-regis-stat-out
error-exception-report-supply
output-work-order
float-candidate-report-out
equipment-recall-delinq-list
management-notification
inop-equip-report-out
work-order-copy-two-out
work-order-copy-three-out
work-order-copy-one-out
work-order-copy-four-out
work-order-copy-five-out
maint-float-exchange-notifictn
part-number-mismatch-listing
rejected-sdc-mr40
transfer-file-data-to-mpom
cust-equip-pickup-notice
supply-activity-reqmnts-out
shop-stock-locator-list
shop-stock-list-cnstr-rep
repair-parts-form
shop-stock-list
manual-parts-requisition-out

row names	column names
1 documentation-identifier-code	1 parts-receipts
2 routing-identifier-code	2 work-request
3 media-and-status-code	3 part-number-changes
4 part-number-field	4 work-request-status-torn
5 unit-of-issue	5 intra-shop-work-order
6 transaction-quantity-received	6 work-order
7 activity-address-code	7 op-readiness-float-lter-qesc
	entity
	entity
	entity
	entity
	entity
	entity
	entity

Figure D-3. Sample LARE Consists Matrix (MPOM)

consists matrix report

row names		column names	
8 document-control-number	element	8 work-order-annotations	entity
9 demand-code	element	9 work-order-jacket	entity
10 supplementary-address-code	element	10 sample-data-collection-mrwo	input
11 signal-code	element	11 shipment-status-may-record	input
12 fund-code	element	12 shop-stock-adj-data-record-a	input
13 weapon-system-designator-code	element	13 shop-stock-adj-data-record-b	input
14 estimated-shipment-date	element	14 repair-part-mortality-data-rec	input
15 issue-priority-designator	element	15 prod-program-mgmt-uetrn-inp	input
16 transportation-control-number	element	16 parameter-report-control	input
17 joblotnu	*** undefined ***	17 snop-supply-list-changes	input
18 mode-of-shipment-code	element	18 snop-supply-list-adj-stmts	input
19 day-of-year-adj-for-shipmnt	element	19 soc-repair-parts-data	input
20 file-input-action-code	element	20 soc-work-order-annotations	input
21 card-designator-code-sams	element	21 soc-work-order-changes	input
22 funds-available-designator	element	22 ssl-adj-header-data-record	input
23 account-processing-field	element	23 float-candidate-rejects-input	input
24 project-code	element	24 supply-recon-data-record-an-x	input
25 condition-design-15-day-table	element	25 supply-recon-data-record-ap-x	input
26 condition-design-conus-location	element	26 supply-status-magnetic-record	input
27 identifying-number-code	element	27 table-build-data-rec-xm-ya	input
28 item-noun	element	28 table-build-data-rec-xm-yb	input
29 advice-code	element	29 table-build-data-rec-xm-yc	input
30 order-and-shipping-time-var	element	30 table-build-data-rec-xm-yd	input
31 source-maint-and-recovl-code	element	31 table-build-data-rec-xm-ye	input
32 storage-location-code	element	32 task-definitn-for-init-inspec	input
33 estimated-unit-part-cost	element	33 task-perf-tact-list-semiannual	input
34 unit-10-code-support-unit	element	34 task-performance-factor-list-upd	input
35 litnu	element	35 task-performance-factor-adj	input
36 identifying-number-code-part	element	36 technician-wo-annotations	input
37 part-number-field-part	element	37 tpf-exception-data-record	input
38 mortality-factor	element	38 personnel-data-changes	input
39 repair-parts-quantity-reqd	element	39 usage-device-component-change	input
40 total-estd-repair-parts-cost	element	40 received-parts-data	input
41 part-source-code	element	41 usage-input-data	input
42 taset1	element	42 wo-consumptn-data-record-labor	input
43 maint-control-number-sams	element	43 wo-consumptn-data-record-parts	input
44 file-identificatn-number-code	element	44 wo-consumptn-data-record-task	input
45 cond-design-reports-req	element	45 wo-maint-consumptn-data-wrksht	input

Figure D-3. Sample LARE Consists Matrix (MPOM) (Cont'd.)

consists matrix report

row names	column names
46 product-control-number	46 wo-parts-adj-data-recordo-xm-w input
47 required-data-source-code	47 wo-registration-data-record-a input
48 hours-expended	48 wo-registration-data-record-b input
49 hours-required	49 wo-reqmnts-data-rec-supl-parts input
50 hours-expended-per-task	50 wo-reqmnts-data-record-parts input
51 suffix-code	51 wo-reqmnts-data-record-tasks input
52 supplementary-address-field	52 wo-status-data-record input
53 jlico	53 wo-center-labor-data-rec-xm-1 input
54 recon-valid-cutoff-date-ordinal	54 parameter-duty-hours-rec input
55 supply-status-code	55 reconciliatn-response-card-rec input
56 estimated-delivery-date	56 parameter-previous-cycle-rec input
57 reply-due-date-ordinal	57 parts-status-data-record input
58 transaction-quantity-requested	58 parameter-norm-norm-rec input
59 transaction-day	59 bench-stock-listing-change-data input
60 source-of-sply-cd-rout-id-cd	60 annotaten-wo-copy-three input
61 unit-price	61 equipment-recall-require input
62 equipment-category-code	62 param-workload-backlog-age input
63 eqcase	63 float-lie-adjustment input
64 work-request-status-code	64 calibration-req-by-customer input
65 work-req-status-description	65 alt-sro-requirements input
66 order-and-shipping-time-ave	66 cross-reference-transaction input
67 ro-qnty	67 bench-stock-data-record-xm-pe input
68 rop-qnty	68 bench-stock-data-record-xm-py input
69 data-element-abbreviation	69 bench-stock-data-record-xm-pt input
70 work-center-code	70 bench-stock-data-record-xm-pd input
71 work-center-description	71 equip-recall-selling-list-regst input
72 date-prepared-ordinal	72 xm-a-decdbl-input input
73 unit-name-support	73 xm-b-decdbl-input input
74 item-nomen-item-noun-field	74 xm-c-decdbl-input input
75 item-nomen-item-noun-field	75 xm-cp-decdbl-input input
76 identifying-number-code-task	76 xm-cs-decdbl-input input
77 part-number-field-task	77 xm-ct-decdbl-input input
78 planneo-task-field	78 xm-d-decdbl-input input
79 tyaaac	79 xm-dl-decdbl-input input
80 std-mannours-by-cmnd-tenths	80 xm-gp-decdbl-input input
81 std-mannours-support-tenths	81 xm-gl-decdbl-input input
82 std-mannours-work-ctr-tenths	82 xm-e-decdbl-input input
83 ave-mh-expend-work-cent-tenths	83 xm-f-decdbl-input input

Figure D-3. Sample LARE Consists Matrix (MPOM) (Cont'd.)

consists matrix report

column names

row names

84 standard-mannhours-tenths	element	84 xm-g-dectbl-input	input
85 task-completed-status	element	85 xm-h-dectbl-input	input
86 avg-mannhours-exp-cmnd-tenths	element	86 xm-i-dectbl-input	input
87 avg-mn-exp-spt-ten	*** undefined ***	87 xm-n-dectbl-input	input
88 hourly-pay-rate	element	88 xm-pa-dectbl-input	input
89 eselocoou	element	89 xm-pu-dectbl-input	input
90 identifying-number-code-comp	element	90 xm-pc-dectbl-input	input
91 part-number-field-component	element	91 xm-pa-dectbl-input	input
92 comp-ser-nu-loc-cont-nu-field	element	92 xm-rap-dectbl-input	input
93 usage-at-removal	element	93 xm-s-dectbl-input	input
94 usage-recorded-when-installed	element	94 xm-t-dectbl-input	input
95 repair-date-ordinal	element	95 xm-u-xmv-jectbl-input	input
96 unit-id-code-customer-unit	element	96 xm-w-dectbl-input	input
97 usage-period-miles	element	97 xm-x-dectbl-input	input
98 usage-period-landings	element	98 xm-y-dectbl-input	input
99 usage-period-hours	element	99 xm-za-dectbl-input	input
100 usage-period-rounds	element	100 xm-zb-dectbl-input	input
101 uspreau	element	101 xm-zc-dectbl-input	input
102 work-order-number	group	102 xm-zo-dectbl-input	input
103 task-or-part-indicator-code	element	103 xa-ze-dectbl-input	input
104 task-sequence-field	element	104 xm-zf-dectbl-input	input
105 type-maint-action-completed	element	105 xa-zg-dectbl-input	input
106 employee-number-field	element	106 xm-zn-dectbl-input	input
107 ordinal-date	element	107 cost-data	input
108 labor-code-worked	element	108 float-adjustments	input
109 labor-distribution-code-suffix	element	109 equipment-recall-new-item-b	input
110 quantity-repaired	element	110 calibration-work-order	input
111 work-center-worked	element	111 parts-receipt-data-record	input
112 manhours-expended-tenths	element	112 maint-proj-data-record	input
113 assignment-code-change	element	113 manual-requisition-data-inp	input
114 teorpinco	*** undefined ***	114 part-number-change-data-rec	input
115 failure-code	element	115 parameter-parts-status-detail	input
116 quantity-consumed-in-maint	element	116 annotated-work-order-copy-tour	input
117 estimated	element	117 maint-proj-reqmts-data-record	input
118 component-serial-number	element	118 inspector-wa-annotations	input
119 equip-usage-measurement-code	element	119 equipment-recall-new-item-a	input
120 work-order-number-losing	element	120 calibration-require-by-item	input
121 task-sequence-field-losing	element	121 float-status-report-out	output

Figure D-3. Sample LARE Consists Matrix (MPOM) (Cont'd.)

consists matrix report

1	1111111112	2222222223	3333333334	4444444445	5555555556	6666666667	7777777778	8888888889	9999999990
1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890
1	1000	1	00000010	0	10000	0	0000000000	1	0000000000
2	1000	1	000	1	000	1	0000000000	1	0000000000
3	1000	1	000	1	000	1	0000000000	1	0000000000
4	1000	1	000	0	10000	00000	0000000000	1	0000000000
5	1000	1	000	0	1000	00000	0000000000	1	0000000000
6	1000	1	000	1	000	1	0000000000	1	0000000000
7	1000	1	000	1	000	1	0000000000	1	0000000000
8	1000	1	000	1	000	1	0000000000	1	0000000000
9	1000	1	000	1	000	1	0000000000	1	0000000000
10	1000	1	000	1	000	1	0000000000	1	0000000000
11	1000	1	000	1	000	1	0000000000	1	0000000000
12	1000	1	000	1	000	1	0000000000	1	0000000000
13	1000	1	000	1	000	1	0000000000	1	0000000000
14	1000	1	000	1	000	1	0000000000	1	0000000000
15	1000	1	000	1	000	1	0000000000	1	0000000000
16	1000	1	000	1	000	1	0000000000	1	0000000000
17	1000	1	000	1	000	1	0000000000	1	0000000000
18	1000	1	000	1	000	1	0000000000	1	0000000000
19	1000	1	000	1	000	1	0000000000	1	0000000000
20	1000	1	000	1	000	1	0000000000	1	0000000000
21	1000	1	000	1	000	1	0000000000	1	0000000000
22	1000	1	000	1	000	1	0000000000	1	0000000000
23	1000	1	000	1	000	1	0000000000	1	0000000000
24	1000	1	000	1	000	1	0000000000	1	0000000000
25	1000	1	000	1	000	1	0000000000	1	0000000000

Figure D-3. Sample LARE Consists Matrix (MPOM) (Cont'd.)

consists matrix report

1	1111111112	2222222223	3333333334	4444444445	5555555556	6666666667	7777777778	8888888889	9999999990
1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890
26	1	1	1	1	1	1	1	1	1
27	1	1	1	1	1	1	1	1	1
28	1	1	1	1	1	1	1	1	1
29	1	1	1	1	1	1	1	1	1
30	1	1	1	1	1	1	1	1	1
31	1	1	1	1	1	1	1	1	1
32	1	1	1	1	1	1	1	1	1
33	1	1	1	1	1	1	1	1	1
34	1	1	1	1	1	1	1	1	1
35	1	1	1	1	1	1	1	1	1
36	1	1	1	1	1	1	1	1	1
37	1	1	1	1	1	1	1	1	1
38	1	1	1	1	1	1	1	1	1
39	1	1	1	1	1	1	1	1	1
40	1	1	1	1	1	1	1	1	1
41	1	1	1	1	1	1	1	1	1
42	1	1	1	1	1	1	1	1	1
43	1	1	1	1	1	1	1	1	1
44	1	1	1	1	1	1	1	1	1
45	1	1	1	1	1	1	1	1	1
46	1	1	1	1	1	1	1	1	1
47	1	1	1	1	1	1	1	1	1
48	1	1	1	1	1	1	1	1	1
49	1	1	1	1	1	1	1	1	1
50	1	1	1	1	1	1	1	1	1

Figure D-3. Sample LARE Consists Matrix (MPOM) (Cont'd.)

Contents Report

Parameters for: cont

file noncila print-security-information noindex noindex levels=all

```

1* 1 boom-data-inputs (set)
1 2 *o-data-and-parts-record (input)
2 3 *o-data-task-section (group)
3 4 partial-work-order-number (element)
4 4 task-part-indicator-code (element)
5 4 task-sequence-field (element)
6 4 identifying-number-code-task (element)
7 4 part-number-field-task (element)
8 4 repair-part-nots-designator (element)
9 4 planned-task-field (element)
10 4 component-breakdown-code (element)
11 4 type-maintenance-action-blind (element)
12 4 quantity-to-be-repaired (element)
13 4 standard-manhours-tenths (element)
14 4 work-center-code (element)
15 4 type-maintenance-action-code (element)
16 4 quantity-repaired (element)
17 4 manhours-extended-tenths (element)
18 4 manhours-retaining-tenths (element)
19 4 standard-time-data-develop-tech (element)
20 4 equip-usage-measurement-code (element)
21 4 usage-recording-when-installed (element)
22 4 component-serial-number (element)
23 3 *o-data-parts-section (group)
24 4 partial-work-order-number (element)
25 4 task-part-indicator-code (element)
26 4 task-sequence-field (element)
27 4 identifying-number-code-part (element)
28 4 part-number-field-part (element)
29 4 repair-part-nots-designator (element)
30 4 item-noun (element)
31 4 failure-code (element)
32 4 repair-part-quantity-required (element)
33 4 part-source-code (element)
34 4 date-received-original (element)
35 4 transaction-quantity-issued (element)

```

Figure D-4. Sample LARE Contents Report (MPOM)

CONTENTS REPORT

36	4	transaction-quantity-requested (element)
37	4	est-indentment-date-ordinal (element)
38	4	transaction-quantity-queue-in (element)
39	4	supply-status-code (element)
40	4	condition-design-revelis-action (element)
41	4	condition-design-master-record (element)
42	4	activity-address-code (element)
43	4	document-control-number (element)
44	4	document-ident-code-supply-act (element)
45	4	date-precator-ordinal (element)
46	4	transportation-control-number (element)
47	4	over-all-ct-labeling-number (element)
48	4	estimated-ship-date-ordinal (element)
49	4	routing-identifier-code (element)
50	4	signal-code (element)
51	4	advice-code (element)
52	4	unit-of-issue (element)
53	4	control-code (element)
54	4	weapon-system-designator-code (element)
55	4	equipment-system-code (element)
56	4	issue-priority-designator (element)
57	4	protect-code (element)
58	4	accounting-processing-field (element)
59	4	quantity-consumption-ratino (element)
60	4	estimated-unit-parts-cost (element)
61	4	storage-location-code (element)
62	4	condesignator-increment-class (element)
63	4	carriage-number-previous (element)
64	4	task-sequence-field-previous (element)
65	4	condesignator-change-indic (element)
66	4	condesign-substitute-iter (element)
67	3	work-order-number (element)
68	4	work-order-number (element)
69	4	work-order-number (element)
70	4	end-iter-component-indic-field (element)
71	4	iter-component-main-field (element)
72	4	identifying-number-code (element)
73	4	part-number-field (element)
74	4	equip-serial-local-control-number-field (element)
75	4	equipment-system-code (element)
76	4	water-readiness-report-design (element)

Figure D-4. Sample LARE Contents Report (MPOM) (Cont'd.)

Contents Report

77	4	ort-transaction-code (element)
78	4	reap-sys-set-loc-cont-nur-flm (element)
79	4	part-no-nur-ret-ock-up-maint-act (element)
80	4	redistribution-serial-nurter (element)
81	4	equipment-model-ident-number (element)
82	4	federal-supply-code-manuf (element)
83	4	quantity-to-no-repairren (element)
84	4	inir-quantity-to-no-repairret (element)
85	4	usage-submiss-of-work-req-nil (element)
86	4	usage-submiss-ao-ren-landings (element)
87	4	usage-submiss-work-ren-hours (element)
88	4	usage-submiss-work-ren-rnds (element)
89	4	usage-submiss-work-ren-autotltns (element)
90	4	maintenance-repair-code (element)
91	4	calibration-level-code (element)
92	4	maintenance-level-code-unit (element)
93	4	type-maint-request-report-code (element)
94	4	recall-interval-code (element)
95	4	calibration-interval-code (element)
96	4	condition-destination-warranty (element)
97	4	calibration-type-standard-code (element)
98	4	failure-detected-during-code (element)
99	4	condition-code (element)
100	4	weapon-system-designator-code (element)
101	4	type-calibration-report-code (element)
102	4	equipment-utilization-field (element)
103	4	line-item-number (element)
104	4	equipment-category-code (element)
105	4	work-account-listed-code (element)
106	4	accounting-processing-field (element)
107	4	project-code (element)
108	4	maint-screen-start-date-ord (element)
109	4	maint-screen-end-date-ord (element)
110	4	on-hand-quantity-for-repair (element)
111	4	quantity-repairret (element)
112	4	maint-ortality-data-desig (element)
113	4	maint-ortality-later-tesio (element)
114	4	manhours-projected-tenths (element)
115	4	manhours-exceeded-tenths (element)
116	4	manhours-repairing-tenths (element)
117	4	material-disposition-code (element)

Figure D-4. Sample LARE Contents Report (MPOM) (Cont'd.)

contents report

118	4	condition-code-completion (element)
119	4	half-junction-description (element)
120	4	work-request-status-code-comb (element)
121	4	work-request-status-code (element)
122	4	ordinal-date (element)
123	4	military-time-of-day (element)
124	4	work-request-status-code-history (element)
125	4	ordinal-date-status-history (element)
126	4	military-time-of-day-status-hist (element)
127	4	equipment-location (element)
128	4	condition-designation-customer (element)
129	4	unit-ident-code-customer-unit (element)
130	4	partial-work-order-number (element)
131	4	eff-on-schedule (element)
132	4	date-accepted (element)
133	4	date-completed-ordinal (element)
134	4	unit-ident-code-support-unit (element)
135	4	unit-ident-code-customer-unit (element)
136	4	procurement-request-order-number (element)
137	2	weekly-workload-amount-aa-ca (input)
138	3	document-identification-code (element)
139	3	filler (element)
140	3	parameter-critical-part (element)
141	3	parameter-days (element)
142	3	parameter-days-range-field (element)
143	3	equipment-category-code (element)
144	2	weekly-workload-reports-cont-card (input)
145	3	document-identification-code (element)
146	3	filler (element)
147	3	condition-design-reports-requirement (element)
148	3	equipment-category-code (element)
149	2	transfer-file (input)
150	2	organ-inn-equin-regist-worksh (input)
151	3	document-identification-code (element)
152	3	partial-work-order-number-organization (element)
153	3	file-input-action-code (element)
154	3	work-request-status-code (element)
155	3	ordinal-date (element)
156	3	military-time-of-day (element)
157	3	identifying-number-code (element)
158	3	part-number-field (element)

Figure D-4. Sample LARE Contents Report (MPOM) (Cont'd.)

LOGICON LFXINGTON VAX SYSTEM

data process report

parameters for: dp

file process dmat dmat pmat dmat

the rows are data names, the columns are process names.

row names	column names
1 valid-transaction-file	1 maintain-support-plan
2 workload-status-listing	2 maintain-workload-data
3 completed-work-order-hist-file	3 maintain-ecc-equip-desc-file
4 work-order-data-and-parts-file	4 maintain-uc-unit-crossref-file
5 uc-unit-name-cross-ref-file	5 maint-cust-unit-priority
6 weekly-workload-mandant-pa-ca	6 store-xna-xnb-card-data
7 status-temp-file	7 maintain-valid-transaction-file
8 workload-status-list	8 purge-old-cworf-data
9 workload-age-listing-out	9 monitor-data-for-reports
	10 monitor-workload-status
	11 monitor-workload-age
	12 print-workload-reports
	13 print-workload-status-list
	14 print-workload-age-list
	15 provide-program-status

Figure D-5. Sample LARE Data Process Report (MPOM)

data process report

data process interaction matrix analysis

data

```

----
valid-transaction-file      (entity)      (row)  1) not derived by any process
workload-status-listing    (output)     (row)  2) not generated by any process
completed-order-history-file (entity)     (row)  3) not derived by any process
work-order-data-and-parts-file (entity)    (row)  4) not derived by any process
ulc-unit-name-cross-ref-file (entity)    (row)  5) not derived by any process
weekly-workload-management-ra-ca (input)     (row)  6) not received by any process
status-temp-file           (entity)     (row)  7) not derived by any process
workload-status-list       (output)     (row)  8) not derived by any process
workload-age-listing-out   (output)     (row)  9) not derived by any process

```

Processes

```

-----
maintain-support-plan      (column)  1) does not interact with any data
maintain-workload-data    (column)  2) does not interact with any data
maintain-ecg-equip-desc-file (column)  3) does not interact with any data
maint-ulc-unit-cross-ref-file (column)  4) does not interact with any data
maint-cust-unit-priority   (column)  5) does not interact with any data
store-vna-xnn-card-data    (column)  6) does not interact with any data
maintn-valid-transaction-file (column)  7) uses data, but does not derive or update anything
putoc-oid-cxort-data       (column)  8) does not interact with any data
monitor-data-for-reports   (column)  9) does not interact with any data
monitor-workload-age       (column) 10) uses data, but does not derive or update anything
print-workload-reports     (column) 11) uses data, but does not derive or update anything
print-workload-status-list (column) 12) does not interact with any data
provide-program-status     (column) 13) uses data, but does not derive or update anything
                           (column) 15) does not interact with any data

```

Figure D-5. Sample LARE Data Process Report (MPOM) (Cont'd.)

data process report

data process interaction matrix

(i,j) value meaning

r row i is received or used by column j (input)
u row i is updated by column j
d row i is derived or generated by column j (output)
a row i is input to, updated by, and output of column j (all)
f row i is input to and output of column j (flow)
1 row i is input to and updated by column j
2 row i is updated by and output of column j

	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure D-5. Sample LARE Data Process Report (MPOM) (Cont'd.)

CADPAT Version 3.2R2

LOGICOM LEXINGTON VAX SYSTEM

APR 24, 1981 19:54:14

page

4

data process report

process interaction matrix (incidence)

the rows and columns are process names from above.
an asterisk in (i,j) means that something derived
or updated by process i is used by process j.

*** matrix empty for rows 1 thru 15 and columns 1 thru 15

Figure D-5. Sample IARE Data Process Report (MPOM) (Cont'd.)

data process report

process interaction matrix analysis

maintain-support-plan	(row/col	1) no interaction, but has subparts and is part of a process
maintain-workload-data	(row/col	2) no interaction, but has subparts and is part of a process
maintain-acc-avail-desc-file	(row/col	3) no interaction, but is part of another process
maintn-lic-unit-crossref-file	(row/col	4) no interaction, but has subparts and is part of a process
maint-cust-unit-priority	(row/col	5) no interaction, but is part of another process
store-xna-xnm-card-data	(row/col	6) no interaction, but is part of another process
maintn-valid-transaction-file	(row/col	7) no interaction, but is part of another process
purge-old-cworf-data	(row/col	8) no interaction, but is part of another process
monitor-data-for-reports	(row/col	9) no interaction, but has subparts and is part of a process
monitor-workload-status	(row/col	10) no interaction, but is part of another process
monitor-workload-age	(row/col	11) no interaction, but is part of another process
print-workload-reports	(row/col	12) no interaction, but has subparts and is part of a process
print-workload-status-list	(row/col	13) no interaction, but is part of another process
print-workload-age-list	(row/col	14) no interaction, but is part of another process
provide-program-status	(row/col	15) no interaction, but is part of another process

Figure D-5. Sample LARE Data Process Report (MPOM) (Cont'd.)

LOGICON, LEXINGTON VAX SYSTEM

data process report

parameters for: dd

file data dmat dmat enat enat

the rows are data names, the columns are process names.

row names	column names
1 completed-work-order-hist-file entity	1 search-cvorf-data-for-rept process
2 evaluation-hold-file entity	2 noniter-workload-status process
3 inoperative-equivalent-file entity	3 process-sort-with-cvorf process
4 format-data-file entity	4 compute-turnaround-time-index process
5 ecc-turnaround-term-data entity	5 sort-data-by-criteria process
6 work-order-data-and-parts-file entity	6 ident-input-errors process
7 valid-transaction-file entity	7 print-sup-maint-turnover-ec process
8 ecc-equivalent-desc-file entity	8 generate-weekly-status-rept process
9 uic-unit-name-cross-ref-file entity	9 update-work-history-file process
10 status-term-file entity	10 valid-valid-transaction-file process
11 cvoh-outrec-term-hold entity	11 load-data-to-labor-tables process
12 temporary-hold-file entity	12 update-uic-crossref-file process
	13 print-workload-status-list process
	14 list-input-errors process

Figure D-5. Sample LARE Data Process Report (MPOM) (Cont'd.)

data process report

data process interaction matrix

(i,j)	value	meaning
r		row i is received or used by column j (input)
u		row i is updated by column j
d		row i is derived or generated by column j (output)
a		row i is input to, updated by, and output of column j (all)
f		row i is input to and output of column j (flow)
1		row i is input to and updated by column j
2		row i is updated by and output of column j

	1	2	3	4	5	6	7	8	9	10	11	12
1	r	r	r	i	i							
2	d			r	i							
3	i											
4												
5												
6												
7												
8												
9												
10												
11												
12												

Figure D-5. Sample LARE Data Process Report (MPOM) (Cont'd.)

data process report

data process interaction matrix analysis

data

```

-----
completed-work-order-hist-file      (entity) (row 1) not derived by any process
inneractive-environment-file        (entity) (row 3) not derived by any process
format-data-file                    (entity) (row 4) not derived by any process
work-order-data-and-parts-file      (entity) (row 6) not derived by any process
valid-transaction-file              (entity) (row 7) not derived by any process
ecc-equipment-desc-file              (entity) (row 8) not derived by any process
ulc-unit-name-cross-ref-file        (entity) (row 9) not derived by any process
cwhc-outrec-term-hold                (entity) (row 11) not derived by any process

```

processes

```

-----
nonint-workload-status              (column 2) uses data, but does not derive or update anything
process-sort-with-cvnmf              (column 3) uses data, but does not derive or update anything
print-sub-alint-turnover-ecc         (column 7) uses data, but does not derive or update anything
update-work-history-file             (column 9) uses data, but does not derive or update anything
valid-to-valid-transaction-file      (column 10) uses data, but does not derive or update anything
load-data-to-labor-tables            (column 11) uses data, but does not derive or update anything
print-workload-status-list           (column 13) uses data, but does not derive or update anything
list-input-errors                    (column 14) uses data, but does not derive or update anything

```

Figure D-5. Sample LARE Data Process Report (MPOM) (Cont'd.)

data process record

Process Interaction Matrix (Incidence)

the rows and columns are process names from above.
an asterisk in (i,j) means that something derived
or updated by process i is used by process j.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1													
2		1												
3			1											
4				1										
5					1									
6						1								
7							1							
8								1						
9									1					
10										1				
11											1			
12												1		
13													1	
14														1

Figure D-5. Sample LARE Data Process Report (MPOM) (Cont'd.)

data process report

process interaction matrix analysis

search-cvorf-data-for-rept	(row/col 1) no predecessors for this process
monitor-workload-status	(row/col 2) no successors for this process
process-sort-1ch-cvorf	(row/col 3) no interaction, but is part of another process
ident-input-errors	(row/col 6) no predecessors for this process
print-sup-maint-turnover-acc	(row/col 7) no predecessors for this process
generate-weekly-status-rept	(row/col 8) no predecessors for this process
update-co-nistrv-file	(row/col 9) no successors for this process
-maintn-valit-transaction-file	(row/col 10) no interaction, but is part of another process
load-data-to-lehor-tables	(row/col 11) no interaction, but is part of another process
print-workload-status-list	(row/col 13) no successors for this process
list-jrcut-errors	(row/col 14) no successors for this process

Figure D-5. Sample LARE Data Process Report (MPOM) (Cont'd.)

DATE 24 APR 81

REV-PERIOD 0

AIRMTCS PROGRAM

LOCATION MAINTENANCE PROGRAM OPERATIONS MANAGEMENT DATA BASE STATUS REPORT

REQ. NO.	RFOUTRFLHT NAME	SYNOMYM	TYPE	CAT	SEUN	STAT	NREV	SY	DS	SR	K*	TK	UD	RG	IT	UU	M4	OR
4.1	NOT FOUND IN DB																	
4.1.a	NOT FOUND IN DB																	
4.1.b	NOT FOUND IN DB																	
4.2	manage-support-plan	masp	PROC					1	1	2	0	0	00	00	00	00	0	4
4.2.a	develop-support-plan	desupl	PROC					1	1	1	0	0	00	00	00	00	0	2
4.2.a.1	maintain-support-plan	masupl	PROC					1	1	1	0	0	00	00	00	00	0	4
4.2.a.2	NOT FOUND IN DB																	
4.2.b	determine-workload-history	dewhl	PROC					1	1	1	0	0	00	00	00	00	0	1
4.2.c	maintain-workload-data	rawoda	PROC					1	1	1	0	0	00	00	00	00	0	6
4.2.d	edit-input-transactions	edintr	PROC					1	1	2	0	0	00	00	00	00	1	4
4.3	list-input-errors	linerr	PROC					1	1	2	0	0	10	00	30	00	1	1
4.3.a	ident-input-errors	idnerr	PROC					1	1	2	0	0	11	50	14	01	0	1
4.3.b	update-vch-history-file	upvohifi	PROC					1	1	1	0	0	39	00	14	01	0	1
4.3.c	status-test-file	statstfi	PROC					1	1	1	0	0	00	00	00	00	0	3
4.3.d	update-con-vch-hist-file	upcovohifi	PROC					1	1	4	0	0	00	00	00	00	0	1
4.3.e	generate-exclv-reports	gewere	PROC					1	1	1	0	0	00	00	00	00	0	4
4.3.f	control-report-generation	corene	PROC					1	1	3	0	0	00	00	00	00	0	3
4.3.g	update-uic-crossref-file	upulcrfi	PROC					1	1	1	0	0	20	00	12	02	0	2
4.3.h	update-ecc-equip-desc-file	upulcrfi	PROC					1	1	1	0	0	00	00	00	00	0	1
4.3.i	accept-ton-inputs	acmoain	PROC					1	1	2	0	0	00	00	00	00	0	1
4.3.j	accept-supply-status-files	acsustfi	PROC					1	1	2	0	0	00	00	00	00	0	1
4.3.k	accept-ton-work-torff	acmwotrp	PROC					1	1	1	0	0	00	00	00	10	0	1
4.3.l	accept-rklid-mgmt-ctrl-card	acxmcmca	PROC					1	1	2	0	0	00	00	10	10	1	1
4.3.m	update-con-vch-hist-file	upcovohifi	PROC					1	1	4	0	0	00	00	00	00	0	1
4.3.n	generate-monthly-reports	gemore	PROC					1	1	2	0	0	00	00	00	00	0	1
4.3.o	control-report-generation	corene	PROC					1	1	3	0	0	00	00	00	00	0	3
4.3.p	accept-rklid-mgmt-ctrl-card	acxmcmca	PROC					1	1	2	0	0	00	00	10	10	1	1
4.3.q	battalion-atg-310	hamadi	INTF					1	0	1	0	0	00	00	00	00	0	0
4.3.r	print-workload-status-list	prwostli	PROC					1	1	1	0	0	10	01	20	00	0	1
4.3.s	refl-for-workload-status	mwost	PROC					1	1	1	0	0	31	00	12	00	0	1
4.3.t	generate-exclv-status-rept	gewere	PROC					1	1	2	0	0	11	00	11	00	0	1
4.3.u	refl-for-workload-age	mwost	PROC					1	1	1	0	0	10	00	00	10	0	1
4.3.v	print-workload-age-list	prwostli	PROC					1	1	1	0	0	10	01	10	01	0	1
4.3.w	refl-for-repair-parts	mwost	PROC					1	1	1	0	0	00	00	00	00	0	1
4.3.x	print-critical-spares-parts-list	prcrspals	PROC					1	1	1	0	0	00	00	00	00	0	1
4.3.y	monitor-tops-availn-parts	mwost	PROC					1	1	1	0	0	00	00	00	00	0	1

Figure D-6. Sample LARE Data Base Report (MPOM)

LOCATION: MAINTENANCE PROGRAM OPERATIONS MANAGEMENT DATA BASE STATUS REPORT

[illegible]

Figure D-6. Sample LARE Data Base Report (MPOM) (Cont'd.)

LOGICAL MAINTENANCE PROGRAM OPERATIONS MANAGEMENT DATA BASE STATUS REPORT

REQ. NO.	REQUIREMENT NAME	SYNOPSIS	TYPE	CAT	SECM	STAT	MREV	SY	DS	SR	K	TK	UD	RG	TI	UU	MM	OR
4.5.e	print-ecm-magntic-transact-rpt	practerep	PROC			COMPL	0	1	1	2	0	0	00	01	10	00	0	1
	monitor-ecm-magntic-transact-rpt	practerep	PROC			COMPL	0	1	1	1	0	0	00	00	01	00	0	1
	control-report-transact-rpt	catrpt	PROC			COMPL	0	1	1	3	0	0	00	00	00	00	0	3
4.7	account-maint-eval-control-card	acraevcoch	PROC			COMPL	0	1	1	2	0	0	00	00	10	00	0	1
4.7.a	edit-input-transactions	editr	PROC			COMPL	0	1	1	2	0	0	00	00	00	00	1	4
	input-input-errors	linerr	PROC			COMPL	0	1	1	2	0	0	11	50	14	01	0	1
4.7.b	list-input-errors	linerr	PROC			COMPL	0	1	1	2	0	0	10	00	30	00	1	1
5.1	NOT FOUND IN DA																	
5.1.a	NOT FOUND IN DA																	
5.1.a.1	NOT FOUND IN DA																	
5.1.a.2	NOT FOUND IN DA																	
5.1.b	NOT FOUND IN DA																	
5.2	NOT FOUND IN DA																	
5.3	update-ctrl-vch-hist-file	uvcvchhfi	PROC			COMPL	0	1	1	4	0	0	00	00	00	00	0	1
5.3.a	store-xm-vch-hist-data	stxvchcda	PROC			COMPL	0	1	1	1	0	0	00	00	01	00	0	1
5.3.b	search-for-ecm-match	sefoecma	PROC			COMPL	0	1	1	1	0	0	00	00	00	10	0	1
	search-ctrl-files	se-ctrl	PROC			COMPL	0	1	1	1	0	0	00	00	00	00	0	5
5.3.c	update-ctrl-vch-hist-file	uvcvchhfi	PROC			COMPL	0	1	1	4	0	0	00	00	00	00	0	1
5.3.d	search-for-ecm	sefoecma	PROC			COMPL	0	1	1	1	0	0	00	00	00	10	0	1
5.3.e	search-for-acklog-rpt	sefoack	PROC			COMPL	0	1	1	1	0	0	00	00	00	00	0	1
5.3.f	search-for-status	sefoack	PROC			COMPL	0	1	1	2	0	0	00	00	00	00	0	1
5.3.g	NOT FOUND IN DA																	
5.3.h	NOT FOUND IN DA																	
5.4	maintain-incom-fulcrum-data	mainetda	PROC			COMPL	0	1	1	2	0	0	00	00	00	00	0	3
5.4.a	maint-valid-transact-file	mainetvfi	PROC			COMPL	0	1	1	4	0	0	10	00	01	00	0	1
	search-valid-transact-file	sevatvfi	PROC			COMPL	0	1	1	1	0	0	00	00	02	00	0	1
	create-new-rpts-records	crnrtare	PROC			COMPL	0	1	1	1	0	0	00	00	11	00	0	1
5.4.a.1	overlay-ecm-on-let	ovesonle	PROC			COMPL	0	1	1	1	0	0	00	00	00	00	1	1
5.4.a.2	sort-incom-status-data	soinstda	PROC			COMPL	0	1	1	1	0	0	00	00	00	00	0	1
5.4.b	sort-rpt-supply-status-file	sortasufsi	PROC			COMPL	0	1	1	1	0	0	00	00	00	00	0	1
5.4.c	add-tel-rpt-records-on-flag	addrtponfi	PROC			COMPL	0	1	1	1	0	0	00	00	01	00	0	1
5.4.d	sort-ctrl-vch-hist	soctrlvch	PROC			COMPL	0	1	1	1	0	0	00	00	11	00	0	1
5.4.e	calculate-labor-costs	calaco	PROC			COMPL	0	1	1	1	0	0	00	00	00	00	0	3
5.5	load-data-to-labor-costs	loadatolac	PROC			COMPL	0	1	1	1	0	0	00	00	00	00	0	1
5.5.a	maintn-valid-transact-file	mainetvfi	PROC			COMPL	0	1	1	1	0	0	50	00	01	00	0	1
5.5.b	NOT FOUND IN DA																	
5.5.c	NOT FOUND IN DA																	

Figure D-6. Sample LARE Data Base Report (MPOM) (Cont'd.)

LOGIC3W MAINTENANCE PROGRAM OPERATIONS MANAGEMENT DATA BASE STATUS REPORT

RFO. P.O.	REQUTRE-PC-T NAME	SYNONYM	TYPE	CAT	SEON	STAT	MREV	SY	DS	SR	NW	TK	UD	PG	IT	UU	MY	OR
5.5.d	print-maint-cost-summary	prmcosu	PROC			comp	0	1	1	2	0	0	00	01	10	00	0	1
5.6	print-support-cost-summary	prsuosu	PROC			comp	0	1	1	2	0	0	00	01	20	00	0	1
5.6.a	print-maint-evaluation-reports	prmaevr	PROC			comp	0	1	1	2	0	0	00	00	00	00	0	3
5.6.b	maintn-valid-transaction-file	mvatrfi	PROC			comp	0	1	1	4	0	0	10	00	01	00	0	1
	rev-entry-invoice-tran	vevnrnp	xxxx			comp	0	1	0	1	0	0	00	00	00	00	0	5
	report-mnt-date-ordinal	elem	elem			comp	0	0	0	1	0	0	00	00	00	00	0	3
	search-corr-date-for-rent	secrdatof	PROC			comp	0	1	0	1	0	0	31	10	11	00	0	1
5.6.c	report-start-date-ordinal	rept-str-date-o	elem			comp	0	1	1	6	0	0	00	00	00	00	1	9
	compute-turnat-unit-time-index	coctutlin	PROC			comp	0	1	1	2	0	0	10	00	12	00	0	2
	evaluation-crit-file	evncfl	PROC			comp	0	1	1	2	0	0	00	00	00	00	0	14
	sort-data-mv-criteria	sortabvcr	PROC			comp	0	1	1	2	0	0	21	00	11	00	0	1
5.6.d	print-sup-maint-turnover-unit	prsumatuin	PROC			comp	0	1	1	2	0	0	00	01	10	00	0	1
5.6.d.1	print-sup-maint-turnover-acc	prsumatuac	PROC			comp	0	1	1	2	0	0	10	01	10	00	0	1
5.6.d.2	print-unit-rent-rent-rent	prunmerenr	PROC			comp	0	1	1	2	0	0	00	01	10	00	0	1
5.6.d.3	print-ent-rent-rent-rent	prcnmerenr	PROC			comp	0	1	1	2	0	0	00	01	10	00	0	1
5.6.e	turne-cj-cvcrf-data	turnlcwda	PROC			comp	0	1	1	1	0	0	00	00	00	00	0	1
5.7	update-valid-transaction-file	upvatrfi	PROC			comp	0	1	1	4	0	0	10	00	01	00	0	1
5.7.a	maintn-valid-transaction-file	mvatrfi	PROC			comp	0	1	1	4	0	0	10	00	01	00	0	1
5.7.b	update-valid-transaction-file	upvatrfi	PROC			comp	0	1	1	2	0	0	00	00	10	00	0	1
5.7.c	NOT FOUND IN DB																	

Figure D-6. Sample LARE Data Base Report (MPOM) (Cont'd.)

AIRPLANE PROGRAM

DATE 24 APR 81

LOGICOM MAINTENANCE PROGRAM OPERATIONS MANAGEMENT DATA BASE SUMMARY REPORT

TYPES OF INFORMATION NO. REQUIRED NO. PRESENT NO. OMITTED

SYNOPSIS	100		
DESCRIPTIONS	100		
SOURCES	215		
KEYWORDS	0		
TRACE KEYS	113		114
USES	0		
DERIVATES	35		
RECEIVES	11		
GENERATES	14		
TRIGGEREDS	52		69
TRIGGERS	44		64
UTILIZES	6		
UTILIZEDS	6		
MEMOS	11		
OTHERS	109		

STATUS SUMMARY:

ADMIN	0
INCOL	0
INCON	0
INSEP	0
INTRN	0
COMI	114

REVISION SUMMARY:

RECENT	0
TOTAL	0

Figure D-6. Sample LARE Data Base Report (MPOM) (Cont'd.)

parameters for: tos

file noindex print empty noindex smarg=5 smarg=20 amarg=10 hmarq=25 rmarq=70 emarg=1 hmarq=40
 nodesignate one-per-line define comment none=page none=line noall-statements
 complementaty-statements line-numbers printeof

formatted problem statement

cross-ref-transaction;

1 input

2 synonyms are: error;

3 attributes are:

4 document-code-identifier

5 x-x,

6 frequency

7 media

8 consists of:

9 document-identification-code,

10 file-input-action-code,

11 carrier-designator-code-saws,

12 unit-ident-code-support-unit,

13 filler,

14 unit-name-support,

15 unit-ident-code-parent-unit,

16 unit-name-parent,

17 conflict-designator-rast-recrd,

18 activity-address-code,

19 accounting-processing-field,

20 unit-name-customer;

21 contained in: mom-tata-inputs;

22 used by: update-uic-crossref-file;

23 source is: n-a.13.05.ay;

24

supply-status-card-record;

25 input

26 synonyms are: sustcare;

27 see-me-n;

28 nt-proc-prior-to-as-all-cards;

29 attributes are:

30 frequency

31 document-code-identifier

32 weekly-200,

33 ac-;

34 received by: ident-input-errors;

35 consists of:

document-identification-code,

routing-identifier-code,

Figure D-7. Sample LARE Formatted Problem Statement (MPOM)

formatted problem statement

```

36 media-and-status-code,
37 part-number-field,
38 unit-of-issue,
39 transaction-quantity-requested,
40 activity-address-code,
41 document-control-number,
42 suffix-code,
43 supplementary-address-field,
44 signal-code,
45 fund-code,
46 distribution-code,
47 project-code,
48 issue-priority-designator,
49 transaction-day,
50 supply-status-code,
51 source-supply-code-route-id-cd,
52 estimated-shipment-date,
53 unit-office;
54 contained in: room-data-inputs;
55 source is: p-a.13.15.1r;
56
57 input key-entry-inputs=mpoom;
58 synonyms are: keyentry;
59 generated by: key-operator;
60 received by: accept-key-date-info,
61 search-conf-data-for-rent;
62 consists of:
63 report-end-date-ordinal,
64 report-start-date-ordinal;
65 contained in: room-data-inputs;
66 source is: p-5.6.p;
67
68 input month-fin-amount=rpt-ent-card;
69 synonyms are: mofimarcn;
70 attributes are:
71 account-code-identifier
72 xre,
73 frequency monthly,
74 volume-per-month
75 one-per-process;
76 consists of:

```

Figure D-7. Sample LARE Formatted Problem Statement (MPOM) (Cont'd.)

formatted problem statement

```

77 document-identification-code,
78 conn-iso-rent-rgmt,
79 filler:
80 contained in: mpom-data-inputs;
81 source is: p-a.13.72.1r;
82
83 input
84 generated by: key-crt-operator;
85
86 input
87 synonyms are: lnopstatf;
88 see-memo: pf-forwarded-to-mnom-from-mom,
89 pf-proc-prior-to-using-lnop-fl;
90 attributes are:
91 frequency daily-200,
92 document-code-identifier
93 xvj,
94 magnetic:
95 consists of:
96
97 document-identification-code,
98 cartial-work-order-number,
99 filler,
100 work-request-status-code,
101 ordinal-date,
102 military-tire-of-day,
103 identifying-number-code,
104 part-number-field,
105 malfunction-description,
106 date-accepted-ordinal,
107 issue-priority-designator,
108 maintenance-repair-code,
109 part-number-tckup-repair-act,
110 manhours-retaining-tenths,
111 equipment-category-code,
112 water-readiness-rept-desig,
113 activity-address-code,
114 file-noun,
115 part-source-code,
116 supply-status-code,
117 transaction-date-ordinal,
    transaction-quantity-required,

```

Figure D-7. Sample LARE Formatted Problem Statement (MPOM) (Cont'd.)

LOGICON LEXINGTON VAX SYSTEM

formatted problem statement

```

118      estimated-shipment-date-ordinal;
119 contained in: mpon-data-inputs;
120 source is: p-a.13.14.9d;
121
122 input
123   synonyms are: neworeoca;
124   see-memo: nt-process-before-normal-cycl;
125   attributes are:
126     frequency possible-l-per-cycle,
127     document-code-identifier
128     xnb;
129 received by: ident-input-errors;
130 consists of:
131   document-identification-code,
132   filler,
133   cond-desig-reports-requirement,
134   equipment-category-code;
135 contained in: mpon-data-inputs;
136 source is: p-a.13.04.1r;
137
138 input
139   synonyms are: shstcare;
140   see-memo: ds-proc-after-am-prior-au;
141   attributes are:
142     frequency weekly-500,
143     document-code-identifier
144     as-;
145 received by: ident-input-errors;
146 consists of:
147   document-identification-code,
148   routing-identifier-code,
149   media-ant-status-code,
150   part-number-field,
151   unit-of-issue,
152   transaction-quantity-requested,
153   activity-address-code,
154   document-control-number,
155   suffix-code,
156   supplementary-address-field,
157   signal-code,
158   fund-code,

```

Figure D-7. Sample LARE Formatted Problem Statement (MPOM) (Cont'd.)

formatted problem statement

```

200 supplementary-address-field,
201 signal-code,
202 func-code,
203 distribution-code,
204 estimated-shipment-date,
205 issue-priority-designator,
206 transportation-control-number,
207 mode-of-shipment,
208 date-shipnoed;
209 contained in: mcom-data-inputs;
210 source is: p-a.13.17.1r;
211
212 inout
213 synonyms are: sumaeirect;
214 attributes are:
215 document-code-identifier
216 xnf,
217 frequency
218 consists of:
219 document-identification-code,
220 filler,
221 cond-issg-rept-rcmt,
222 equipment-category-code;
223 contained in: mcom-data-inputs;
224 source is: p-a.13.31.1r;
225
226 inout
227 synonyms are: eccrtedate;
228 see-memo: pf-post-to-data-base-on-rcot,
229 of-proc-before-other-user-brcs;
230 attributes are:
231 document-code-identifier
232 xnc,
233 frequency as-required,
234 volume-per-month
235 %20,
236 media
237 consists of:
238 document-identification-code,
239 file-innut-action-code,
240 equipment-category-code,

```

Figure D-7. Sample LARE Formatted Problem Statement (MPOM) (Cont'd.)

formatted problem statement

```

241      equipment-category-description;
242      contained in:  mnom-data-inputs;
243      source is:    p-a.13.06.9v;
244
245      input
246      synonyms are:  wodaanlare;
247      attributes are:
248      frequency      weekly;
249      volume-per-month
250      consists of:  rec3100/cna170-rec500/cha500;
251
252      wo-data-task-section,
253      wo-data-parts-section,
254      wo-data-registration-section;
255      contained in:  mnom-data-inputs;
256      source is:    p-a.13.01.8a;
257
258      input
259      synonyms are:  orgnarewr;
260      see-also:     of-proc-prior-to-xxx;
261      attributes are:
262      document-code-identifier
263      xrf,
264      frequency      weekly=1000;
265      consists of:
266
267      document-identification-code,
268      partial-wo-number-organization,
269      file-input-action-code,
270      work-request-status-code,
271      ordinal-rate,
272      military-time-of-day,
273      identifying-number-code,
274      part-number-field,
275      iter-number,
276      equip-ser-icl-con-no-fld,
277      maintenance-repair-code,
278      reason-innerative,
279      partial-work-order-number,
280      equipment-category-code,
281      rate-readiness-rent-desla;
282      contained in:  mnom-data-inputs;

```

Figure D-7. Sample LARE Formatted Problem Statement (MPOM) (Cont'd.)

formatted problem statement

282 source is: p-a.13.11.4d;
283
284 input supply-status=file-record;

285 synonyms are: sustifire;

286 attributes are:

287 frequency weekly=2000-4000,
288 media magnetic;

289 consists of:

290 document-identification-code,
291 routing-identifier-code,
292 media-and-status-code,
293 part-number-field,
294 unit-of-issue,
295 transaction-quantity-requested,
296 activity-address-code,
297 document-control-number,
298 estimate-employment-date,
299 suffix-code,
300 supplementaty-address-field,
301 signal-code,
302 func-code,
303 distribution-code,
304 project-code,
305 issue-priority-designator,
306 transaction-day,
307 supply-status-code,
308 source-supply-code-route-id-cd,
309 unit-office,
310 transportation-control-number,
311 mode-of-shipment,
312 date-shipped;
313 contained in: mpom-data-inputs;
314 source is: p-a.13.14.4v;

315 input org-internal-parts-data=#k-rcd;

316 synonyms are: original;

317 see-to: cf-process-after-xr1-ins;

318 attributes are:

319 document-code-identifier
320 xxx,

321 frequency daily=200;

Figure D-7. Sample LARE Formatted Problem Statement (MPOM) (Cont'd.)

formatted problem statement

```

323 consists of:
324 document-identification-code,
325 partial-wj-number-organization,
326 file-input-action-code,
327 activity-address-code,
328 document-control-number,
329 item-noun,
330 identifying-number-code,
331 part-number-field,
332 part-source-code,
333 issue-priority-designator,
334 supply-status-code,
335 transaction-date-ordinal,
336 transaction-quantity-required,
337 estimated-shipment-date-ordinal,
338 filler;
339 contained in: woom-data-inputs;
340 source is: p-a.13.12.1d;
341
342 input direct-indirect-lab-ofn-crd;
343 synonyms are: dlinlacvot;
344 attributes are:
345 document-code-identifier
346 xnd,
347 volume-per-month
348 varies;
349
350 consists of:
351 document-identification-code,
352 filler,
353 equipment-category-code,
354 hourly-rate,
355 indirect-on-hourly-rate-field,
356 reserve,
357 other-costs;
358 contained in: woom-data-inputs;
359 source is: p-a.13.21.1r;
360
361 input weekly-workload-maintenance-pa-ca;
362 synonyms are: wacacaca;
363 seen-ter: of-proc-prior-to-prod-cycles;
  
```

Figure D-7. Sample LARE Formatted Problem Statement (MPOM) (Cont'd.)

formatted problem statement

```

364      document-code-identifier
365      xra,
366      frequency weekly-1;
367      received by: ident-input-errors;
368      consists of:
369          document-identification-code,
370          filler,
371          parameter-critical-part,
372          parameter-navs,
373          parameter-days-range-field,
374          equipment-category-code;
375      contained in: mnom-data-inputs;
376      used by: monitor-workload-ade;
377      source is: p-a.13.03.1r;
378
379      input
380      synonyms are: orinrededa;
381      see-me-o: of-process-prior-to-xmk;
382      attributes are:
383          frequency daily-2u0,
384          document-code-identifier
385          xmf;
386      consists of:
387          document-identification-code,
388          partial-work-order-number,
389          file-input-action-code,
390          work-request-status-code,
391          ordinal-date,
392          military-time-of-day,
393          identifying-number-code,
394          part-number-field,
395          item-noun,
396          equip-serial-local-cont-no-fld,
397          equip-ser-icl-con-no-fld,
398          maintenance-repair-code,
399          reason-inoperative,
400          equipment-category-code,
401          water-readiness-rent-desig;
402      contained in: mnom-data-inputs;
403      source is: p-a.13.13.4h;
404

```

Figure D-7. Sample LARE Formatted Problem Statement (MPOM) (Cont'd.)

name list

name	type	synonym	paragraph
67 assignment-change-code	element	asccch	m-a.12.04.kz.fld.16
68 assignment-code-change	element	atrza-dntv-orf	m-a.12.40.kv.fld.7
69 author-quant-on-readiness-flt	element	auquobrefl	m-b.02.10.4v.fld.6
70 autorth-usa-at-sbms-of-wrk-reg	element	auusatbbof	m-a.12.02.kz.fld.14
71 ave-mh-expand-work-cent-tenths	element	use-sbm-wrk-reg-autortns	m-a.12.02.kz.fld.14
72 average-monthly-issued-ssl	element	avg-mh-expd-wrk-cent-ten	m-a.12.62.4v.fld.15
73 avg-manhours-exp-cmnd-tenths	element	avmaxxcoe	m-a.12.62.4v.fld.15
74 avg-manhours-exp-supn-tenths	element	avg-mo-isd-ssl	m-b.02.41.4v.fld.21
75 back-status-sub-equip-cat-reg	output	avmoiss	m-b.02.39.4m.fld.23a
76 back-up-maintenance-activity	interface	avmaexcote	m-b.02.41.4v.fld.9
77 backlog-one-by-equip-cat-reg	output	avg-mh-expd-cmnd-ten	m-a.12.63.4m.fld.11
78 backlog-status-subsection-ren	output	bestsuedca	m-a.12.63.4m.fld.11
79 backlog-maintenance-activity	interface	baupmaac	m-b.02.03.4d
80 battalion-management	interface	bna	m-b.02.03.4d
81 battalion-wateriel-office	interface	baenbyeoca	m-b.02.03.4d
82 bench-stock-adjustment-data	set	bastsure	m-4.20.a
83 bench-stock-data-record-xm-bd	input	hama	
84 bench-stock-data-record-xm-be	input	bestad	m-a.12.17.kv
85 bench-stock-data-record-xm-bf	input	bestdatexmnd	
86 bench-stock-data-record-xm-bg	input	bestdatexmne	
87 bench-stock-list	set	bestdatexmof	
88 bench-stock-list-report	output	bestdatexmop	
89 bench-stock-listing-file	set	bestli	m-b.02.42.4v
90 bench-stock-listing-change-data	input	bestlire	m-4.12.a
91 bng-control-activity	interface	bestlixi	m-4.12.a
92 cadecoca	*** undefined ***	bestlichda	
93 calbr-cd	*** undefined ***	brcoac	
94 calibration-interval-code	element	calnco	m-a.12.31.4n.fld.13
95 calibration-level-code	element	calbr-intv-l-cd	m-a.12.32.4q.fld.14
		calnr-lvl-cd	m-a.12.31.4n.fld.12
		caleco	m-a.12.32.4n.fld.13

Figure D-8. Sample LARE Name List (MOM)

name list		SYNONYM	paragraph
name	type		
96 calibration-reg-bv-customer	input	carebycu	
97 calibration-required-bv-item	input	carebyit	
98 calibration-required-bv-data	set	carebda	m-8.12.31.4q
99 calibration-reg-clerk	interface	cat-cl	
100 calibration-tear-levels	interface	catele	
101 calibration-technician	interface	cate	
102 calibration-type-standard-code	element	calbr-type-std-cd	
103 calibration-work-center	interface	catvstco	m-4.14.q.3
104 calibration-work-order	input	cawoce	m-14.h.1
105 car4-designator-code-sams	element	cawoor	m-a.12.97.kv.f1d.2 m-a.12.98.kv.f1d.3 m-a.12.30.kv.f1d.3 m-a.12.98.f1d.3 m-a.12.96.kv.f1d.2 m-a.12.99.kv.f1d.3 m-a.12.13.kz.f1d.2 m-a.12.17.kv.f1d.3 m-5.10.a.1 m-5.10.e m-5.10.d.2 m-5.8.j m-5.8.d m-5.8.e
106 change-bench-stock-list	process	cadecosa	
		card-dso-cd-sams	
107 change-equip-recall-data	process	chbestli	
108 change-float-authorization	process	chedreda	
109 change-indication-code	*** undefined ***	chflau	
110 change-maint-prod-prgm-data	process	chwaprpra	m-5.8.f
111 change-part-number	process	chpenu	m-5.8.1
112 change-personnel-assignments	process	crpeas	m-5.8.h m-5.10.e m-5.10.d.2 m-5.10.a m-5.8.j m-5.8.h m-5.8.1
113 change-shop-stock-list	process		
114 change-toe-tde-authorization	process	chmstli	
115 change-work-request-status	process	chtotdau	
116 characters-per-line	*** undefined ***	ch-o-rest	
117 chd-prt-72-fid	*** undefined ***	chocell	
118 close-out-40	process	clouuo	
119 close-supply-transactns-rot	output	clutrrfp	m-b.02.37.4w
120 closed-work-order-register	set	clwoorre	m-4.11.s

name list						
name	type		synonym	paragraph		
121	comp-ser-nu-loc-cont-nu-field	element	comp-ser-1cl-con-no-fld cosenu	m-a.12.51.kv.fld.7		
122	compare-float-calc-params	process	cosenu	m-a.12.51.kv.fld.7		
123	component-breakdown-code	element	codechln codechln	m-a.12.03.kz.fld.7		
124	component-serial-number	element	codechln	m-a.12.03.kz.fld.7		
125	con-no-fld	*** undefined ***	codechln	m-a.12.13.kz.fld.18		
126	cond-desig-chance-indicator	element	cond-dsq-ch-indic	m-a.12.13.kz.fld.18		
127	cond-desig-master-record	element	codechln	m-a.12.99.kv.fld.8		
128	cond-desig-reimbursable-cust	element	codechln	m-a.12.99.kv.fld.8		
129	cond-desig-reports-reg	element	codechln	m-a.12.01.kz.fld.15		
130	cond-desig-substitute-item	element	codechln	m-a.12.01.kz.fld.15		
131	cond-designator-results-action	element	codechln	m-a.12.98.kv.fld.5		
132	cond-dsq-rep-rprt	*** undefined ***	codechln	m-a.12.13.kz.fld.8		
133	cond-dsq-rpt-rprt	*** undefined ***	codechln	m-a.12.13.kz.fld.8		
134	cond-dsq-rpt-act-code	element	codechln	m-a.12.03.kz.fld.6		
135	condition-code	element	codechln	m-a.12.03.kz.fld.6		
136	condition-desig-15-day-table	element	codechln	m-a.12.17.kv.fld.9		
137	condition-desig-all-records	element	codechln	m-a.12.96.kv.fld.4		
138	condition-desig-census-location	element	codechln	m-a.12.17.kv.fld.21		
139	condition-designator-code	element	codechln	m-a.12.17.kv.fld.21		
140	condition-designator-warranty	element	codechln	m-a.12.17.kv.fld.10		
141	condition-dsq-req-action	element	codechln	m-a.12.17.kv.fld.9		
142	condition-master-record	element	codechln	m-a.12.17.kv.fld.9		
143	condlth-desig-document-history	element	codechln	m-a.12.02.kz.fld.18		
			codechln	m-a.12.07.kv.fld.18		
			codechln	m-b.02.36.4v.fld.9		
			codechln	m-a.12.13.kz.fld.12		
			codechln	m-a.12.13.kz.fld.12		

Figure D-8. Sample LARE Name List (MOM) (Cont'd.)

name list				
name	type	synonym	paradigm	
144 control-float	process			
145 cost-data	input	cofi		
146 crd-dst	*** undefined ***	coda	m-4.10.o.5	
147 cross-ref	*** undefined ***			
148 cross-reference-transaction	input	cretr	m-a.12.99.kv	
149 cross-reference-transaction-data	set	cretrda	m-4.10.v	
150 cust-equipo-pickup-notice	output		m-4.10.u	
			m-4.10.t	
			m-4.14.c	
151 customer	interface	cuepino		
152 customer-maint-control	interface	cu		
153 customer-related-forms-repts	set	curaco		
154 customer-work-order-reconcil	set	curefore		
155 date-of-vr-aval-shout	element	cuvooire		
156 daps-file	set	dafi	m-b.02.06.4w	
157 daily	attribute-value		m-b.02.35.4d.fld.67	
158 daily-supply-transactions-data	set	dasutda	m-4.11.o	
159 daily-supply-transactions-rept	output			
160 data-element-abbreviation	element	dasuttre	m-b.02.33.4d	
161 date-accepted-ordinal	element	daelab	m-4.12.f.10	
		de-abbr	m-4.12.f.1	
162 date-asad-ord	element	daacor	m-a.12.96.kv.fld.5	
163 date-assigned-ordinal	element	date-acpt-ord	m-a.12.96.kv.fld.5	
164 date-completed-ordinal	element	daasor	m-b.02.02.41.fld.41	
		date-ason-ord	m-b.02.30.4w.fld.18	
165 date-of-receipt-ordinal	element		m-b.02.02.4d.fld.10	
166 date-pre-ord	*** undefined ***		m-b.02.02.4d.fld.26	
167 date-ordered-ordinal	element	dacoor	m-b.02.57.Rw.fld.4	
		data-comol-ord	m-b.02.51.4r.fld.15	
		daofreor	m-a.12.70.kv.fld.15	
		date-rec-ord	m-a.12.70.kv.fld.15	
			m-b.02.09.4w.fld.10	
			m-b.02.09.4w.fld.11	
			m-b.02.05.5d.fld.9	
			m-b.02.07.4m.fld.14	
			m-b.02.04.4w.fld.9	
			m-b.02.09.4w.fld.32	
			m-b.02.33.4v.fld.1	
			m-a.12.52.8r.fld.12	
			m-a.12.50.kr.fld.12	

Figure D-8. Sample LARE Name List (MOM) (Cont'd.)

name	type	name list	synonym	paragraph
168	date-start-ord			m-b.02.06.4w.fld.1
169	date-status-ordinal			m-a.12.61.4m.fld.1
				m-b.02.20.4r.fld.1
				m-b.02.34.4y.fld.1
				m-b.02.11.4y.fld.1
				m-b.02.08.4m.fld.1
				m-b.02.07.4m.fld.1
				m-b.02.23.4m.fld.1
				m-b.02.22.4m.fld.1
				m-b.02.02.4d.fld.1
				m-b.02.40.4r.fld.1
				m-b.02.36.4w.fld.1
				m-b.02.35.4d.fld.1
				m-b.02.10.4v.fld.1
				m-b.02.30.4w.fld.29
				m-b.02.30.4w.fld.1
				m-b.02.50.4w.fld.1
				m-b.02.47.4v.fld.1
				m-b.02.41.4v.fld.1
				m-b.02.39.4m.fld.1
				m-b.02.38.4v.fld.1
				m-b.02.36.4w.fld.23
				m-b.02.21.4w.fld.1
				m-b.02.37.4w.fld.1
				m-b.02.32.4d.fld.34
				m-b.02.32.4d.fld.5
				m-b.02.04.4w.fld.1
				m-b.02.01.4d.fld.1
				m-a.12.62.4v.fld.1
				m-b.02.31.4d.fld.1
				m-b.02.05.5d.fld.1
				m-b.02.09.4w.fld.1
				m-b.02.51.4r.fld.1
				m-a.12.63.4m.fld.1
				m-a.12.31.4d.fld.1
				m-a.12.32.4q.fld.1
				m-b.02.09.4w.fld.8
				m-b.01.01.4d.fld.5
				m-b.01.01.4d.fld.5

dapor
 date-prep-ord
 dastor
 date-sta-ord

element
 element

Figure D-8. Sample LARE Name List (MOM) (Cont'd.)

	name	type	name list	synonym	paragraph
170	day-of-year-avail-for-shipment	element		day-of-yr-avail-shipment	m-a.12.16.8d.f1d.18
171	days-awaiting-oarts	element		dayofveavfo	m-a.12.16.8d.f1d.18
172	delete-lookup-table-data	process		da-awto-prts	m-b.02.30.4w.f1d.20
173	demand-code	element		daawpa	m-b.02.30.4w.f1d.20
				delntada	m-4.10.tt.2
					m-a.12.15.8d.f1d.9
					m-b.02.35.4d.f1d.47
					m-a.12.17.kv.f1d.1A
					m-b.02.42.4v.f1d.17
					m-a.12.16.8d.f1d.9
					m-b.02.35.4d.f1d.30
					m-b.02.35.4d.f1d.12
					m-a.12.17.kv.f1d.17
					m-b.02.35.4d.f1d.60
					m-a.12.03.kz.f1d.1A
					m-a.12.08.8m.f1d.16
					m-4.14.e
174	determine-required-calibration	process		deco	
175	determine-required-maint	process		dmd-cd	
176	determine-required-maint-service	process		deteca	m-4.14.e
177	distribute-trf-data	process		detera	m-4.18.b
178	distribution-code	element		detemase	
				dltoda	
				dico	
179	document-code-identifier	attribute		distr-ca	m-a.12.15.8d.f1d.8
180	document-control-number	element		docoid	m-a.12.16.8m.f1d.7
					m-a.12.16.8d.f1d.8
					m-b.02.31.4d.f1d.7
					m-b.02.30.4w.f1d.24
					m-b.01.02.4d.f1d.5
					m-b.02.08.4w.f1d.25
					m-b.02.32.4d.f1d.21
					m-b.02.33.4v.f1d.12
					m-b.02.32.4d.f1d.31
					m-b.02.35.4d.f1d.46
					m-b.02.35.4d.f1d.29
					m-b.02.35.4d.f1d.11
					m-b.02.34.4v.f1d.5
					m-b.02.35.4d.f1d.59
					m-b.02.36.4w.f1d.5

Figure D-8. Sample LARE Name List (MOM) (Cont'd.)

name list			
name	type	synonym	parengraph
181 document-id-code-supply-action	element		m-b.02.35.4d.f1d.68
182 document-register-data-out	Set		m-b.02.37.4w.f1d.5
183 documentation-identifier-code	*** undefined ***		m-a.12.03.kz.f1d.14
184 documentation-identifier-code	element		m-a.12.13.kz.f1d.5
			m-a.12.13.kz.f1d.17
			m-b.02.36.4w.f1d.16
			m-b.01.01.4d.f1d.1
			m-a.12.99.kv.f1d.1
			m-a.12.15.8d.f1d.1
			m-a.12.03.kz.f1d.1
			m-a.12.02.kz.f1d.1
			m-b.02.35.4d.f1d.23
			m-a.12.04.kz.f1d.1
			m-a.12.06.kv.f1d.1
			m-a.12.05.kz.f1d.1
			m-a.12.40.kv.f1d.1
			m-a.12.12.kv.f1d.1
			m-a.12.13.kz.f1d.9
			m-a.12.13.kz.f1d.1
			m-a.12.18.8m.f1d.1
			m-b.01.02.4d.f1d.1
			m-b.02.35.4d.f1d.54
			m-a.12.20.kr.f1d.1
			m-a.12.17.kv.f1d.1
			m-a.12.16.8d.f1d.1
			m-a.12.98.f1d.1
			m-b.02.35.4d.f1d.5
			m-a.m-a.12.17.kv.f1d
			m-a.12.97.kv.f1d.1
			m-a.12.98.kv.f1d.1
			m-b.02.35.4d.f1d.41
			m-a.12.06.kv.f1d.1
			m-a.12.30.kv.f1d.1
			m-a.12.70.kv.f1d.1
			m-a.12.50.kr.f1d.1
			m-a.12.60.kv.f1d.1
			m-a.12.51.kv.f1d.1

doconu
docu-con-no
dic-sup-act
doidcosnac
doredaou

Figure D-8. Sample LARE Name List (MOM) (Cont'd.)

name	name list	type	synonym	paragraph
185	doidcosu	element		
186	ds-data-missing-oq-a44	memo		m-4.10.g
187	ds-maintenance	interface	dic	m-a.12.01.kz.fid.1
188	ds-shop-operator-makes-repairs	memo	doidco	m-a.12.13.kz.fid.14
189	edit-ord	element		
190	edit-transactions	process	dama	m-4.10.11
191	employee-number-field	element	dsshonnare	m-b.02.36.4w.fid.22
			entr	m-5.8.p
			ennufi	m-b.02.52.8w.fid.1
			enol-no-fid	m-b.02.51.4r.fid.8
192	end-item-component-indic-field	element		m-a.12.04.kz.fid.7
				m-a.12.70.kv.fid.8
				m-a.12.30.kv.fid.14
				m-a.12.33.8m.fid.4
				m-a.12.01.kz.fid.9
193	end-item-national-stock-number	element	end-item-comp-ind-fid	m-a.12.07.8m.fid.16
			enitcolnfi	m-b.01.01.4d.fid.7
			el-nsn	m-4.17.a
			enitnastnu	m-4.15.b
194	ent-assigned-work-order-number	process	enaswoornu	m-4.17.j
			enprormode	
195	ent-prog-program-mgmt-detrn	process	enalstrda	
196	enter-alt-sro-data	process	enbschrda	m-4.12.a
197	enter-bslf-change-data	process	enclinfoda	m-4.10.c.5
198	enter-class-inspecn-form-data	process	endafora	m-4.12.e
199	enter-da-form-data	process	enegaufa	m-4.10.1j
200	enter-enunpment-usage-data	process	enflad	m-4.13.1
201	enter-float-adjustments	process	enflch	m-5.11.f
202	enter-float-changes	process	enlnwoda	
203	enter-initial-xo-data	process	enlausda	m-5.8.c.1
204	enter-labor-usage-data	process	enlotada	m-5.8.d
205	enter-lockup-table-data	process	enncada	m-5.8.f
206	enter-parameter-data	process	enpanuch	m-4.10.rr
207	enter-part-number-change	process	enp10.c.1	m-5.10.c.1
208	enter-parts-receipt-data	process	enpareda	m-5.8.k
209	enter-parts-status-data	process	enp5.10.c.2	m-5.10.c.2
210	enter-parts-usage-data	process	enpastda	m-5.p.k
211	enter-personnel-data-channels	process	enp5.8.c.2	m-5.8.c.2
			enpedach	m-4.10.gg

Name list

	name	type	synonym	paragraph
212	enter-production-wo-data	process	enprwoda	m-5.10.c.3
213	enter-reconciliation-parts-data	process	entrepada	m-5.8.k
214	enter-registration-wo-data	process	entewoda	m-5.8.a
215	enter-repair-parts-form-data	process	entepafoda	m-4.12.b
216	enter-shop-supply-data	process	enssuda	m-4.12.f.7
217	enter-standard-wo-data	process	entstoda	m-5.8.c
218	enter-task-completion-data	process	entacoda	m-5.8.m
219	enter-task-perf-factor-atl	process	entapefaad	m-5.9.n
220	enter-usage-data	process	enusda	m-4.10.q
221	enter-wo-completion-data	process	enwocoda	m-4.10.r
222	enter-wo-availability-data	process	enwoevda	m-4.10.s
223	enter-wo-parts-reassignments	process	enwopara	m-5.10.b
224	enter-wo-requirerevents-data	process	enworeda	m-5.8.b
225	enter-wo-supplymntal-parts-data	process	enwosupada	m-4.12.k
226	enter-wo-update-data	process	enwoudada	m-4.12.h
227	enter-work-order-parts-data	process	enwoordada	m-4.12.d
228	enter-work-order-status-update	process		m-4.10.aa
229	enter-work-order-task-data	process	enwoorstun	m-4.10.bb
230	enter-work-order-time-expanded	process	enwoorstun	m-4.10.w.3
231	eqarenelta	input	enwoortada	m-4.10.x.5
232	equlio-level-maint-all-chrt-out	output	enwoortlex	m-4.10.g
233	equlio-orar-ready-net-change	element	eqlemalchou	m-4.10.q
234	equlio-recall-delinq-list-rent	input	eqorenech	m-4.7.c.1
235	equlio-recall-delinq-report	output	eqredelire	m-4.14.m
236	equlio-recall-delinq-ency-list	set	eqredere	m-b.02.23.4m
237	equlio-ser-icl	*** undefined ***	eqredell	m-4.14.m
238	equlio-ser-icl-con-no-field	element		m-b.02.06.4w.fid.8

Figure D-8. Sample LARE Name List (MOM) (Cont'd.)

name list		name list		name list	
name	type	name	type	name	type
239 equip-ser-lo-cont-no-flid	element	equip-ser-lcl-con-no-flid	equip-ser-lcl-con-no-flid	m-b.02.23.4m.fld.8	Paragraph
240 equip-ser-lcl-con-no-flid	element	eqseloconu	eqseloconu	m-b.02.22.4m.fld.8	
241 equip-set-lcl-con-no-flid	*** undefined ***	equip-use-meas-cd	equip-use-meas-cd	m-b.01.01.4d.fld.9	
242 equip-usage-measurement-code	element	equsmeco	equsmeco	m-b.02.04.4w.fld.12	
243 equipment-category-code	element			m-a.12.04.kz.fld.13	
				m-a.12.04.kz.fld.13	
				m-b.01.02.4d.fld.12	
				m-a.12.96.kv.fld.4	
				m-a.12.33.8m.fld.16	
				m-h.02.01.4d.fld.4	
				m-b.02.01.4d.fld.44	
				m-b.02.01.4d.fld.24	
				m-b.02.30.4w.fld.8	
				m-b.02.03.4d.fld.22	
				m-b.02.03.4d.fld.4	
				m-b.02.03.4d.fld.51	
				m-a.12.30.kv.fld.14	
				m-a.12.02.kz.fld.8	
				m-a.12.07.8m.fld.24	
244 equipment-category-description	element	ecc	ecc	m-b.02.01.4d.fld.37	
245 equipment-category-descriptions	element	exaco	exaco	m-b.02.03.4d.fld.52	
		exacate	exacate	m-b.02.03.4d.fld.23	
				m-b.02.03.4d.fld.5	
				m-b.02.01.4d.fld.58	
				m-b.02.30.4w.fld.9	
246 equipment-location	element	equip-cat-descr	equip-cat-descr	m-b.02.01.4d.fld.17	
		exlo	exlo	m-b.02.22.4m.fld.12	
		equip-loc	equip-loc	m-a.12.30.kv.fld.11	
247 equipment-model-ident-number	element			m-a.12.33.8m.fld.18	
248 equipment-model-identif-no	element			m-b.01.01.4d.fld.8	
				m-a.12.32.4a.fld.10	
				m-a.12.33.8m.fld.10	
				m-a.12.30.kv.fld.9	
249 equipment-operator	interface	eqmoldnu	eqmoldnu	m-a.12.31.4a.fld.10	
		equip-modl-ident-no	equip-modl-ident-no		
		eqoo	eqoo		

Figure D-8. Sample LARE Name List (MOM) (Cont'd.)

name list			
name	equipment-readiness-code	type	element
250	equipment-readiness-code	element	
251	equipment-recall-data	set	
252	equipment-recall-data-xx	set	
253	equipment-recall-deliv-1st	output	
254	equipment-recall-deliv-out	set	
255	equipment-recall-new-item-a	input	
256	equipment-recall-new-item-b	input	
257	equipment-recall-require	input	
258	equipment-recall-schedule	set	
259	equipment-recall-schedule-out	output	
260	equipment-recall-schedule-rep	output	
261	equipment-serial-number	element	
262	equipment-svstem-	system-parameter	
263	equipment-svstem-code	element	
264	equipment-utilization-code	element	
265	equipment-utilization-field	element	
266	err-msg	element	
267	error-exception-report-supply	output	
268	error-exception-report-wo	output	
269	establish-maint-production-dry	process	
270	estimated	element	
271	estimated-delivery-date	element	
272	estimated-delivery-date-order	element	
273	estimated-repair-parts-cost	element	
274	estimated-shipment-date	element	
name list			
name	equipment-readiness-code	type	element
250	equipment-readiness-code	element	
251	equipment-recall-data	set	
252	equipment-recall-data-xx	set	
253	equipment-recall-deliv-1st	output	
254	equipment-recall-deliv-out	set	
255	equipment-recall-new-item-a	input	
256	equipment-recall-new-item-b	input	
257	equipment-recall-require	input	
258	equipment-recall-schedule	set	
259	equipment-recall-schedule-out	output	
260	equipment-recall-schedule-rep	output	
261	equipment-serial-number	element	
262	equipment-svstem-	system-parameter	
263	equipment-svstem-code	element	
264	equipment-utilization-code	element	
265	equipment-utilization-field	element	
266	err-msg	element	
267	error-exception-report-supply	output	
268	error-exception-report-wo	output	
269	establish-maint-production-dry	process	
270	estimated	element	
271	estimated-delivery-date	element	
272	estimated-delivery-date-order	element	
273	estimated-repair-parts-cost	element	
274	estimated-shipment-date	element	

Figure D-8. Sample LARE Name List (MOM) (Cont'd.)

APPENDIX E

BIBLIOGRAPHY

- Johnson, Larry A., Ph.D., Integration of CADSAT with General-Purpose System Functional Simulation Technology, Prepared for Space Systems Division (AFSC), Los Angeles, CA 90009, Contract No. F04701-77-C-0069, 18 July 1980, Logicon, Inc., Lexington, MA 02173.
- Johnson, L., F. Coker and D. Smith, Management Information System Requirements for ESD Program Offices, Prepared for Directorate of Computer Systems Engineering (TOI), Hanscom Air Force Base, MA 01731, Contract No. F19628-77-C-0178, Logicon, Inc., Lexington, MA 02173, Report No. DSJ-R77024, 31 March 1978.
- McDonough, Martin F., Analysis of the Application of PSL/PSA for Management Information Systems in the USACSC Environment, Final Report-Volume I, Prepared for U. S. Army Computer Systems Command, Logicon, Inc., Lexington, MA 02173, Report No. ESD-R0022, July 1980.
- McDonough, Martin F., CADSAT E-3A Application, Phase II Technical Report, Prepared for HQ Electronic Systems Division/TOI and E-3A Program Office, Hanscom Air Force Base, MA 01731, Contract No. F19628-79-C-0072, Logicon, Inc., Lexington, MA 02173, Report No. ESD-R80017, 28 May 1980.
- "Missing Computer Software," Business Week, September 1, 1980, pp. 46-53.
- Problem Statement Analyzer (PSA), Command Reference Summary (Version A5.1), ISDOS Project, Department of Industrial and Operational Engineering, The University of Michigan, Ann Arbor, MI 48109, ISDOS Ref. No. 79A51-0175-4, September 1979.
- PSL/PSA Example, Version A4.2, ISDOS Project, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, MI 48109, January 1978, ISDOS Ref. No. 7742-0186-2.
- Requirements Standards Study: Volume I - Technical Report Summary, Prepared for Rome Air Development Center, Logicon, Inc., Lexington, MA 02173, December 1978.
- REVS Quick Reference User's Manual, Prepared for Information Processing Operations Systems Engineering and Integration Division, TRW Defense and Space Systems Group, Huntsville, AL 35805, No. 27332-6921-035, 1 November 1979.
- REVS User's Manual, Revision B, NADC Edition, Prepared for Naval Air Development Center, TRW Defense and Space Systems Group, Huntsville, AL 35805, No. N62269-79-C-0434, 28 June 1979.

APPENDIX E

BIBLIOGRAPHY (Cont'd.)

SAMSO CADSAT Analysis, Final Report, Logicon, Inc., 255 W. Fifth Street,
San Pedro, CA 90733, Contract No. F04701-77C-0069 (P00006),
Report No. R:DS-R79044, 28 September 1979.

Smith, Daniel G., Requirements Engineering Guidebook, Requirements Engineering
Using an Automated Tool: PSL/PSA, Prepared for U. S. Army Computer
Systems Command, Logicon, Inc., Lexington, MA 02173, June 1980.

Teichroew, Daniel and Elliot Chikofsky, An Introduction to the Problem
Statement Language (PSL) and the Problem Analyzer (PSA), ISDOS Project,
Department of Industrial and Operational Engineering, The University of
Michigan, Ann Arbor, MI 48109, August 1980.

The IORL Operator User's Manual, Version 1-B, Systems Division, Teledyne
Brown Engineering, Huntsville, AL, UM-SD-08-80-921A, September 1980.

The IORL User's Language Manual, Version 1-B, Systems Division, Teledyne
Brown Engineering, Huntsville, AL, UM-SD-08-80-921, August 1980.