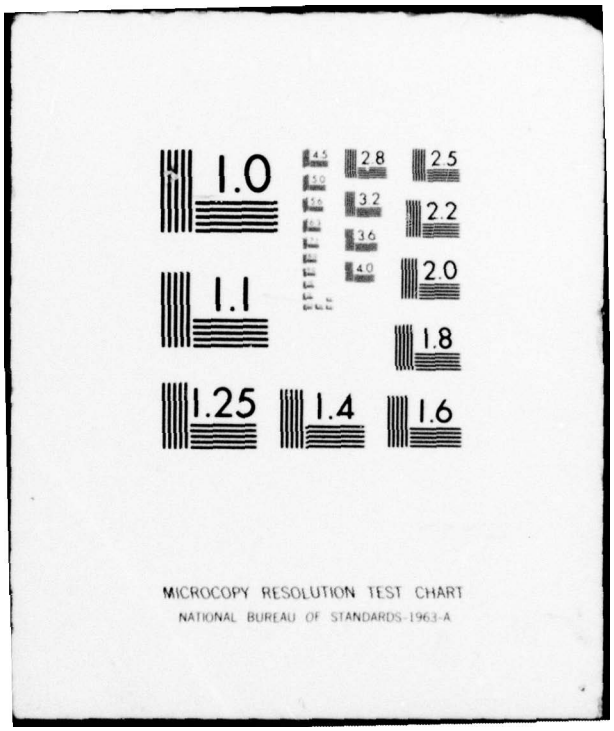
AD-A050 468 NAVAL INTELLIGENCE PROCESSING SYSTEM SUPPORT ACTIVITY--ETC F/G 5/2
INTEGRATED DATA BASE DEVELOPMENT AND DESIGN GUIDE. VERSION 1.1.(U)
DEC 77 L E TOWNER

UNCLASSIFIED NL

1 OF 3
AD A050468


1.0
1.1
1.25
1.4
1.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(1)

(6) INTEGRATED DATA BASE
DEVELOPMENT AND DESIGN GUIDE. Version 1.1.

NAVAL INTELLIGENCE PROCESSING SYSTEM
SUPPORT ACTIVITY
(NIPSSA)

(10) L. E. Towner

VERSION 1.1

(11) DEC[redacted]77

(12) 246 p.

PREPARED BY
SYSTEMS MANAGEMENT DIVISION
(NIPSSA 51)

→ SEE AD-A048176 (used as rept no.)
Not Report no.

410508

ACKNOWLEDGMENT

The author wishes to express his grateful appreciation to the NIPSSA personnel who assisted in the preparation of this Guide. Their comments, critiques, and proofing of the material contributed significantly to the preparation of a significant tool for integrated data base design. Special thanks go to Mrs. Patti Small and Mr. Bill Potter for their critical review and recommendations and to Mrs. Kitty Mears who coordinated the preparation of the Guide materials.

L. E. Towner

REPLACEMENT/ADDITION PAGES FOR CHANGE 1

1. Replace Table of Contents.
2. Add Record of Changes page.
3. Replace pages 1.2 and 1.3.
4. Replace Section 2 from 2.2 to end of section.
5. Replace Section 3 entirely.
6. Replace A.03-05, A.07-09, A.20-21, A.23, A.27, A.30, A.34, A.85.
7. Replace E.01, E.11.
8. Add E.14-E.26.
9. Replace A.06, A.18, A.19, A.32, A.33, A.38, A.41, A.45, A.62, A.68, A.72, A.90, A.107, A.109, A.110, A.117, A.124, A.125, A.126, A.127

TABLE OF CONTENTS

RECORD OF CHANGES

1.01	11/15/77	Typographical corrections
1.1	12/14/77	Replace Sections 2 and 3, E9-12, typrographical and format corrections

1. INTRODUCTION.

1.1. Background. The NIPSSA Integrated Data Base Development and Design Guide is the result of several years of experience developing integrated data base applications. It brings together into a structured methodology techniques which have survived trial by implementation. Some of the procedures within the Guide are relatively new and may require additional clarification. Comments and recommendations are welcomed and should be addressed to the NIPSSA-05 Data Base Administrator (DBA).

1.2. Purpose of the Guide. The development of an integrated data base is an expensive and highly detailed project. The speed with which applications can be added or enhanced is directly proportional to the analysis resources available. The most time-consuming part of the analysis is the definition of the data elements and their relationships. Once this is done the remainder of the design effort falls rapidly into place. The Guide provides a step-by-step set of instructions which lead to a subsystem specification of the user's desired application. At the same time it will permit the user, who knows more about the data than anyone else, to perform the initial phases of the analysis.

The Guide is meant to provide the complete picture and steps required to complete the design of a data base application. For this reason, all of the procedures to be followed by both user personnel and Database Administration personnel are included. Section 2.3 identifies who will perform each step of the design sequence.

1.3. Scope of the Guide. The Guide begins with the initial definition of an application requirement by the potential user. It then proceeds through several analysis steps, including periodic reviews and decision points, to the completion of a subsystem specification document which will be used to support funding and software development. Hardware support for the application is not specifically addressed by the Guide. Where hardware must be procurred to support the application, validation and procurement approval must be performed in parallel with the preparation of software specifications. This hardware requirement must be documented as part of the subsystem specification.

The Guide does not include a number of common features of the integrated data base which are provided independent of application subsystems. These features include recovery of data bases during processing malfunctions, logging of transactions, user access protection, and security

control. Detailed explanations of these common features and their relationship to applications is available through the DBA staff.

1.4. Data as a Resource. Historically, ADP has viewed data as a possession of individual users. Design of ADP systems has been oriented to the outputs, typically reports, desired by a single user. The data files rarely contained data elements which were not required for the immediate needs of the primary user. The concept of an integrated database has forced a change in this thinking. Database encompasses entire organizations. The data stored is often used by more than one person and may be organized for many different outputs. DATA BECOMES A RESOURCE of the organization instead of a possession. The development concepts associated with this approach have moved from an output-oriented philosophy to a complete resource philosophy. This requires that the system designer significantly alter his/her thinking and look beyond the immediate benefits of a data file to the potential needs of the user organization as a whole. This Guide, with its methodology, will assist the database designer in following this approach.

2. PROCEDURAL OVERVIEW.

This section describes, in general terms, the steps which are to be performed to accomplish the description of a data base application. Detail step instructions are presented in Section 3.

2.1. The Integrated Data Dictionary (IDD) Facility. It is essential that the design of a large and complex data base be controlled and monitored. This monitoring is best performed using a software tool known as a data dictionary. The data dictionary, similar to a standard reference dictionary, defines and relates terms and data descriptions that are used within the data base. The integrated data base uses the Integrated Data Dictionary (IDD) software package developed by Cullinane Corporation. IDD works in close association with the Integrated Database Management System (IDMS), also marketed by Cullinane. IDMS is the heart of the integrated data base and performs all data base support functions.

IDD is used extensively during the design process. Each major step of Section 3 includes entries to the data dictionary or the data base itself. This approach centralizes all design efforts and assures continuity with the design of the existing operational data base. Reports produced by IDD permit rapid

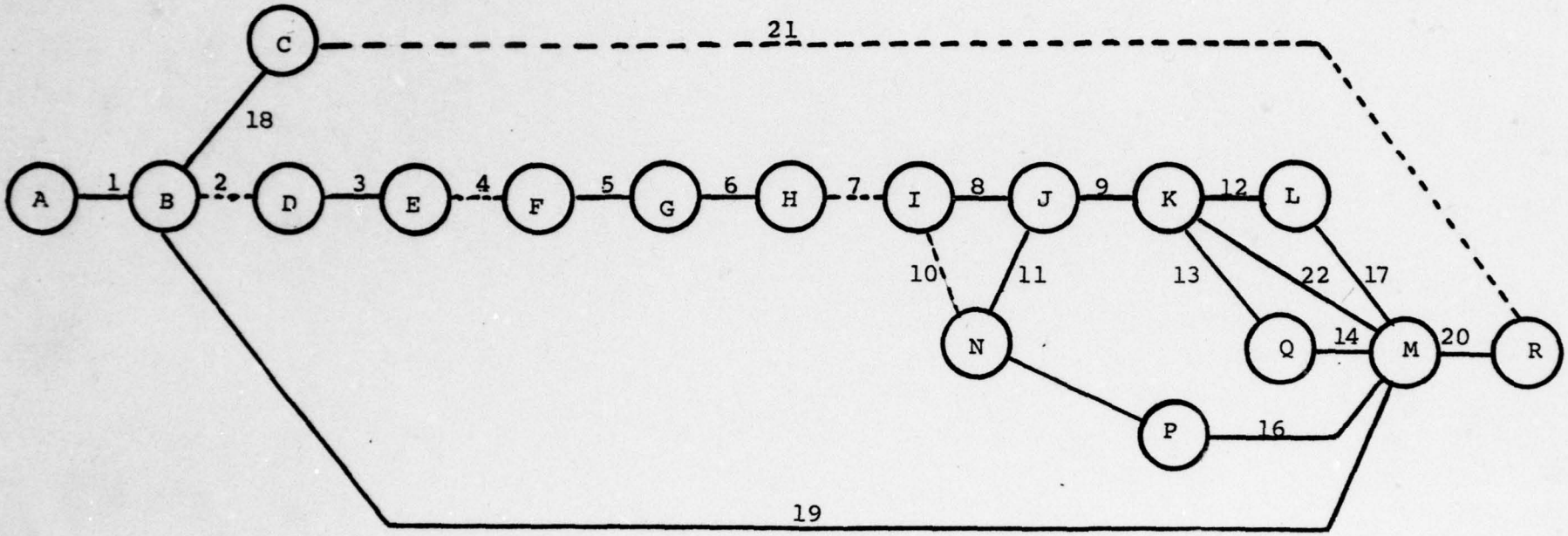

A
B
C
D
E
F
G
H
I
J
K
L
M
N
P
Q
R
1
2
3
4
5
6
7
8
9
10
11
12
13
14
16
17
18
19
20
21
22

Figure 2.2.

Reports produced by IDD permit rapid review of the design efforts and improve the accuracy of the overall process.

2.2. Design Step Sequence. Figure 2.2 illustrates the order and relationships between the design tasks. Section 2.3 provides a brief description of the individual tasks which are indicated in Figure 2.2 as lines. Tasks performed by user personnel are shown as dashed lines. Remaining tasks are performed by the DBA staff, often with user assistance. Section 2.4 briefly describes the milestones to be achieved during design. Milestones are illustrated as circles.

2.3. Summary of Tasks. Subsection numbers are identical to the task numbers shown in Figure 2.2.

2.3.1. Identify the application. Determine if the application coincides with an existing data base capability. If so, much of the design process is bypassed.

2.3.2. Define the individual data elements required by the application. Prepare IDD entries defining each data element

2.3.3. Review data element definitions with DBA staff. The DBA staff assists user personnel in correcting and modifying definitions, as required, to conform to standards and conventions. Determine which data elements exist in the data base. Define whether the application is an update to existing data base capability or a new capability.

2.3.4. Group data elements which are related. User personnel prepare IDD entries establishing groups of data elements where an interrelationship exists.

2.3.5. Review data element groups. The DBA staff determines the proper ordering and association of data elements assisted by user personnel.

2.3.6. Identify the tie points to an existing data base. The DBA staff defines which data elements and records in the existing data base are concurrently used by the application.

2.3.7. Define records. User personnel determine which data elements/groups constitute a data base record. Define those groups which occur more than once. Determine the manner in which the record relates to other records in the application and the data base as a whole. Identify the data elements within the record]

completely. A narrative description of each element is required. This task is performed by user personnel.

2.3.3. Review data element definitions with DBA staff. The DBA staff assists user personnel in correcting and modifying definitlons, as required, to conform to standards and conventions. Determine which data elements exist in the data base. Define whether the application is an update to existing data base capability or a new capability.

2.3.4. Group data elements which are related. User personnel prepare IDD entries establishing groups of data elements where an interrelationship exists.

2.3.5. Review data element groups. The DBA staff determines the proper ordering and association of data elements assisted by user personnel.

2.3.6. Identify the tie points to an existing data base. The DBA staff defines which data elements and records in the existing data base are concurrently used by the application.

2.3.7. Define records. User personnel determine which data

elements/groups constitute a data base record. Define those groups which occur more than once. Determine the manner in which the record relates to other records in the application and the data base as a whole. Identify the data elements within the record which the user will desire to access specific occurrences of the record in the data base.

2.3.8. Define the application structure. The DBA staff, assisted by user personnel, finalizes the relationship between records and defines the conditions under which those relationships will exist.

2.3.9. Review the application structure for consistency with standards and conventions. Prepare appropriate IDD entries. Prepare entries to define the application to the IDMS system. This step is performed by the DBA staff.

2.3.10. Define the basic output requirements. User personnel determine which data elements are required for each output. Prepare IDD entries defining the output program.

2.3.11. Review structure to assure that basic reports are

efficiently supported. It is possible to improve the response time for a report/output by modifying the relationship between application records. This step is performed by the DBA staff.

2.3.12. Update the IDD with tie points to existing data base. Prepare entries which define common relationships between records and data elements in the new application and the current data base.

2.3.13. Define input processing (IP) data formats for batch input. The DBA staff prepares IP coding instructions and specifications to permit the user to enter volume data into the data base.

2.3.14. Merge input specifications into subsystem specification document. This step is performed by the DBA staff.

2.3.15. Prepare detail report specifications. These specifications will be used as instructions to programmers and are prepared by the DBA staff.

2.3.16. Merge report specifications into the subsystem

specification document. This step is performed by the DBA staff.

2.3.17. Merge IDMS definitions, application data relationship diagram, and data base tie points into the subsystem specification document. The DBA staff performs this step.

2.3.18. Train the user in use of existing data base capability. Where an existing capability will support the application requirements, provide the user with necessary training. Training is performed by the DBA staff.

2.3.19. Prepare the basic subsystem specification document. The DBA staff organizes the individual sections and adds supporting material.

2.3.20. Review and finalize the subsystem specification. The DBA staff assures that the document completely defines the desired application to be developed. Obtain formal user approval.

2.3.21. Once training is completed (where an existing capability is to be used), turn the capability over to the user for utilization.

2.3.22. Prepare on-line display screen specifications. The DBA staff, assisted by user personnel, prepares IDD entries defining on-line input processing screen display formats and requirements.

2.4. Summary of Design Milestones.

2.4.1. A- Initial service request has been received from the user.

2.4.2. B- The application definition and purpose have been established. Determination has been made that the request can be serviced by existing capability, modification of an existing capability, or development of a new capability.

2.4.3. C- An existing capability will support the user. The user has been trained in the use of the capability utilizing existing hardware and software facilities. Manuals have been provided to the user.

2.4.4. D- The data element descriptions necessary to support the application have been prepared using IDD entries.

2.4.5. E- The Data Base Administrator (DBA) has reviewed the data element definitions for corrections and adherance to standards and conventions. A determination of the scope of the development has been prepared.

2.4.6. F- Data elements have been grouped according to their inter-relationships.

2.4.7. G- The DBA staff has reviewed the initial data element groupings for consistency and made any necessary adjustments.

2.4.8. H- The DBA staff has identifled the tie points of data elements to the existing data base structure.

2.4.9. I- Data base records/record modifications have been defined. Entry points into the application data base have been identified. IDD entries have been prepared.

2.4.10. J- The estimate of the scope of the development has been updated including basic outputs.

2.4.11. K- All records in the proposed data base have been identified and the relationships between records defined. An IDMS data base definition (schema) has been prepared.

2.4.12. L- The IDD reflects the addition of the user's application includlng basic outputs.

2.4.13. M- The subsystem specification is complete in draft form.

2.4.14. N- Basic output definitions are complete. IDD entries describing the outputs have been prepared and stored in the data dictionary.

2.4.15. P- Detail programming specifications for outputs are complete.

2.4.16. Q- Input Processing formats and programming specifications are complete. Teleprocessing screen definitions for input processing are defined.

2.4.17. R- An existing capability has been turned over to the user for utilization. A new or modified capability has been defined and has been approved by the user.

2.5. Review and Decision Points.

The user and the DBA staff must periodically review and make decisions about the service request. Based on

analysis decisions on the future and direction of the application must be made. The service request may be terminated or altered significantly at any of the decision points. In some cases a return to the beginning may be indicated. The review and decision points are defined by user and DBA grouping.

2.5.1. User Decision Points.

2.5.1.1. B- The user must decide whether to pursue the application where a modified or new development is required. If a decision to continue is made, the user must elect to perform tasks two and three: (a) in-house; (b) through user-obtained contractual assistance; or (c) through NIPSSA-obtained contractual assistance.

2.5.1.2. E- The user must decide whether to continue the project based on estimates of effort and time developed to this point. It is assumed that the performance method chosen in (B) would be continued.

2.5.1.3. J- The user must decide whether to continue the project through completion of the subsystem specification.

2.5.1.4. R- The user must approve the subsystem specification document and provide funding necessary to perform development of the required software. A determination will be made, by the DBA staff, of the approach to be followed in assigning performance of the software development phase. The interface between the DBA staff and the development programmers makes NIPSSAobtained contractual support desirable.

2.5.2. DBA/NIPSSA Decision Points.

2.5.2.1. B- The DBA staff must determine whether the proposed application can be supported in the time frame desired by the user. An impact evaluation must be performed to determine what affects the proposed application has on the existing system.

2.5.2.2. H- The DBA staff must determine if the previous decision regarding new/modification is correct. Appropriate adjustments in the scope of the project may be required.

2.5.2.3. K- The DBA staff must schedule the restructuring of the existing data base if modification is the selected approach.

2.5.2.4. R- The DBA staff must approve the subsystem specification and the resource/time development estimates.

2.6. How To Use The Guide. The Guide is laid out in a sequential step mode. It's organization encourages the use of methodical, step-by-step progress from the beginning request by a prospective user to the definition of the data base subsystem which provides the requested support.

Figure 2.2 identifies some tasks which may be performed in parallel. The figure is, however, primarily defining the inter-dependencies of the various tasks. Until the designer has completed a development and has a firm grasp on the design techniques, parallel task efforts are not encouraged.

The Guide does not separate those tasks performed by user personnel and those performed by Database Administration personnel. It, instead, identifies who performs each task in the development path. It is the responsibility of use and DBA personnel to work closely together, keeping the progress of development flowing smoothly.

It is important at this point to point out that the effective and productive data base design requires strict adherance to established standards and conventions. These

standards were established to make expansion of the data base easier and less costly. They also affect the ability of the data base to be utilized by other users. When conditions or data are encountered which are not standardized, standards may be defined and recommended for adoption. Refer such situations to the DBA staff for action.

The DBA staff will provide each requestor with a design support package. This package consists of two copies of this guide and copies of the necessary forms for defining the data base. A supply of continuation forms is also included for additional description and comment lines where required.

Complete and accurate descriptions and comments are essential to the effective definition and understanding of the proposed system. As many comments as are necessary to fully describe the sytem, its functions, purpose, scope, and data content should be provided. The data dictionary will hold all the information describing the application. It is, however, not possible for the dictionary to contain and assist in the definition of the system what the user does not enter but keeps in his/her head. The final product will only be as good as its original definition. Extensive change after implementation is very costly.

Section 3 of the Guide and related appendices provide detail instructions for each step of the data base design. It is recommended that those sections and appendices which apply to individual members of the design team be copied so that each person has a personal copy. Questions and comments about the Guide should be grouped and referred to the DBA staff.

It is very important that the design team consider an application from several levels. The integrated data base is a resource which may be used by all levels within an organization. Typical design considerations include the needs of the persons who will enter detail data into the data base. What kind of results do they require? What makes the effort of feeding the data base worthwhile from their standpoint? The application may have as a major goal the support of upper level management with information. However, it is necessary that each level between those who prepare the data and the higher level of user be considered too.

When the flow and use of data from the lowest to the highest level within the organization is considered, the data base truly becomes an organizational resource. Without this consideration, it becomes a limited tool which will never achieve its full potential. Keep in mind that storing

data in its raw form, without summarizing, permits its use in a wide variety of ways by all levels of potential users.

One of the phenomena which often occurs during data base design is a sudden expansion in scope of the application. It is common to suddenly see additional potentials for development and use of the data base. It is possible for the application to suddenly get out of hand as all of the possibilities for future capabilities are identified. Therefore it is very important to identify the boundaries which are to be placed on the application under consideration. These boundaries may be moved to accommodate minor changes but should generally be firmly held in place.

When additional capabilities are identified during the design process, it is often desirable to define "hooks" in the design which can be used for development of the capabilities at a later date. "Hooks" take the form of dummy records which are not currently used or completely defined but are inserted into the design to reduce effort when the capability is added. It is very important that the designer clearly and completely identify, as part of the dictionary comments, the purpose of each of the hooks. Remember that the hook may not be developed for some time and the originator may not be available for consultation when development begins.

It is normal to assign all of the records for a particular application to the same data base area. This optimizes storage definition and assures that all of the data is physically stored together. It is possible to locate some records in a separate area, particularly when the record is sensitive and should be isolated. The volume of data may also have some bearing on this decision. As design progresses, records which have special conditions or restriction should be identified so that area location decisions can be made.

3. DETAIL DESIGN AND DEVELOPMENT INSTRUCTIONS.

Each of the summary steps described in Section 2.3 is defined in detail in the subsections below. The subsection number corresponds to the task number in Figure 2.2. Most subsections are further clarified with detailed instructions in the appendices.

3.1. Identify the Application. This process begins with the formulation of the need for ADP services by the user. The user may have existing ADP applications which require expansion. The requested service may be new for the requesting user but available within the system supporting other users. Finally, a completely new requirement may be proposed. The majority of this document assumes that the user's request requires the development and design of a new application to meet the service request.

Upon receipt of the request for ADP services, NIPSSA will assign an analyst as an initial point of contact to review the proposed project with the user and make a determination of the scope of the project. The analyst will also determine if this prospective user is currently receiving ADP services from NIPSSA. If not, IDD entries must be prepared to identify the new user to the data dictionary. Appendix A.1 provides instructions for

preparing IDD user entries. At the same time the user is identified to the Naval Intelligence Command Management Information System (NIMIS) accounting by preparing NIMIS organization entries described in Appendix A.2.

The DBA staff will prepare an initial resources estimate and implementation estimate to assist the user in determining whether to proceed with the application.

A NIMIS project number is assigned once approval is received from the user to proceed. See instructions in Appendix A.3.

IDD subsystem entries are prepared if the application is new. See Appendix A.4. Where the application is a modification to an existing system or a matter of adding the user to an existing system the IDD entries are prepared as defined in Appendix A.5.

3.2. Define the Individual Data Elements Required by the Application. Each data element to be used by the application must be explicitly defined. It is essential that as much detailed information about the data element and its characteristics be included in the definitions as possible. Appendix A.6 provides detailed instructions for completing IDD entries for each data element. During the definition process, analysts should use the data element definition conventions

provided in Appendix B as a guide for naming and structuring data element definitions.

3.3. Review Data Element Definitions with the DBA staff. Once all data elements which will be required by the application have been defined, the DBA staff reviews the definition entries for completeness and content. A cross-reference check is then performed using IDD data element reports to determine if any of the desired elements are present in the data base.

Data elements which correspond to existing elements in the data base are reviewed to determine if the definition is consistent with the data base definition. Inconsistencies must be resolved before proceeding to the next step. When only a few elements correspond, a new application is likely. The corresponding elements are likely candidates for tie points into the data base (See 3.6.).

Review elements to assure correspondence with published standards and conventions, in both name and structure. Modify the IDD entries as required to establish correspondence and enter into the data dictionary.

An initial estimate of the labor resources, machine time, and calendar time required to develop the data base capability is prepared. This estimate is exclusive of report/output modules and covers input requirements only. An estimate of

the time and costs associated with providing an initial data base loading must also be provided. Special purpose loading programs must be included in the estimate. The information must be adequate for the user to decide whether to continue or cancel the application. Special purpose loading programs are described to the data dictionary using entries defined in Appendix A.23.

3.4. Group Related Data Elements. The user must now identify data elements which are related to each other. The number of times data elements and groups of elements may occur must also be defined. Appendix A.7 provides directions for preparation of IDD entries to group data elements. This step is very important to the overall design effort because the logical relationship of data elements is often subtle. The incorrect association of data elements can lead to serious degradation of system performance and costly corrective effort. Careful review by knowledgable user personnel is essential and the initials of the user person reviewing the element grouping should appear beside the initials of the analyst who prepared the IDD entries in the PREPARED BY clause.

3.5. Review Data Element Groups. The DBA staff reviews the user-defined groupings of data elements. The positioning of each data element within the group is reviewed for logical ordering and adjustments are made as required. IDD entries prepared in the previous step are stored in the data dictionary and reports prepared. The reports are reviewed to assure consistency with the data base and corrections made, if required.

3.6. Identify the Tie Points to the Existing Data Base. Data elements for the application which correspond to those existing in the data base are potential tie points (see Section 3.3). The DBA staff must determine which corresponding elements to use as tie points. Where possible the tie points should relate to existing records which are stored in a CALC (random) mode with the data element which corresponds between the two records used as the reference key. Where possible this data element should be the CALC key of the associated record. The data element should be the object of a secondary index if this is not the case. See Appendix A.24 for instructions for defining secondary indexes to the IDD. One of these two modes is desirable if rapid cross-over between areas and applications within the data base is to be accomplished

during retrieval. A data base modification request must be completed if it is necessary to establish a secondary index within the existing data base. See Appendix A.8 for instructions on preparing this request.

3.7. Define Records. Determine which data elements and element groups constitute a data base record. Organize the order of elements/groups so that key fields are placed at the beginning of the record, followed by optional fields, and repeating fields. Where an OCCURS DEPENDING ON (variable repeating field) group is present an OCRS-xxxx data element must be defined and placed at the beginning of the data portion of the record. The variable field(s) must be placed at the end of the record. If the record is to be protected control and record statements must be placed at the very beginning of the record description.

Two sets of record definition entries must be prepared:

a. IDD entries describing the record and defining the subordinate element groups and elements. See Appendix A.9 for instructions.

b. Data base schema definition entries which describe the record to the schema (data base structure definition) are prepared according to instructions in Appendix A.10.

Determine which records are required as entry points

into the data base and which records are dependent upon other record occurrences as part of the definition process. Data base entry records should be defined as randomly accessed (CALC), permitting direct retrieval of the record through use of the key field value. A secondary index (alternate access point) should be defined if the nature of the record's relationship to the application makes random access impractical.

3.8. Define the Application Structure. This step is one of the most critical in the design process. Here the relationship of one record to another must be defined. Depending on the relationship, it is possible for certain special purpose records to be defined. In addition, it is possible that some data elements may be relocated from one record to another. These questions must be asked about the relationship between any two records in the application:

a. Is the presence of one record's occurrences dependent upon the presence of another record's occurrences? If so, the dependent record is the "member" of a relationship called a "set." The other record is the "owner" of the set.

b. Can an occurrence of the member exist in the data base

without a corresponding occurrence of the owner record? If so, the relationship is established under control of the user and is termed "manual." The relationship is established under control of IDMS and is termed "automatic" if an occurrence of the owner must be present before the member occurrence can be stored in the data base.

c. Is it possible to change the relationship of a member record occurrence from one owner record occurrence to another owner record occurrence in the data base? If so, the relationship is termed "optional." The relationship is termed "mandatory" if a member occurrence must remain related to a specific owner record occurrence as long as the member occurrence remains in the data base.

d. Is the relationship between an owner record occurrence and several occurrences of the member record based on a key field which is sequentially sorted? Is there a logical ordering of the member record occurrences which is advantageous to the reporting or displaying of the member record occurrence? Where the answer to either of these questions is positive, a sorted relationship between owner and member records should be considered. When considering a sorted relationship the manner in which the data is received is important. If, for example, the sorted data is in chronological order and is updated chronologically,

most of the new data will be added to the end of the member data chain. This requires that IDMS search through the majority of the member data chain to insert each new record occurrence. Where this condition is prevalent the sort should be defined as "descending" instead of "ascending." The data may still be accessed in ascending order by reading the data chain in reverse order.

e. The use of "owner" pointers will improve retrieval speed if the member record is often accessed through another set relationship and retrieval of the owner record is frequently desired. The use of "prior" pointers is recommended in nearly all cases. Prior pointers permit reading data records forward and backward. They are important as a factor in the updating efficiency of the DBMS.

f. Where the order in which member records are entering the data chain is unimportant, the "next" order option is recommended. This is the most efficient storing option. This option generally loads new data near the front of the data chain. The "prior" option, just as efficient, will generally load new data near the rear of the data chain. Less efficient is the "first" option which always places the new member at the front of the data chain and "last" which always places the new member at the end of the data chain. This is useful when last-in-first-out

(LIFO) or first-in-first-out (FIFO) ordering, respectively, is desired.

g. The next question which must be asked: Does any application record type own another record which is a mirror copy of itself? An example of this situation is two organizations who are related to each other, such as a department and its subordinate division. A single record type will describe both and any other organizations as well. The key is to define to the data base how they are related. The method of relating the two (or more) occurrences of the same record type is the basis for bill-of-materials processing programs, otherwise known as "DBOMP." The owner record type is connected to a specially defined member through two parallel sets. Appendix C.2 describes this structure and presents the conventions for its definition. Where this structure need exists, it will be necessary to define the data elements and record of the special member.

h. The final major question which must be answered is whether the owner and member record types exist in what is known as a many-to-many relationship. This relationship occurs when one occurrence of the owner may have many occurrences of the member and, at the same time, one occurrence of the member may have many occurrences of

the owner. In this case, the owner may also be a member. IDMS will not process this type of record structure without some help. The approach is to define another special member record which is placed between the original owner and member (which now becomes an owner record of the special member). Each association of the two original records is established by storing an occurrence of the special member record in the data base. This record may contain data elements which occur only when the two original records are associated. A detailed description of this record type and the conventions for its use are defined in Appendix C.3.

Once all record relationships have been defined data base definition statements defining each relationship as a set are completed. Appendix A.11 describes the procedures for preparing set entries.

An update of the original development estimate is prepared to assist the user in deciding whether to continue the project. Estimates produced here are for input and data loading activities only.

3.9. Review the Application Structure for Consistency With Standards and Conventions. The DBA staff reviews the structures defined in the previous steps. The entries prepared

are examined to be sure that standards and conventions have been followed. Changes are made where necessary. The data base definition (schema) entries are organized into the form and order which the schema processor requires. An initial schema compilation may be performed following loading of the schema entries into the programming support library (PSL). This will locate initial errors in the definitions. Final compilation of the schema must wait until the steps in section 3.11 have been completed.

3.10. Define Basic Output Requirements. The application will require basic reports to provide the user with a picture of the data base supporting the application. Appendix A.12 defines the procedures required to define the format of a report. Appendix A.13 describes IDD entries required to define a report program. IDD file definition entries must also be prepared if an output from the application is in the form of punched cards, magnetic tape, or magnetic disk. See Appendix A.14.

3.11. Review Structure to Assure That Basic Reports Are Efficiently Supported. It is often possible to improve the effectiveness of report or output preparation by modifying the structural relationship between data base records. This is

most commonly done by sorting some data sets or adding special purpose sets between records. This must be done with care. Such additions will improve the responsiveness of retrieval operations but will also generally slow the updating of the data base. Where update volume is very high a compromise may be necessary to prevent unacceptably long update runs. It is particularly dangerous to connect two record types which are stored randomly (CALC) by a sorted set. The increase in input-output operations during updating is dramatic and the time required increases factorially as the number of record occurrences grows.

Modify or add the schema definition entries as required and compile the schema. The DBA staff will review the schema for errors and incompatibilities and store the final schema definition in the data dictionary. The DBA staff will then develop the Device-Media Control Language (DMCL) module. See Appendix A.15. A second module, known as a "subschema", is prepared by the DBA staff, (Appendix A.16). These modules will be processed and entered into the data dictionary.

An estimate of the labor hours, machine time requirements, and calendar time necessary to prepare the specified output modules is developed. This information is merged with that for input and data loading requirements to define

the total scope and schedule of the project. The user must have sufficient information to make a decision whether to continue or cancel the project.

3.12. Update IDD with Tie Points to Existing Data Base. Once application definition has proceeded to this point, it is necessary to connect the application to related areas of the data base. This is accomplished by relating those data elements and records which are used as tie points between the operational data base and the new application. Appendix A.17 defines the IDD entries which are prepared to accomplish the association. A complete set of dictionary reports is produced once the data dictionary has been updated with all information about the application. The reports are reviewed by the DBA staff to assure that no inconsistencies exist between the new application and the total data base.

3.13. Define Input Processing (IP) Data Formats for Batch Input. Even if the user plans to operate the application in an on-line mode it will be necessary for initial or volume data to be entered in a batch mode. Appendix A.18 defines the procedures for preparing IP specifications. As part of the IP specification preparation user coding

instructions and coding form layouts must be prepared. Appendix A.19 defines the format for user coding instructions. Appendix A.20 illustrates coding form layouts and provides guidelines for layout preparation.

The combination of IP specifications, user coding instructions, and IP layout forms makes up the complete specification which will be used to produce IP software.

3.14. Merge Input Specifications Into Subsystem Specification Document. Appendix D provides instructions for the preparation of a subsystem specification document. Input specifications are included in the subsystem specification in section 4.2.

An estimate of the labor hours, machine time requirements, and calendar time required to produce each IP module is prepared for scheduling and costing purposes.

3.15. Prepare Detail Report/Output Specifications. Preliminary report/output specifications prepared in Section 3.10 are now expanded to fully define the scope and content of an output. The recommended programming language and operating requirements for the defined output module are specified. Section 4.2 of the subsystem specification should be used as a guideline for information to be

contained in the specification. IDD data dictionary information on the program module and any special files generated are included as part of the specification.

An estimate of the labor hours, machine time requirements, and calendar time required to produce each output module is prepared.

3.16. Merge the Report Specifications Into the Subsystem Specification Document. Appendix D defines the format of the subsystem specification document. Each output becomes a separate entity within section 4.2 of the specification.

3.17. Merge IDMS Definitions, Application Data Relationship Diagram, and Data Base Tie Points into the Subsystem Specification. Information about the application data base will be included as Appendix A of the subsystem specification document.

3.18. Train the User in Use of Data Base Capability. This step is executed when the user has requested use of an existing data base facility.

3.19. Prepare the Basic Subsystem Specification Document. The introductory sections of the subsystem specification are

written as defined in Appendix D. Particular attention should be paid to the material for sections two and three of the specification. The information provided in these sections will be used as a basis for final approval of the application, procurement of required hardware, funding for software services, and scheduling of the development.

3.20. Review and Finalize the Subsystem Specifications. The DBA staff performs final review of the specifications to assure continuity and consistency. ADP management reviews the specification to insure that it is consistent with short and long range policy. The user reviews the specifications to insure that it provides an ADP solution to the specified requirement. Both user and ADP management formally approve the document.

Previously prepared estimates of labor hours, machine and calendar time requirements are reviewed and updated to reflect the latest estimates and anticipated development schedule for the project. The user defines the source of funding and agrees to the development schedule.

3.21. User Turnover. This step follows step 18 (Section 3.18) where the user will utilize an existing system facility. The user is added to the data dictionary as authorized

to access the data base. Appendix A.21 describes updates to the IDD user entries defining authorization information.

3.22. Prepare On-line Display Screen Specifications. This section is to added.

APPENDIX A

DESIGN INFORMATION PREPARATION INSTRUCTIONS

APPENDIX A.1.

NEW IDD USER ENTRIES

Each user of the data base is an unique entity. A user for this purpose is defined as an organizational entity, of varying size and structure, which is identified individually to the system. Each user may include one or more persons who are identified to the data dictionary as subsets of the user organization.

Each user is assigned a 16-character abbreviation of the full organizational name stored in the ORGANIZAT-UNIT record within the data base. A single coding sheet, Figure A.1, is used to prepare the new user entries. When specific entries on the same page are not applicable they should be lined out to prevent punching.

1. ADD USER NAME IS. The use name is composed of two segments, each 16 characters in length and enclosed by single quote marks. Segment one contains the alphanumeric acronym/abbreviation of the user identification/name. The field is required and must be unique. The name will be verified by the DBA staff. The second segment contains the last name of the individual person within the user organization who has data base responsibilities of any kind. Access of the data base is restricted to user personnel identified to the data dictionary. If the person name

segment is omitted the organization as a whole is identified. This serves to define an organization to the data dictionary but does not grant data base access.

2. OF SUBSYSTEM xxxxxxxx VERSION 0101. This line is completed if the new user will utilize an existing application. The DBA staff will change the entry to reflect the correct version if the subsystem version is other than 0101. This line is omitted if the application is new.

3. USER DESCRIPTION IS. This entry permits expanding the name of the user organization. A single quotation mark (') is placed at the end of the expanded name within the space provided.

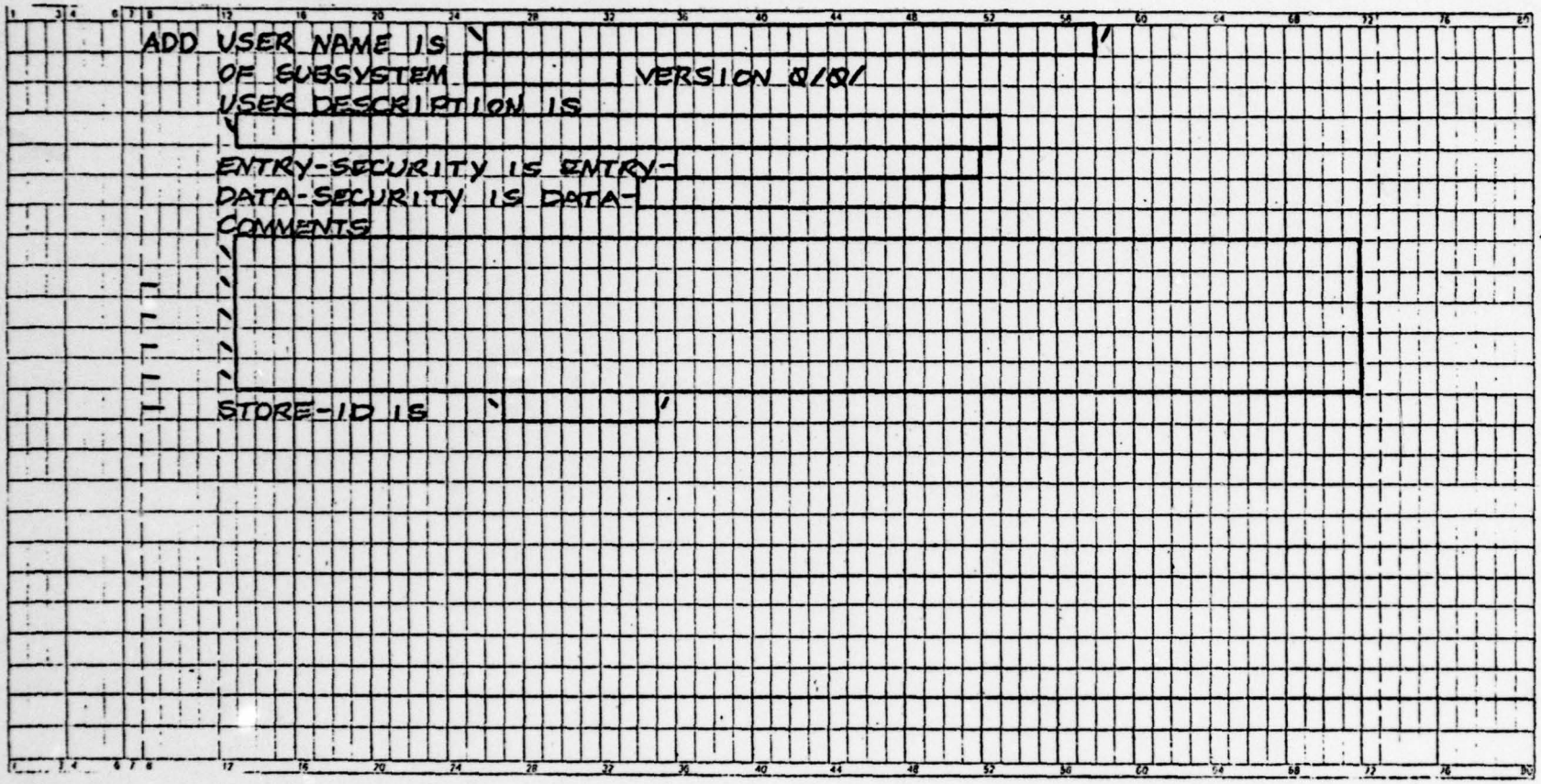
4. ENTRY-SECURITY IS. This statement identifies the security classification associated with the information about the user organization recorded in the data dictionary. Valid entry values are described in Appendix E.1.

5. COMMENTS. This statement permits a description of the user organization and location. Five lines are provided on the coding sheet. An unlimited number of lines may be coded as long as they are all contiguously stored in the dictionary. A single quote mark (') must terminate the last line of comments.

PAGE IS INTENTIONALLY BLANK

INTEGRATED DATA BASE DEVELOPMENT AND DESIGN GUIDE
ADD USER TO DATA DICTIONARY

PAGE 1 OF 1



APPENDIX A.2.

NIMIS ORGANIZATION ENTRY DEFINITION

All organizations served by NAVINTCOM are stored in the data base. This information is used for accounting purposes and for origin points of retrieval of data belonging to the organization.

Three entries are used to enter the complete organization name and address data into the data base. These are:

1. AAM which stores the basic organization record in designated areas of the data base and includes a user code used in ADP accounting and the full name of the organization.

2. AAN. This entry may be repeated up to seven times to store address information. If the name exceeds the 30 characters permitted in the AAM line the name may be continued on subsequent AAN lines.

3. AAO. This entry permits the user to identify associations with other organizations. When the new organization is entered as "owner", other organizations shown in the "owned" field are considered subsidiary to the new organization. If the new organization is shown in the "owned" field, it is subsidiary to the organization named in the "owner" field. As many of these entries as are required may be prepared.

Detailed preparation instructions for each entry follow.

AAM Organization Initiation Entry.

1. The literal "AAM" must appear in the first three positions of the input entry.

2. The literal "S" must appear in position four of the input entry to store a new organization in the data base.

3. Positions 5 through 20 contain the acronym of abbreviated name of the organization. Hyphens (-) are eliminated from organization acronyms. The DBA staff will verify that the organization name is unique. This name is identical to the user organization name defined in Appendix A.1. Positions 21 through 34 are unused.

4. Positions 35 through 37 contain a unique identifier used by ADP machine accounting. This identifier will be assigned by the DBA staff.

5. Positions 38 through 67 contain the full name of the organization. If the full name cannot be spelled in 30 characters, lines of the AAN address entry may be used to complete the full name (See AAN, item 4).

AAN Organization Address Entry.

1. The literal "AANM" must appear in the first 4 positions of the input entry.

2. Positions 5 through 20 contain the acronym/

abbreviation of the organization as stored in the data base. Positions 21 through 34 are unused.

3. Position 35 contains a numeric literal ranging from two through eight. This literal defines the address line identified by the entry.

4. Positions 36 through 65 contain the 30-character address line. The entry is free-form and normally left justified.

AAO Organization Association Entry.

1. Positions 1 through 4 must contain the literal "AAOI".

2. Positions 5 through 20 contain the acronym/abbreviation of the organization which owns other organizations (to which other organizations report). Positions 21 through 34 are unused.

3. Positions 35 through 50 contain the acronym/abbreviation of organizations owned (reporting to) the organization named in positions 5 through 20.

NOTE: Prior to submitting the entry the DBA staff must determine that the other organizations named in the AAO entry are already present in the data base.

FIGURE A.2.1 TO BE ADDED

APPENDIX A.3.

NIMIS PROJECT INITIATION.

Each task for ADP service is assigned a NIMIS project identifier for accounting purposes. All personnel and computer resources used to accomplish the tasking are recorded using the assigned NIMIS project.

Seven different entry lines may be used to record the NIMIS project. Some of these lines are used only in certain circumstances:

1. AAT is required to initially store the NIMIS project in the data base. It provides association information to the requesting and performing organizations, the NIPSSA accounting project identifier, and the applicable budget action program.

2. AAU and ABF provide the project title in either unclassified or classified mode, respectively. The ABF entry is prepared only if the project title itself is classified.

3. AAX entry is used to provide a brief expanded description of the project. It is optional but recommended and appears on summary reports describing NIMIS projects.

4. AAY entry defines the various dates and the project status.

5. AAW entry provides a variety of miscellaneous

information about the project.

6. ABG entry defines the NIPSSA personnel assigned to the project and the period of their assignment.

Detailed descriptions of each individual entry follow.] Where descriptions of individual fields have been omitted those fields are not required for this type of project.

AAT NIMIS Project Initiation Entry

1. Positions 1 through 4 must contain the literal "AATS."

2. Positions 5 through 16 contain the identifier of the NIMIS project. The project number is assigned by the DBA staff.

3. Positions 17 through 32 contain the acronym/abbreviation of the NIPSSA organization which will coordinate the NIMIS project. This will normally be the DBA staff unit.

4. Positions 33 through 48 contain the acronym/abbreviation of the requesting organization for whom the project is being performed. This will normally be the same organization identified by the NIMIS entries in Appendix A.2.

5. Positions 49 through 64 contain the NIPSSA internal ADP accounting number assigned to the project. The number is assigned by the DBA staff.

6. Positions 65 through 74 contain a serial number of

the document requesting the ADP services.

7. Positions 77 through 80 contain the budget action program identifier associated with the project.

AAU/ABF NIMIS PROJECT TITLE ENTRY

1. Positions 1 through 4 of the entry must contain either the literal "AAUM" if the title text is unclassified or "ABFM" if the title text is classified.

2. Positions 5 through 16 contain the NIMIS project identifier assigned by the DBA staff.

3. Positions 17 through 49 contain the title of the NIMIS project. If the title is too long for the space and is unclassified, use this space for an abbreviated title and expand the title with the AAX entry.

AAX NIMIS Comments Entry

1. Positions 1 through 4 of the entry must contain the literal "AAXM."

2. Positions 5 through 16 contain the NIMIS project identifier assigned by the DBA staff.

3. Positions 17 through 80 contain a free-form comment generally describing the NIMIS project. This entry is not intended to fully describe the project. That function is performed by the ABQ comments entry to be described later.

AAY NIMIS Date/Status Entry

1. Positions 1 through 4 of the entry must contain the literal "AAYM."

2. Positions 5 through 16 contain the project identifier assigned by the DBA staff.]

3. Positions 17 through 22 contain the date the project was originated. Format is YYMMDD.

4. Positions 23 through 28 contain the date when when the project was assigned to NIPSSA personnel for performance. Format is YYMMDD.

5. Positions 29 through 34 contain the date when the design portion of the project is to be delivered to the user. Format is YYMMDD.

6. Positions 35 through 40 contain the latest estimated date when the design portion of the project will be completed. Format is YYMMDD. ABQ comments entries must be prepared whenever a delivery date is modified.

7. Positions 41 through 46 contain the date the design portion of the project was delivered to the user. Format is YYMMDD.

8. Positions 47 through 52 contain the date the user desires to have an operational capability available. Format is YYMMDD. This date will be passed to a follow-on project if development is initiated by the user.

9. Position 53 defines the status of the project. Appendix E.2 provides a list of valid project status codes.

AAW NIMIS Miscellaneous Information Entry

1. Positions 1 through 4 must contain the literal "AAWM."

2. Positions 5 through 16 contain the NIMIS project identifier assigned by the DBA staff.

3. Position 34 contains the priority of the NIMIS project. Appendix E.3 defines the valid codes for this field.

4. Positions 35 through 41 contain the NAVINTCOM tasking identifier if the project originated through such a tasking.

5. Position 64 contains the security classification of the project. This includes the classification of all documentation prepared as a result of the project. Appendix E.4 lists the valid security codes for this field.

6. Position 67 defines the type of work to be performed by the NIMIS project. The code for design of a new application for the data base is "H." A complete list of codes is provided in Appendix E.5.

7. Positions 69 through 73 contain the initial estimate of labor resources required to perform the design phase of the

project. This field registers whole hours only and must be right justified.

8. Positions 74 through 78 contain the initial estimate of computer resource hours required to perform the design phase of the project. The field registers whole hours only and must be right justified.

ABG NIMIS Personnel Resources Assignment.

1. Positions 1 through 4 must contain the literal "ABGS" when assigning a person to a NIMIS project.

2. Positions 5 through 16 contain the NIMIS project identifier assigned by the DBA staff.

3. Positions 17 through 19 contain the literal "ALL."

4. Positions 20 through 29 contain the identifier of the person to be assigned. The NIPSSA person identifier (three characters), if used, is entered into the left-most positions of the field. The social security number of the person, if used, is entered into the right-most positions of the field.

ABQ NIMIS Extended Comments Entry

This entry permits the entering of detailed comments and description information about the project to be performed. A definition code identifies the type of comment entry

being entered. Entries are chronologically ordered within the entry type.

1. Positions 1 through 3 must contain the literal "ABQ."

2. Position 4 must contain the literal "S" for the first entry item of a comment group. Following entry items of a comment group must contain the literal "M."

3. Positions 5 through 16 contain the NIMIS project identifier assigned by the DBA staff.

4. Positions 17 through 22 contain the date of the entry. Format is YYMMDD.

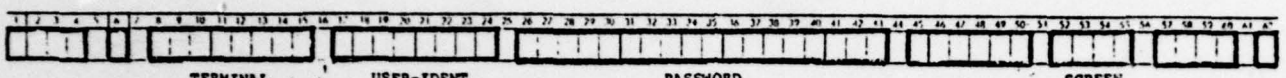
5. Positions 23 through 24 contain a sequence count of the number of groups entered for a specific date. The first entry of a comment group for a specific date is "01."

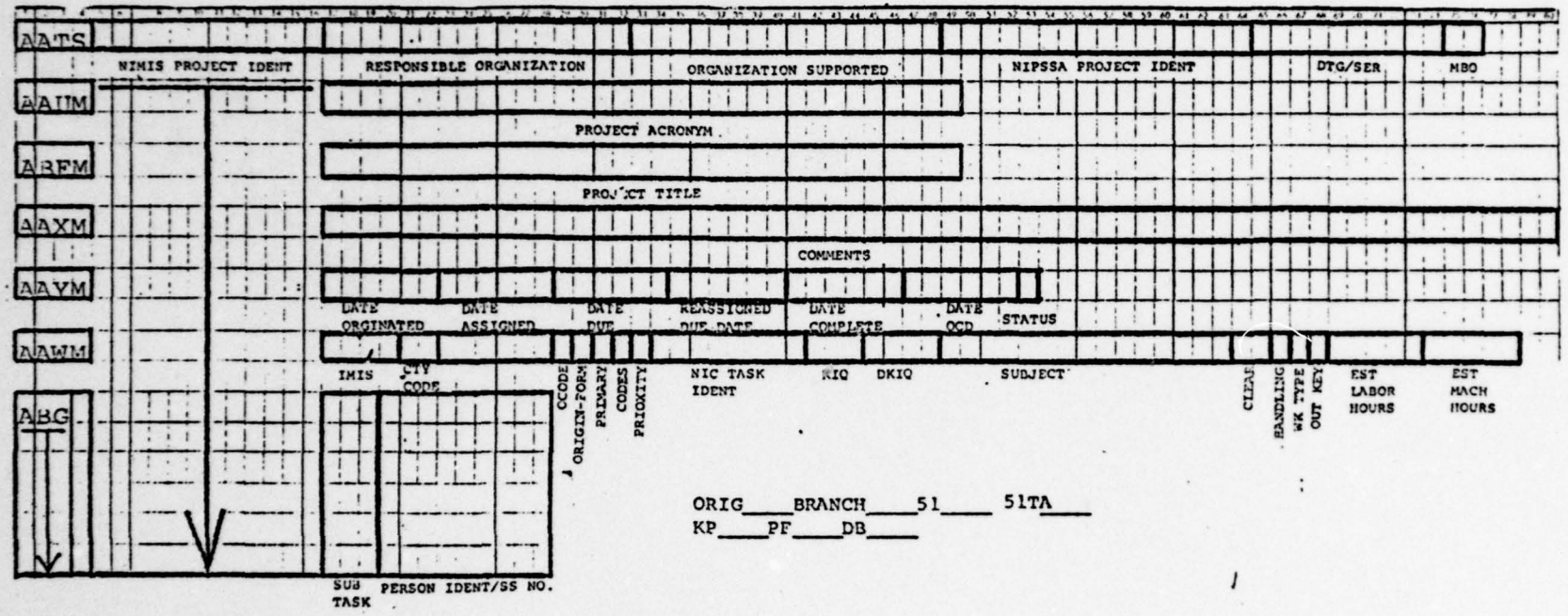
6. Positions 25 through 26 contain the sequence number of the line within the comment group. A comment group may contain a total of 10 lines of 60-characters each, totalling 600 characters of free-form comments. The first entry of the group must contain a line identifier "01."

7. Position 26 contains the comment type code. A list of these codes is provided in Appendix E.6. This code is entered in the first entry of a group only.

8. Positions 27 through 74 contain free-form comments.

NIMIS/NICOLS
NIMIS PROJECT DEFINITION

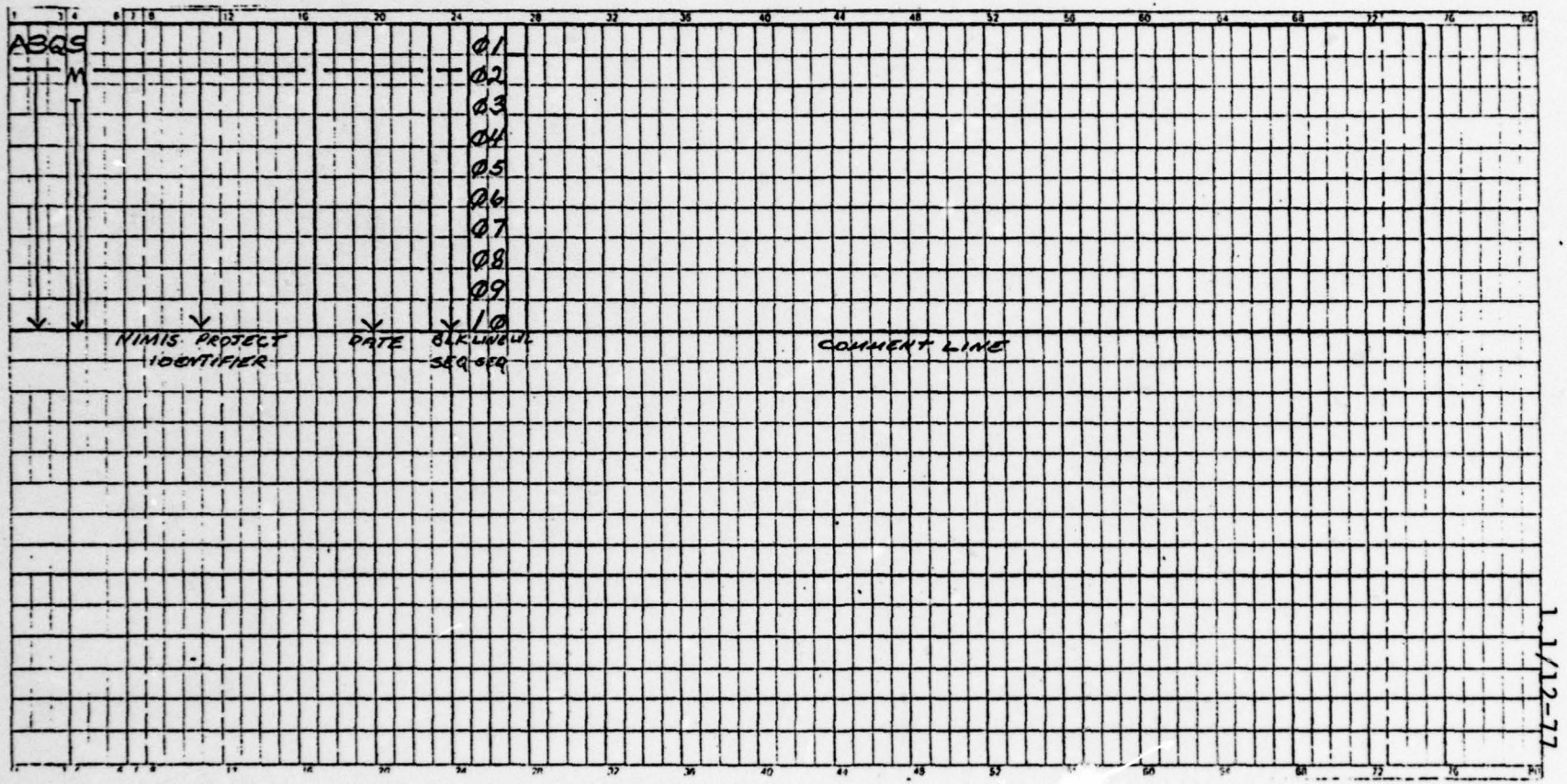




ORIG____BRANCH____51____ 51TA____
KP____PF____DB____

NIMIS COMMENTS ENTRY

PAGE 1 OF 1



1.1/12-77

APPENDIX A.4.

DATA DICTIONARY ENTRIES FOR SUB-SYSTEM DEFINITION

Each major application area of the data base is defined as a subsystem. This approach permits separating applications, users, and data into easily managed and controlled entities.

Subsystems developed as part of the integrated data base are all considered to be part of the integrated data base system as far as the data dictionary is concerned.

Figure A.4 illustrates the subsystem definition entries. When coding entries unused statements should be lined through to eliminate punching. The definitions for each statement follow:

1. ADD SUBSYSTEM NAME IS. This statement defines the internal name of the subsystem. The name is limited to eight characters in length and the first character must be alphabetic. The subsystem name is assigned by the DBA staff.

2. SUBSYSTEM DESCRIPTION IS. This literal field, enclosed in single quotate marks ('), provides an expanded name of the subsystem.

3. ENTRY-SECURITY IS. This statement defines the security classification of the subsystem description entries. See Appendix E.1 for a list of valid classifications.

4. COMMENTS. This section permits an unlimited free-form description of the subsystem. Its purposes and scope are

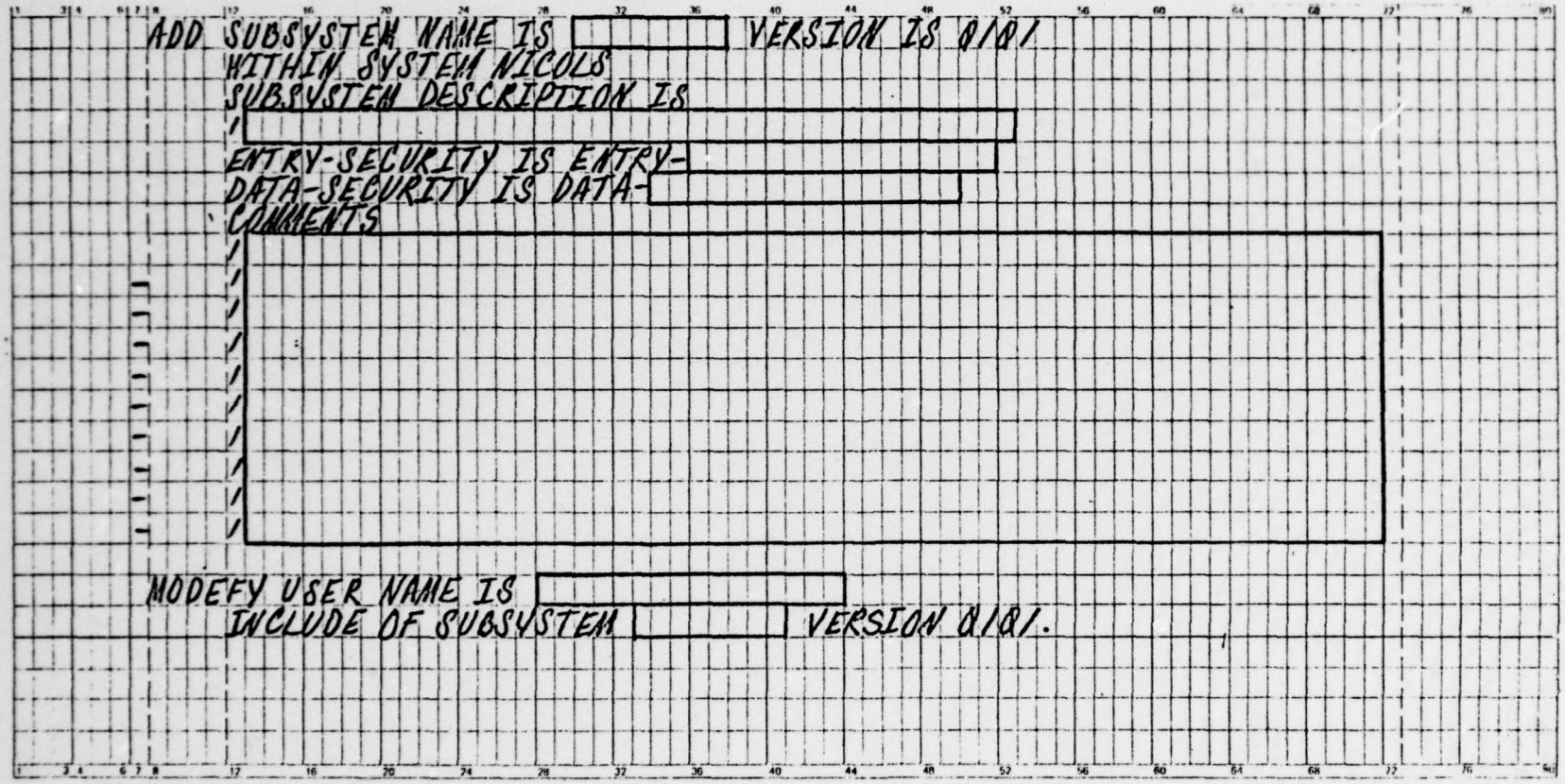
described in detail. Ten comment lines are provided on the coding sheet. As many additional lines as desired may be coded as long as all lines are entered into the data dictionary at the same time. The last line of comments must be terminated with a single quote mark.

5. PERIOD (.). A period must terminate the subsystem entry. This period may be placed immediately following the single quote mark at the end of the last comment line or on a separate line as shown in the illustration.

6. MODIFY USER NAME IS. This statement is used where the new subsystem will support a user already described to the data dictionary. The user name, as stored in the dictionary, is entered.

7. INCLUDE OF SUBSYSTEM. The subsystem name as defined in (1) above is entered.

ADD IDD SUBSYSTEM
DATA BASE DEVELOPMENT & DESIGN GUIDE
PAGE 1 OF 1



APPENDIX A.5.

DATA DICTIONARY USER/SUBSYSTEM ASSOCIATION ENTRY

When an existing user of ADP resources requests an existing data base capability it is necessary to authorize the user, through the data dictionary, to utilize the facility. Figure A.5.1 illustrates the IDD entries required.

1. MODIFY USER NAME IS. The user name as defined in the data dictionary is entered.

2. INCLUDE OF SUBSYSTEM. The name of the subsystem supporting the requested facility is entered.

3. VERSION 0101. If the version number of the current subsystem is other than 0101, the number must be changed to reflect the current version.

FIGURE A.5.1 TO BE ADDED

APPENDIX A.6.

DATA DICTIONARY DATA ELEMENT DEFINITION ENTRIES

The basis for all ADP System development is the data which will be stored in the data base to support the user's requirements. It is essential that each data element within the system be clearly and completely defined. The entries described below provide the facility to enter complete data element definitions into the data dictionary for use in analysis and design of the supporting subsystem.

This step is the most important of the steps resulting in a subsystem specification. It is best performed by the user of the proposed capability who can provide the most detailed information about data elements and their use. For this reason it is assumed that user personnel not familiar with ADP terms and technology will be preparing data element definitions. Figures A.6.1 and A.6.2 illustrate the entries to be used. Unused statements should be crossed out to reduce unnecessary punching.

Coding sheets, upon completion, should not be punched until after review with the DBA staff to eliminate inconsistencies and possible conflicts with existing data base definitions.

Prior to defining any data elements the user should keep in mind that the names assigned to the element should be as descriptive as possible, within the limits imposed by the

rules and conventions defined in Appendix B. This appendix should be thoroughly reviewed prior to beginning the data element definition process.

Two textual areas are provided in the definition entries. They are intended for similar but specific purposes. The element definition will be used as the basis for user documentation and coding instructions. The definition shown in this section must be very clear and specific and aimed at the non-ADP user at the experience level of the normal data preparer. The comments section contains more specific information about the data element which is useful to the designer and analyst. Specific ADP terminology may be used in this section of the description. During review of the data elements with the DBA staff additional comments may be entered by the DBA staff.

Individual entry descriptions are:

1. ADD DATA ELEMENT NAME IS. This name will be used for all future references to the data element. It is useful to enter the full data element descriptive name (item 4 below) prior to establishing the element ADP name. Using the guidelines provided in Appendix B, define the name within the 16 character limitation. No spaces or special characters may be used.

2. PREPARED BY. The initials of the preparer are

entered.

3. SAME AS ELEMENT. This field will be entered by the DBA staff if it is determined that the element is identical to an element already present in the data base. This entry is used only when the element is an exact duplicate of the existing data base entry.

4. ELEMENT DESCRIPTION IS. This field contains a literal, the expanded name of the data element. More than one word is permitted if this is required by the full element name.

5. ENTRY-SECURITY IS. This statement defines the security classification of the data element definition entries. Appendix E.1 identifies the valid security classifications for this entry.

6. DATA-SECURITY IS. This statement defines the security classification of the individual data element occurrences within the NICOLS data base. Appendix E.1 identifies the valid security classifications for this entry.

7. PICTURE IS. The user may enter this information in one of two ways, whichever is easier: a pencil entry of the number of characters followed by "alpha" for alphabetic, "numeric" for numeric data, or "AN" indicating either alphabetic or numeric data is allowed, is tne simplest method. The DBA staff will convert this notation to the ADP version

as described in Appendix B.3.

8. USAGE IS. This statement identifies the internal representation of the data element within the data base. It will be completed by the DBA staff. Appendix B.4 identifies the standards for this entry.

9. VALUE IS. This statement defines the initial value which is to be present in the data description used by programs accessing the data element. The field is completed by the DBA staff according to conventions defined in Appendix B.5.

10. JUSTIFY IS. This entry identifies non-standard justification of data within the data element. Normally data loaded to numeric fields is "right justified" which means it is loaded from the right-most position of the field to the left with leading zeros placed in unfilled left-most positions. Alphabetic or mixed data is loaded "left justified" which means that the data is loaded from the left-most position of the firld to the right with unused right-most positions cleared to blanks. The value "OFF" is entered in the statement if normal justification is desired for the data element. The value "ON" is entered if the justification is to be reversed.

11. ELEMENT DEFINITION IS. This entry contains a non-ADP definition of the data element and the manner in which

it is entered into the data base. The entry provides six lines but as many lines as required are permitted. All must be entered into the data dictionary together. It is particularly important that this definition be understandable by the person who normally enters such data into the data base. If fact, it is useful to ask the person entering the data to explain how the data is prepared and what it means and then use that explanation as the basis for this entry.

12. COMMENTS. This entry is used to further define the data element. More technical information or conditional situations affecting the data may be described here. Seven lines are provided on the coding sheet but as many additional lines as desired may be added as long as all are entered into the data dictionary at the same time. The last line of comments must be terminated by a single quote mark (').

13. DIA-REFERENCE-NO IS. This field identifies the relationship of the data element to a specific DIA reference. The purpose is to provide effective cross-referencing of the data element.

14. REFERENCE-PUB IS. This field permits the user to define publications which reference the data element and add to its definition or understanding. Multiple entries may be coded where more than one reference exists.

15. STANDARD IS. This field is used to define the existence of a standard definition of the data element. A list of applicable standards is provided in Appendix E.7.

16. USER IS. ThIs Statement identifies the user organization which is responsible for definition of the data element. This is the same as the user's organization identifier in the NIMIS.

17. RANGE IS. This statement, shown repeated several times, is used to define specific valid values which the data element may contain within the data base. Valid values entered are used to validate raw data submitted for data base updating. An unlimited number of range value statements may be prepared.

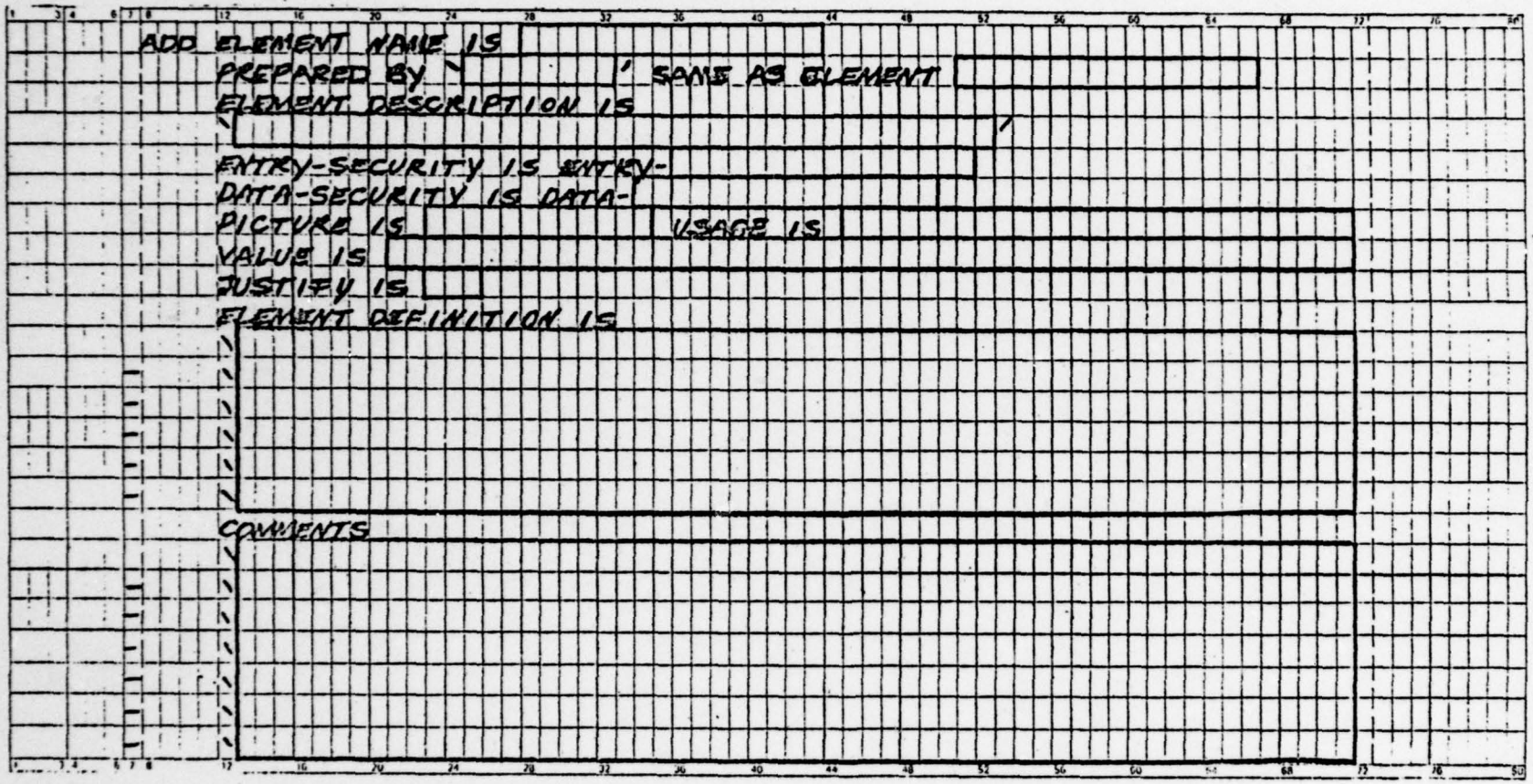
18. RANGE IS xxxxx THRU yyyyy. This version of the range statement permits the definition of two inclusive values which identify the valid values which may occur within the data element. Multiple statements may be prepared which define several ranges, including overalapping ranges.

NOTE: For both of the above range entries the range values must be enclosed in single quote marks (') if the value is not numeric. Where the second form of the range clause is used, the "xxxxx" value must be less than the "yyyyy" value.

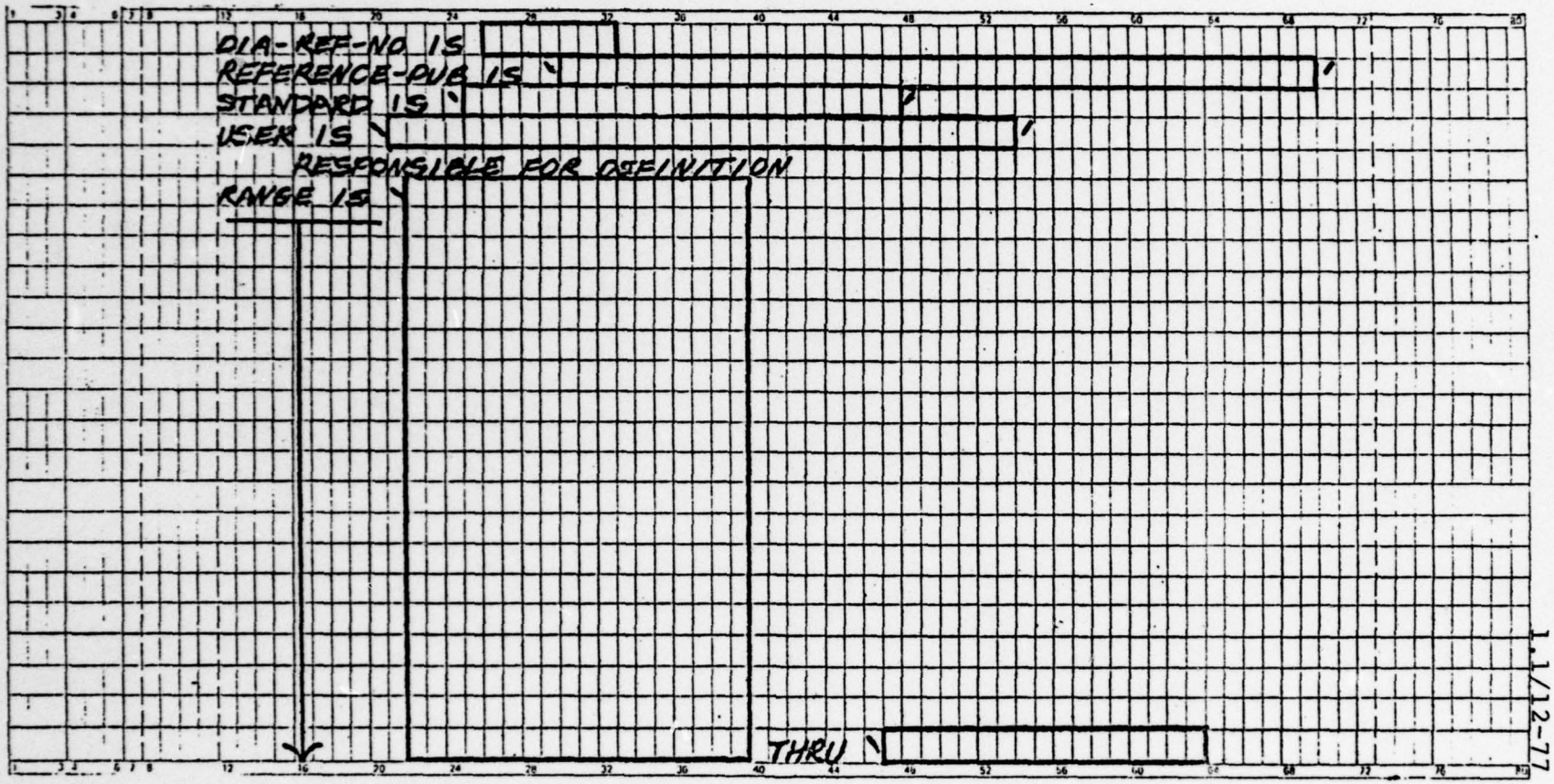
IT IS VERY IMPORTANT THAT A PERIOD BE PLACED AT THE END

OF THE LAST STATEMENT IN THE DATA ELEMENT DEFINITION TO TELL THE DATA DICTIONARY THAT THE DATA ELEMENT DEFINITION IS COMPLETE.

ADD IDD DATA ELEMENT
DATA BASE DEVELOPMENT & DESIGN GUIDE
PAGE 1 of 2



ADD IDD DATA ELEMENT
DATA BASE DEVELOPMENT & DESIGN GUIDE
PAGE 2 OF 2



11/12-77

APPENDIX A.7.

DATA DICTIONARY DATA ELEMENT GROUP DEFINITION

Data elements within an application are normally related in some manner. This relationship may be very loose, with the only connection being the application as a whole. Some elements, however, are very closely related. When this occurs it is desirable to link those elements within the IDD so that they cannot be inadvertently separated during the design process. Grouping also improves the organization of the data base and the usability of the stored data.

The easiest way to begin the grouping of data elements which were defined in Appendix A.6 is to physically group the coding sheets for related elements. Each group should be clipped together to prevent confusion. Once the groupings have been made each group should be reviewed to determine the ordering of elements within the group. While this order can be by any criteria the following order is suggested as a general guideline:

1. Key elements which determine an identity or ordering of the group, particularly if the group is repeated within the record occurrence.

2. Any other elements which make the group unique.

3. Any elements which will be used as entry pointers into the data base should be placed as close to the front of

the group as possible.

4. Dates.

5. Fixed length numeric fields.

6. Fixed length alphabetic fields.

7. Textual fields.

The effect of this ordering approach is to place fields which control the group at the beginning, followed by fixed fields which are likely to be completely filled by data, and ended by fields which may not be completely data filled. This organization optimizes the data compression feature used by the DBMS.

Figure A.7.1 illustrates the statements required to group data elements together. A detailed description of these statements follows:

1. ADD ELEMENT NAME IS. This entry assigns the name by which the grouped elements may be accessed as a whole. The name must conform to data name conventions defined in Appendix B. As with data elements it may be easier to define the full name of the group (step 3 below) and then establish the shorter group name.

2. PREPARED BY. The initials of the preparer are entered.

3. ELEMENT DESCRIPTION IS. This statement permits a more detailed expansion of the name of the group.

4. ENTRY-SECURITY IS. This statement identifies the security classification of the description of the data element group. Valid security classifications are defined in Appendix E.1.

5. DATA-SECURITY IS. This statement identifies the security classification of the contents of the data element group within the data base. Valid security classifications are defined in Appendix E.1. NOTE: The data security entry defines the aggregate classification of the grouped elements. The defined classification must be at least equal to the highest classification of any of the individual elements.

6. ELEMENT DEFINITION IS. This statement permits a detailed description of the data element group. The description should be in non-ADP terms which can be readily understood by the person who will prepare the data for entry into the data base.

7. DIA-REFERENCE-NO IS. This field permits associating a DIA reference to the data element group.

8. REFERENCE-PUB IS. This field identifies reference publications which are associated with the data element group. Multiple entries are permitted.

9. STANDARD IS. This field identifies a specified ADP standard which is related to the data element group. More

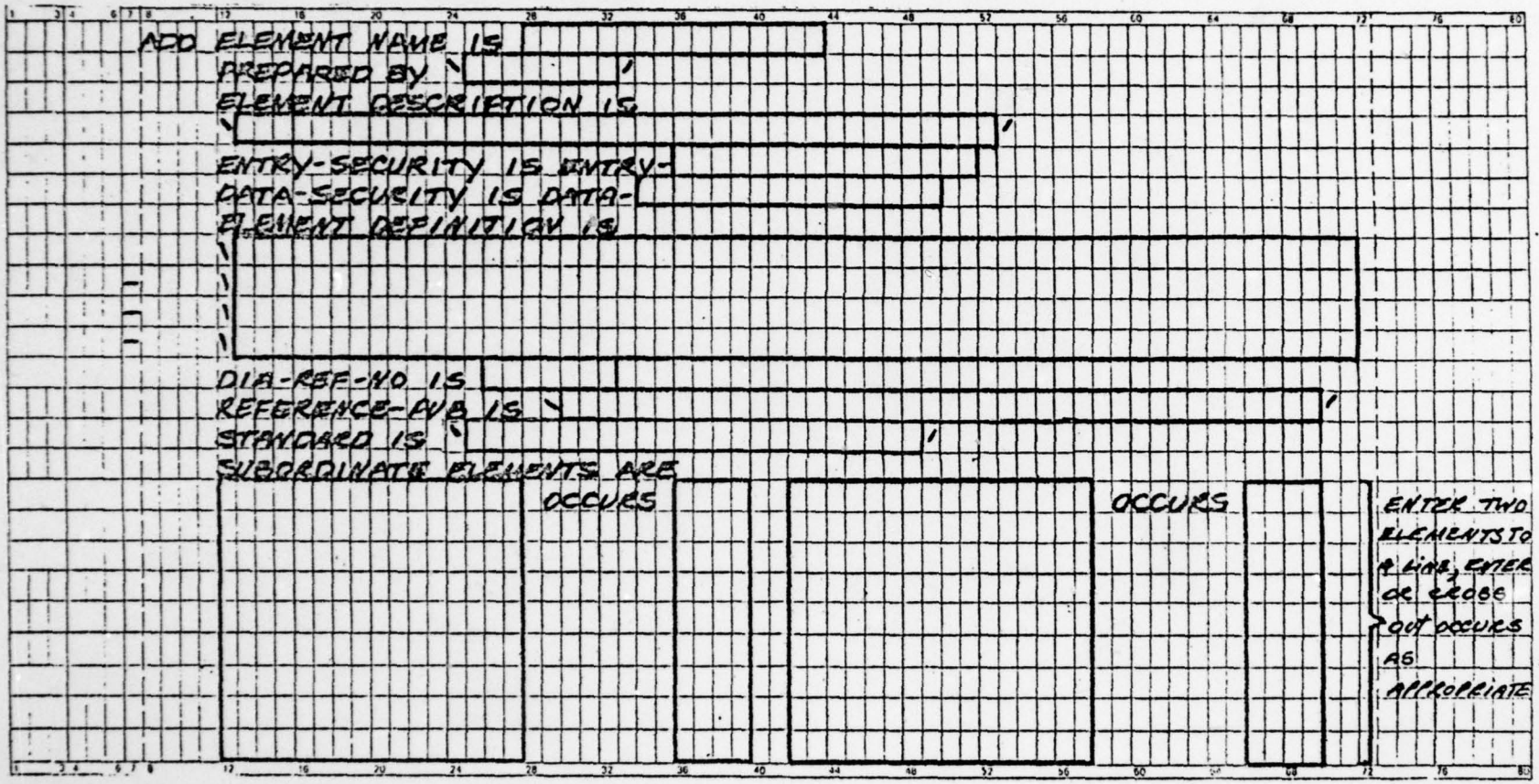
than one entry is permitted. Appendix E.7 identifies applicable standards.

10. SUBORDINATE ELEMENTS ARE. These statements define the individual data elements which comprise the group. The order in which the elements are placed in the group is determined by the order in which they are entered in this statement. The coding form provides for multiple occurring fields with the OCCURS statement. Cross out the word if the element does not occur more than once. Enter OCCURS as applicable followed by the number of time the element occurs if the element occurs more than once.

11. COMMENTS. The comments entry is not shown on the coding sheet. It may be coded in the same manner as the comments entry described in Appendix A.6 for data elements. The comments entry is specifically for the use of ADP-oriented information which is beneficial to the user and designer.

GROUP 100 DATA ELEMENTS
DATA BASE DEVELOPMENT & DESIGN GUIDE

PAGE 1 OF 1



APPENDIX A.8.

DATA BASE MODIFICATION REQUEST PREPARATION

It is important that all modifications to the operational data base be carefully evaluated and controlled. The data base modification request is the vehicle to assure that all such actions are properly handled.

The request is prepared as follows:

1. The entries in the upper right corner of the form are completed during review of the request. No action is required during preparation of the request.

2. The entries in the upper left corner of the form are completed during initial preparation:

a. SCHEMA. Enter the name of the data base subsystem which is to be modified.

b. VERSION. Enter the version number, four digits long. This will be provided by the DBA staff.

c. RECORD. Enter the record name if a data base record currently in the system description (schema) is to be modified.

d. ID. Enter the numberic identifier of the record named in (c) above.

e. REQUESTOR. Enter the name and telephone number of the person requesting the modification.

f. DATE REQUESTED. Enter the date the modification

request was prepared.

3. The entry lines "ABCS" and "ABDS" are completed at a later time by the DBA staff. These lines define the milestones and sub-tasks within a NIMIS project that will account for the requested modification.

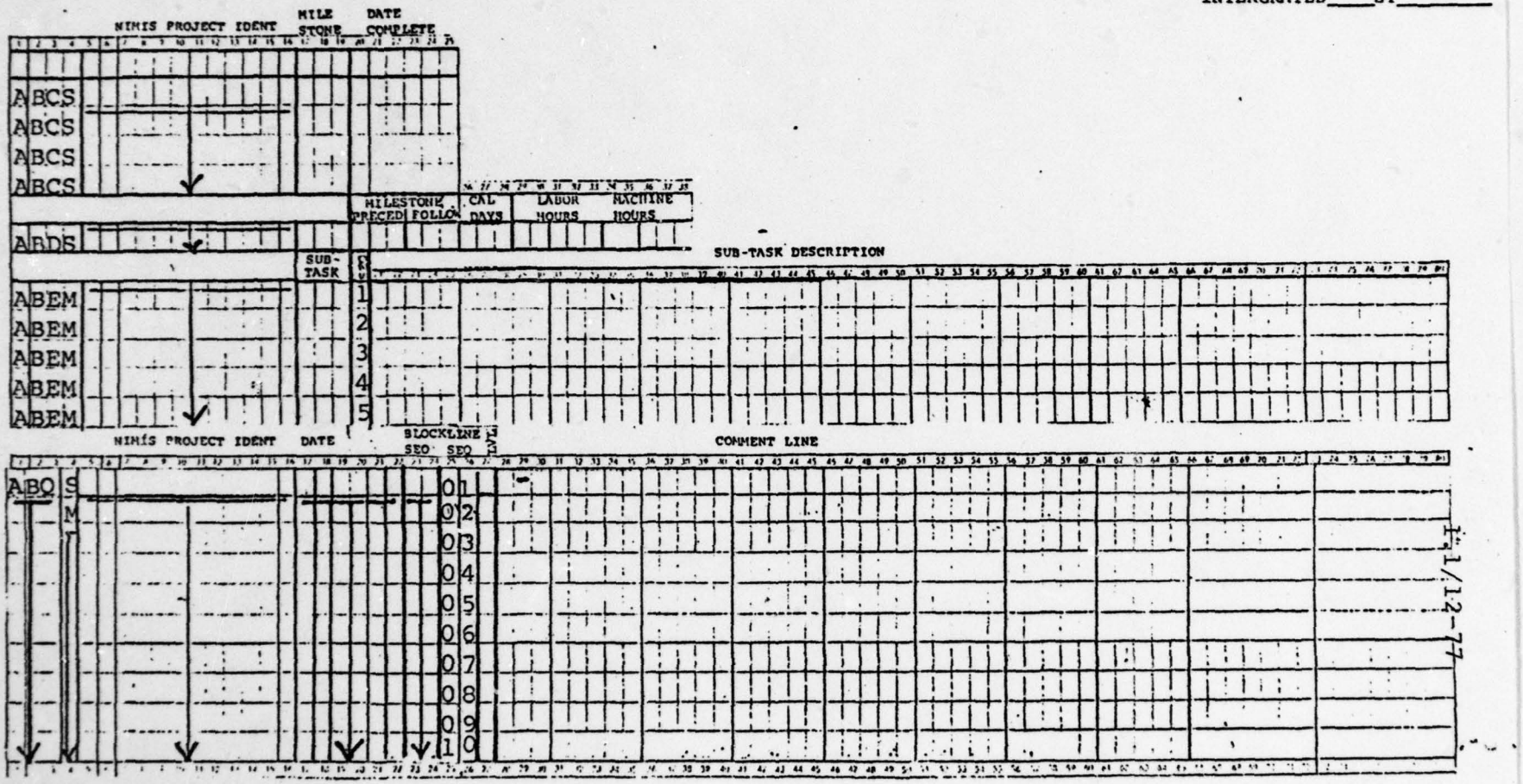
4. THE "ABEM" lines provide a detailed description of the technical functions to be performed by the modification. These lines describe the sub-task generally but sufficiently to define the level of resources and the end result desired.

5. THE "ABQ" lines permit additional and more detailed description of the modification to be performed. This group is entered into NIMIS as a comment modifying a system update project. The narrative description begins in position 31 of the first line and continues free-form through position 73. Subsequent lines begin at position 26. The date of the modification request is entered in positions 17-22 of the first line (format YYMMDD). A second group of ABQ lines may be prepared if additional space is required to fully describe the modification request. If this is done, increment the sequence within the date for each additional comment group prepared.

INTEGRATED DATA BASE MODIFICATION REQUEST

SCHEMA_____VERSION_____
RECORD_____ID_____
REQUESTOR_____
DATE REQUESTED_____

SCG ACTION
RECEIVED_____BY_____
APPROVED_____BY_____
MODIFIED_____BY_____
REJECTED_____BY_____
INTERGRATED____BY_____



A.41

11/12-77

APPENDIX A.9.

DATA DICTIONARY RECORD DEFINITION ENTRY

The final act of associating data elements and data element groups is to organize these elements and groups into records which will identify the data to the data base itself.

Figure A.9.1 illustrates the data dictionary entries required to create a record definition. Fields and statements are prepared as follows:

1. ADD RECORD NAME IS. This statement identifies the record to the data dictionary. It should be named as descriptively as possible. All users of the record within the data base will refer to the record by this name. The name must conform to the basic naming conventions defined in Appendix B. It may be useful to describe the record in its expanded name (see item 4 below) first and then establish the ADP name.

2. PREPARED BY. The initials of the preparer are entered.

3. SAME AS RECORD. This field is completed by the DBA staff. It is used only if the defined record is identical in all ways to another record present in the data base.

4. RECORD DESCRIPTION IS. Tnis entry provides the facility for an expanded record name which is more descriptive.

5. ENTRY-SECURITY IS. This field defines the security

classification of the record entry within the data dictionary. Valid security classifications are described in Appendix E.1.

6. DATA-SECURITY IS. This field defines the security classification of the record data stored within the data base. Valid security classifications are described in Appendix E.1. NOTE: The data security entry defines the aggregate classification of all elements within the record. The defined classification must be at least equal to the highest classification of any of the individual or grouped elements.

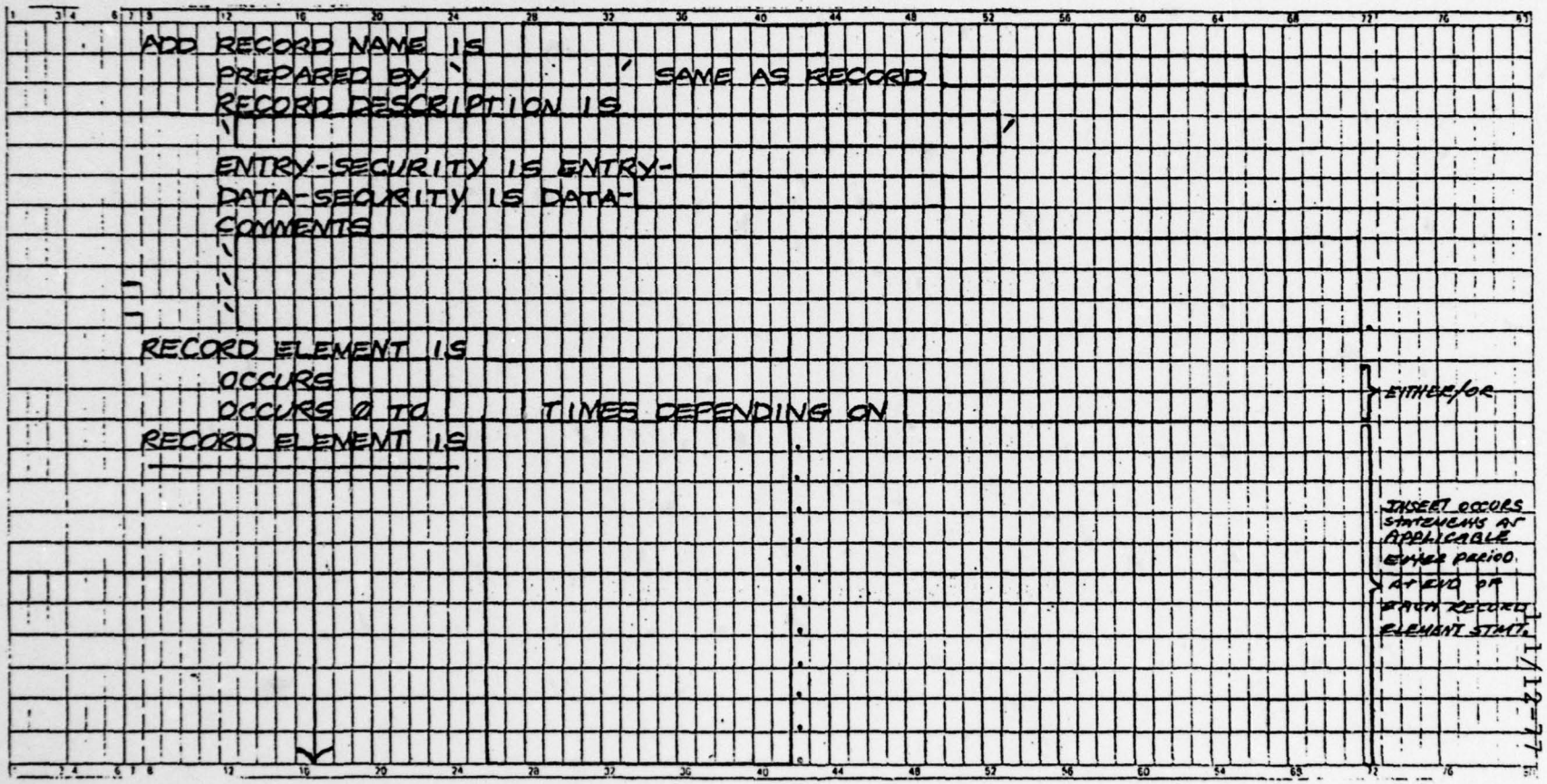
7. COMMENTS. The comments entry on the coding sheet allows for three lines of entries. If additional lines are required, an unlimited number may be coded as long as they are all entered into the data dictionary together. The last line of comments must be terminated by a single quotation mark ('). Comments should fully describe the record and its purpose within the data base. Technical verbage should be avoided as these comments will be available to system users who wish to find out more about the record prior to accessing the data base.

8. RECORD ELEMENT IS. Three forms of the statement are shown. All three may be used to describe the elements and groups associated with the record. It is recommended that elements or groups which occur a specific/fixed number of

times be located near the end of the record. Elements or groups whose occurrences depend on a variable value MUST be placed at the end of the record. Only one OCCURS DEPENDING ON element or element group may be defined in a record. The data element upon which the occurrence count depends must be the first active element in the record and must conform to the data name rules in Appendix B.

Each record element statement is a complete sentence. It must be terminated by a period. Each statement names a single data element or data element group. The order within the record is as they are named in the entry.

ADD 100 RECORD
DATA BASE DEVELOPMENT & DESIGN GUIDE
PAGE 1 OF 1



11/12-77

APPENDIX A.10.

DATA BASE SCHEMA RECORD DEFINITION

Once a record definition has been prepared and stored in the data dictionary, an association of the record to the actual data base structure must be performed. This appendix describes that preparation. Figure A.10.1 illustrates the format for record definition.

Individual statements are complete sentences and must be terminated by a period. Instructions for statement preparation are:

1. The first three lines are comment lines to the data base compiler and provide visual separation between record definitions.

2. RECORD NAME IS. This statement contains the ADP name of the record as defined in data dictionary entries prepared using Appendix A.9. The statement is terminated by a period.

3. RECORD ID IS. This statement is completed by the DBA staff upon assigning a four-digit identifier to each record defined within the application. A period follows the number.

4. LOCATION MOOE IS. This statement identifies the manner in which the physical data record occurrences will be stored in the data base. Appendix B.6 defines the criteria

for assignment of location modes. If the "CALC" location is specfied, a decision must be made whether to allow duplicate records (records with the same value in the direct location key field). The clause "DUPLICATES NOT" completes the location mode statement if each occurrence of a CALC record is to be uniquely identified.]

When duplicate records are permitted they may be stored either FIRST or LAST. When they are stored FIRST each additional duplicate record stored is placed at the front of the list of duplicate record occurrences. This has the effect of last-in-first-out (LIFO) where the most recent occurrence is retrieved first. Duplicate occurrences are stored at the end of the record list if the records are stored LAST. The affect is then first-in-first-out (FIFO). The location mode is completed with either "DUPLICATES FIRST" or "DUPLICATES LAST" if either of these options are chosen.

Where practical, it is desirable to use the DUPLICATES NOT option. This permits the data base system to perform a measure of input data validation against incoming data as compared to the existing data base.

5. WITHIN xxxxx AREA. This statement identifies the logical area of the data base in which the record will be stored. Most applications will use a single area of the data base. This area will be assigned by the DBA staff for

exclusive use of the application subsystem.

6. The MINIMUM ROOT and MINIMUM FRAGMENT statements are used to establish control parameters for variable length records. These statements may be omitted if the size and/or composition of the record make variable-length unnecessary. CALL statements following which specify IDMSCOMP or IDMSDCOM are also omitted. The DBA staff should make the final decision whether to utilize variable-record structuring.

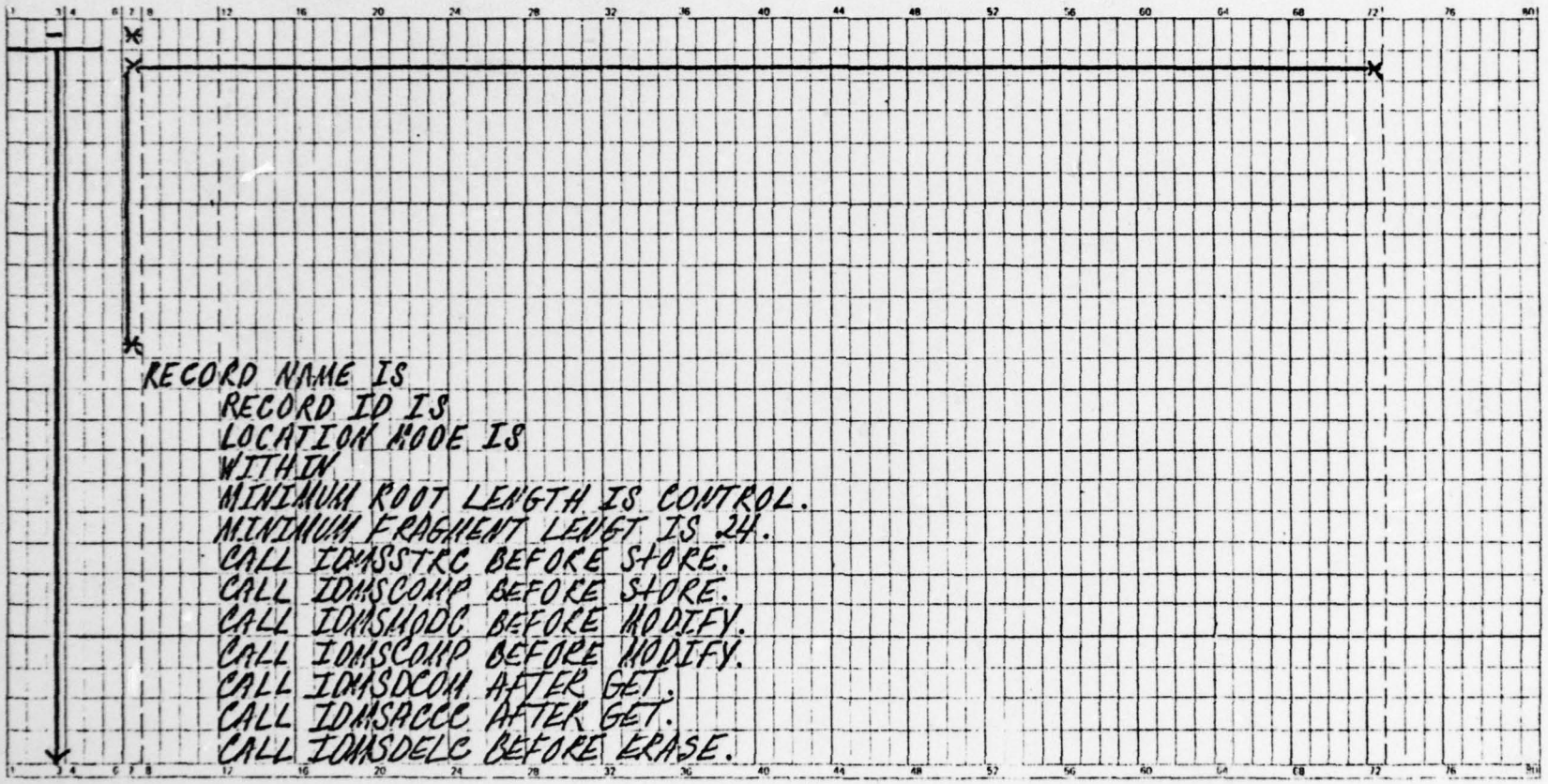
7. The CALL statements listed perform special functions to the data base records during input and output operations. As described above the IDMSCOMP and IDMSDCOM routines are used with variable-length records and perform data compression and decompression functions to conserve physical storage space on disk. The four remaining CALL operations: IDMSSTRC, IDMSMODC, IDMSDELC, and IDMSACCC are used to control access to records which have been defined as protected. A protected record is one in which each occurrence of the record is stored with a key identifying the user who stored the record. Access to and the ability to manipulate that record occurrence is limited by the security functions of the NICOLS system. If the protection feature is not implemented for a particular record type these CALL statements are omitted.

8. COPY xxxxx RECORD VERSION 0101. This statement instructs the schema compiler to copy the detailed data element

and element group definitions of the record defined using Appendix A.9 as part of the schema record definition.

SCHEMA RECORD DEFINITION

90-10-01



APPENDIX A.11.

DATA BASE SCHEMA SET DEFINITION

The data base system relates different record types through the use of "sets". This relationship defines the manner and conditions upon which a relationship is established between physical occurrences of related record types in the data base.

The establishment of sets linking different record types is one of the most important functions in designing the data base structure. The capability of the data base to support the user's requirements is dependent upon proper association of records.

When a set is specified between two record types, certain relationships are established:

1. One of the records is defined as the owner or primary record and the other as the member or subsidiary record.

2. The conditions upon which the relationship is established are determined:

a. The member may be automatically associated with its owner record occurrence when a member occurrence is stored in the data base. This is known as "AUTOMATIC" membership.

b. The member may be associated with the owner as a result of conditional situations which cannot be predetermined. This is known as "MANUAL" membership.

3. The conditions upon which the relationship is maintained are defined:

a. Once an occurrence of a member record is attached to an occurrence of its owner the membership cannot be broken except by physically deleting the member record from the data base. This is known as "MANDATORY" membership.

b. An occurrence of a member record may be removed from its association with an owner record occurrence and attached to another occurrence of the owner record type as a result of conditions which may occur. This is known as "OPTIONAL" membership.

4. The order of the member record occurrences as related to the owner record occurrence:

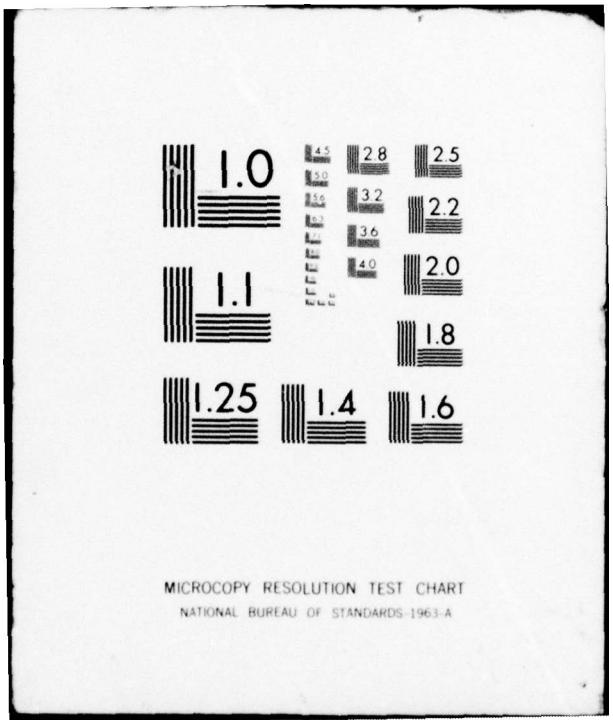
a. The member records may be sorted in ascending or descending order on a data element within the member record. The presence of duplicate values within the ordering data element may be controlled, denying any duplicates or storing duplicate occurrences at the front or back of the group of duplicates. If possible, it is desirable to deny duplicates, achieving a measure of input data validation by the system and improving the simplicity of retrieval.

b. The member record occurrences may be stored in the series at the very beginning or end of the member list by defining the order as "FIRST" or "LAST", respectively.

AD-A050 468 NAVAL INTELLIGENCE PROCESSING SYSTEM SUPPORT ACTIVITY--ETC F/G 5/2
INTEGRATED DATA BASE DEVELOPMENT AND DESIGN GUIDE. VERSION 1.1.(U)
DEC 77 L E TOWNER

UNCLASSIFIED NL

2 OF 3
AD A050468


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

c. The member record occurrences may be stored at the next available position at the front or rear of the member list by defining the order as "NEXT" or "PRIOR", respectively.

5. If the member record may be accessed through another set or directly and the information stored in the owner is frequently desired too, processing speed is improved by specifying "OWNER" pointers to access the owner record occurrence directly without having to read the rest of the member records in the set.

Each of these conditional associations is important to the overall efficiency and effectivenss of the data base as a resource of information. Each must be carefully weighed before completing the schema definition phase.

Figure A.11.1 illustrates the coding format used to define sets for the data base compiler. Instructions for preparing the information are:

1. The top area of the coding form provides a comment area for definition of the set. These comments should define the purpose of the set and the conditions under which it is established.

2. SET NAME IS. This statement, terminated by a period, defines the ADP name of the set as used within the data base system. Appendix B.7 defines the conventions used to establish set names.

3. ORDER. This statement defines the order relationship between the owner and member record occurrences and the ordering of related member occurrences. The terms available are: NEXT, PRIOR, FIRST, LAST, and SORTED. The term is followed by a period.

4. MODE IS CHAIN LINKED TO PRIOR. This statement identifies the method of associating owner and member records. This version indicates that a chaining technique is employed which links records both forward and backward. While the backword linkage (LINKED TO PRIOR) is optional it is recommended for a majority of cases. Consult the DBA staff for exceptions.

5. OWNER IS. This clause in the OWNER statement defines the name of the owner record type. The record type must be one defined as belonging to the new application unless approved by the DBA staff.

6. NEXT DBKEY POSITION IS. This clause defines internal record pointer assignments. These assignments will be made by the DBA staff.

7. PRIOR DBKEY POSITION IS. This clause also defines internal record pointer assignments. The clause must be present if the LINKED TO PRIOR clause defined above is present. The pointer assignment is made by the DBA staff. A period must follow the clause. If the clause is omitted,

a period must follow the NEXT clause above.

8. MEMBER IS. This is the first clause of the member statement and defines the name of the member record in the set. Convention requires that the member be located in the same data base area as its owner record type. Any exception to this convention must be approved by the DBA staff.

9. MANDATORY/OPTIONAL AUTOMATIC/MANUAL. This clause defines the conditions under which association is established between an occurrence of the owner and member record types. Any combination of the two groups is permitted but at least one of each group must be selected.

10. NEXT DBKEY POSITION IS. As with the owner record this clause defines internal record pointer assignments and] is performed by the DBA staff.

11. PRIOR DBKEY POSITION IS. The prior pointer clause is required when LINKED TO PRIOR is selected. The DBA staff will assign the pointer values.

12. LINKED TO OWNER. This clause, along with the next clause is required if a direct owner pointer capability is desired to increase retrieval speed.

13. OWNER DBKEY POSITION IS. The clause is used to define internal pointers when the owner pointer option is employed. The DBA staff also assigns this pointer value.

14. ASCENDING/DESCENDING KEY IS. This clause is used

whenever the order option is SORTED. ASCENDING instructs the system to order member records in ascending order by the field named following IS. The field named must be defined within the member record type and must not be a field which occurs more than once.

15. DUPLICATES. This part of the key clause defines the manner in which member records are to be handled when the key field contains a value duplicated by a record already stored in the data base. The duplicated record will be rejected if DUPLICATES NOT is selected. The duplicated record will be placed after its duplicate in the data base if DUPLICATES LAST is selected.

NOTE: The last clause in the member statement must be terminated by a period.

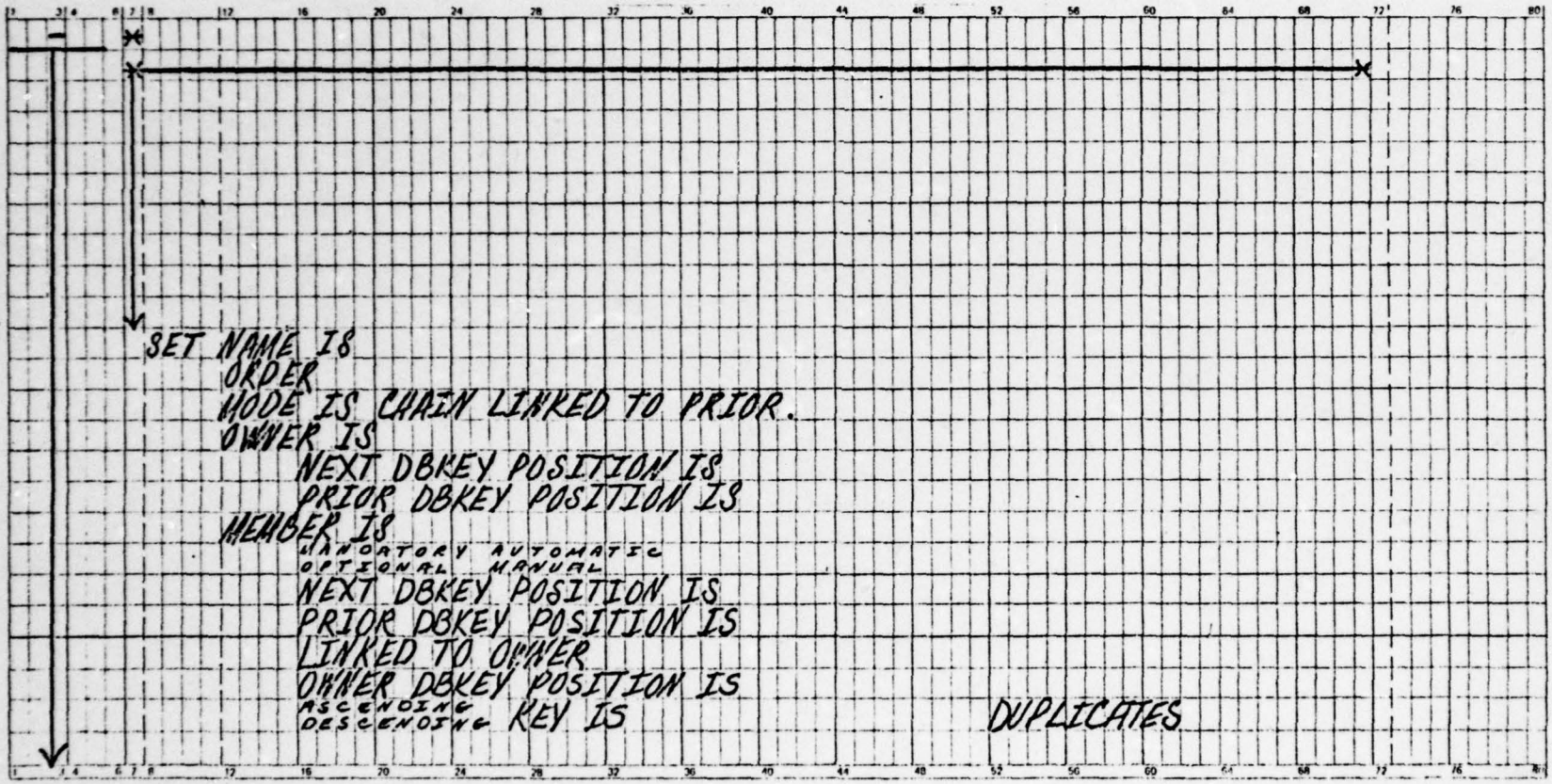
Multiple member statements may be written for the same set. This has the effect of placing multiple member record types in the same string. This is useful when two or more member record types are similar in definition and use. It has the advantage of saving storage overhead space by reducing the number of pointers required within the owner record type.

The use of multiple member sets should be avoided where any one member record type may have many occurrences stored in the data base. The combination may produce a large quantity

of record occurrences within one list and impact system performance.

SET DESCRIPTIONS

90-10-01



APPENDIX A.12.

REPORT SPECIFICATION PREPARATION

All ADP systems require the preparation of certain specific output products. One of these products is the printed report. Printed reports are used in a different manner and often are designed to emphasize that difference when used in conjunction with on-line terminal-supported retrieval.

The types of reports which should be considered during the design of a new application for the integrated data base are:

1. Reports for use in publications. These reports will be structured for ease of reading and continuity. They may contain special narrative and other enhanced features to increase their usefullness as a reference document.

2. Reports for special purposes. These reports, usually with more information than can be easily displayed on a terminal screen, provide a synopsis of data base information.

3. Reports for verifying the data base. This type of report is commonly developed during the initial implementation of a data base application. They assist in determining the accuracy of software developed to store information in the data base. Following development these reports are used to assist the user in verifying the contents of the data base.

The relative complexity of the reports described above

range from first to last with the first report being the most complex to develop. Reports of the first type are generally written in a compiler language such as COBOL. The second type of report may be written in COBOL and in CULPRIT depending on its level of complexity. The third type of report is usually be written in CULPRIT.

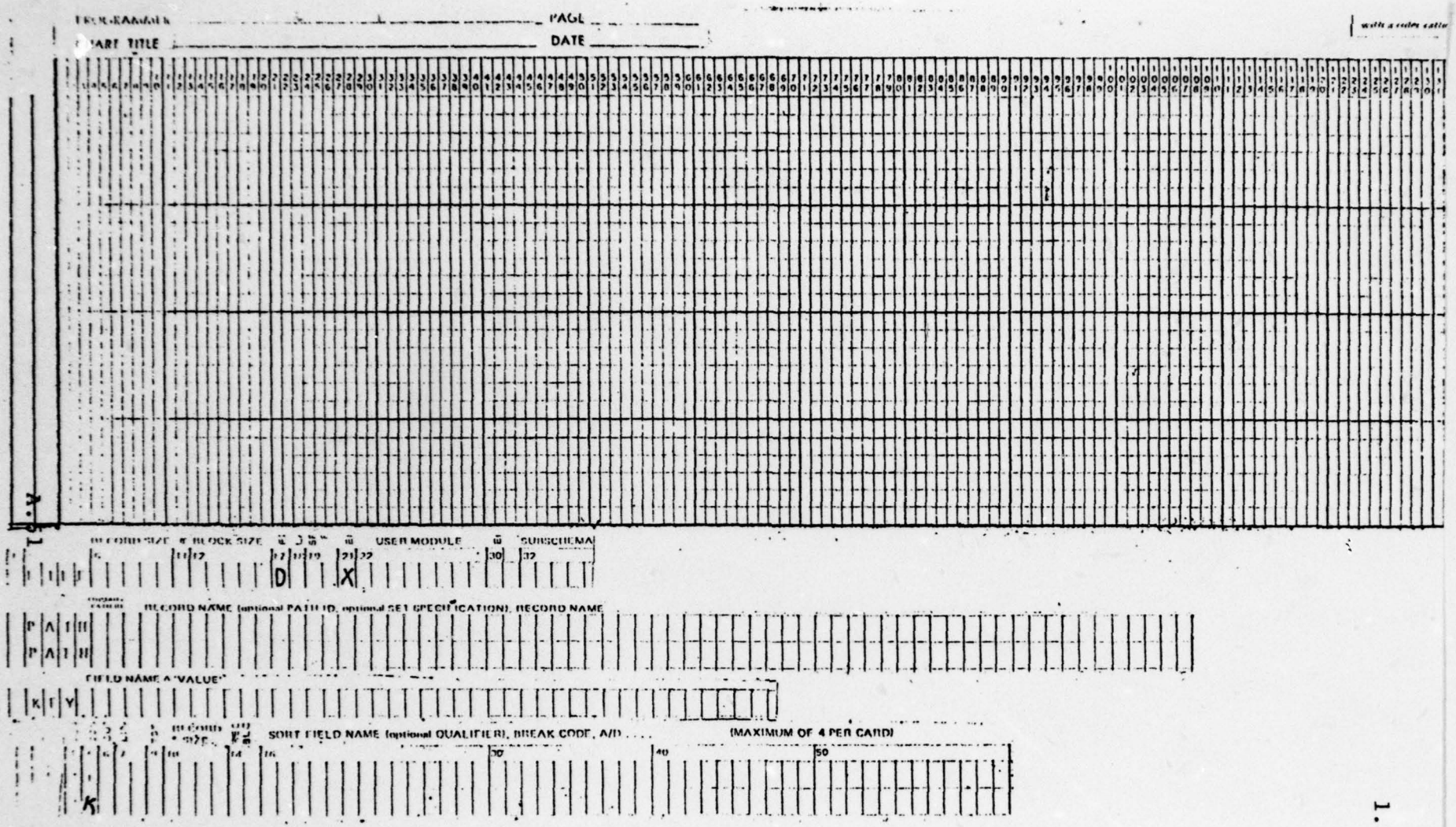
Flexibility is important when designing reports. Where possible it is wise to design the report in a modular fashion, using common lines and procedures wherever possible. This approach not only produces a series of reports which appear uniform but reduces the development cost and time. Prior to developing the body of any reports it is recommended that a standard report heading and title configuration be adopted. This standard heading should be illustrated on a report development form such as shown in Figure A.12.1. A sample of such a heading is illustrated in Figure A.12.2.

Figure A.12.1 is designed to be used as a specification for either COBOL or CULPRIT reports. The fields at the bottom of the sheet are CULPRIT coding entries. However, the information defining subschema, record names of records to be accessed, sort fields, and specific key records to be accessed are usable for COBOL reports as well.

The data dictionary program entry, described in Appendix A.13, should be prepared in conjunction with the specification

sheet.

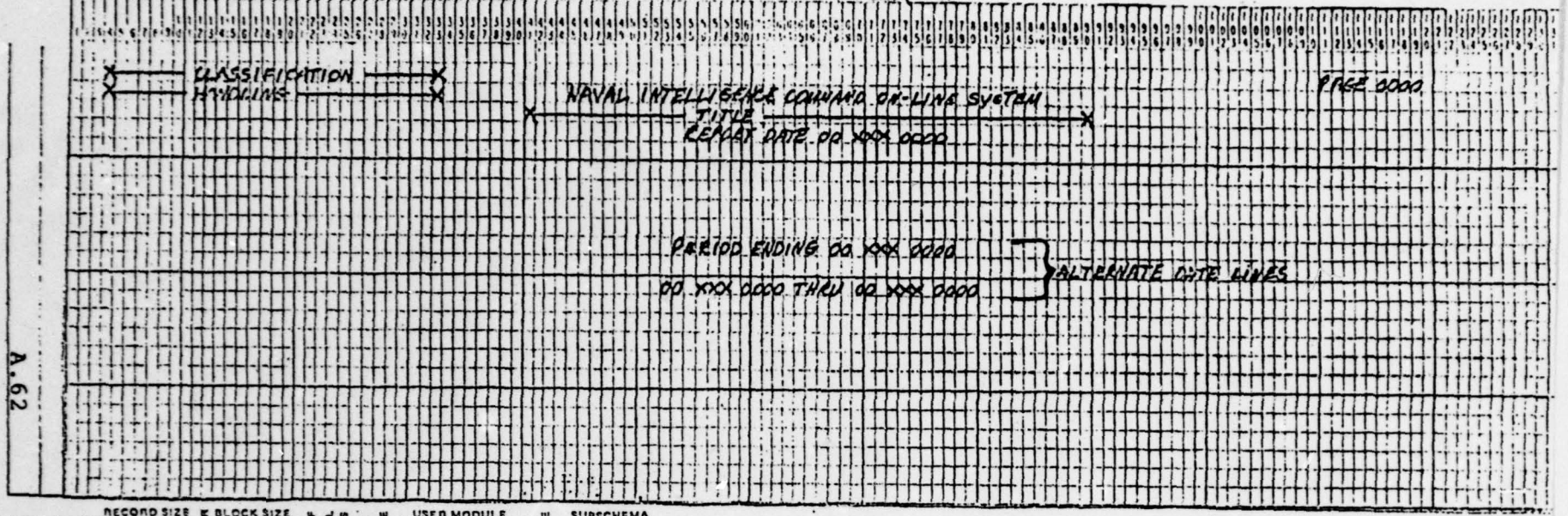
Figure A.12. illustrates a report specification which has been completed defining a simple CULPRIT report.


PROGRAMMER
PAGE
PART TITLE
DATE
A.01
RECORD SIZE
BLOCK SIZE
USER MODULE
SUBSCHEMA
D
X
RECORD NAME (optional PATH ID, optional SET SPECIFICATION), RECORD NAME
FIELD NAME A "VALUE"
RECORD SIZE
SORT FIELD NAME (optional QUALIFIER), BREAK CODE, A/D . . . .
(MAXIMUM OF 4 PER CARD)
K

1.01/11-77

PROGRAMMER _____ PAGE _____

CHART TITLE STANDARD HEADINGS DATE _____



A.62

RECORD SIZE | BLOCK SIZE | USER MODULE | SUBSCHEMA

FILE D X

RECORD NAME (optional PATH ID, optional SET SPECIFICATION), RECORD NAME ...

PATH

PATH

FIELD NAME = 'VALUE'

KEY

REPORT NO.	PARAMETER	CONTINUATION	OUTPUT TYPE	LINES/PAGE	TOTAL/DETAIL	RECORD SIZE	RECORDS/ BLOCK	SORT FIELD NAME (optional QUALIFIER), BREAK CODE, A/D .... (MAXIMUM OF 4 PER CARD)
		J						
Punch on all	2	K						

1.1/12-77

APPENDIX A.13.

DATA DICTIONARY REPORT PROGRAM DEFINITION PREPARATION

Each report program must be defined to the data dictionary to complete the picture of the scope of an application. Figure A.13 illustrates the format of this entry. The specific clauses of the entry are:

1. ADD PROGRAM NAME IS. This clause defines the ADP name assigned to the report program by the DBA staff.

2. PREPARED BY. The initials of the preparer are entered.

3. SAME AS PROGRAM. The ADP name of the identical program is entered if the requested program already exists identical to the user's request. This field is used exclusively by the DBA staff. VERSION of the identical program is entered.

4. PROGRAM DESCRIPTION IS. This entry clause provides the full title of the report. If the title is greater than the space available, the title is shown fully under COMMENTS below and abbreviated in this space.

5. ENTRY-SECURITY IS. The security classification of the program description is entered. See Appendix E.1 for a list of acceptable classification statements.

6. DATA-SECURITY IS. The security classification of the data within the data base accessed by the report program is

indicated. See Appendix E.1 for a list of acceptable classification statements.

7. LANGUAGE IS. The computer language which is used to develop the report is entered. See Appendix E.8 for a list of applicable languages.

8. OUTPUT-SECURITY IS. The security classification of the output report is entered. This classification will be printed on the pages of the report unless otherwise directed in the comments below. The classification prefix OUT- is dropped when printing the classification. See Appendix E.1 for a list of applicable classification statements.

9. WITHIN SUBSYSTEM. This clause identifies the subsystem which the report program supports.

10. SUBSCHEMA IS. The DBA staff will assign a subschema to support the program. OF SCHEMA defines the data base description which applies to the subsystem. VERSION identifies the version of the data base description to be used. This information is provided by the DBA staff.

11. PROGRAM-TYPE IS REPORT. This clause defines the program as a report in contrast to other possible program types.

12. COMMENTS. This clause is used to provide a general description of the report program as well as a number of specific items of information. All of the entries are considered comments by the data dictionary.

13. PURPOSE. This comment should describe the purpose of the report program in general non-ADP terms where possible. It is important to fully identify the underlying reasons why the report is being prepared. This will assist analysts in determining the best approach to preparing the report output.

14. DISTRIBUTION nn COPIES. This clause defines the average number of copies of the report which will be printed each time the report is prepared. Where this number exceeds three an explanation should be provided in the general comments section.

15. OUTPUT FORM. This clause identifies the type of paper or form to be used to prepare the report. Appendix E.9 lists the readily available output forms and paper type. If a special form is required, enter "SPECIAL" and describe the form in detail in the general comments section.

16. AVERAGE SIZE nnn PAGES. This clause identifies the anticipated size of the prepared report. An explanation must be provided in the general comments section if the report size is expected to exceed 100 pages.

17. LEVELS OF TOTALS. The number of levels of totals which will be produced should be indicated if the report produces totals of data within the data base.
For example: If totals will be produced by organization, person, and date, this is three levels of totals. The data

base element which will be used as a basis for total determination must be defined in the general comments section. The spacing and page ejection requirements for each level of totals should also be defined in the general comments section .

18. DATE RANGE. This clause defines the date range to be used in extracting data for report preparation. Enter]
"VARIABLE" if the range is variable. Enter "INCLUSIVE" if the range is inclusive of two dates. Enter "NONE" if no determination is required. Describe the method which is used to determine the date range in the general comments section.

19. STANDARD COVER PAGE. This clause defines whether the standard NIPSSA cover page is to be produced as part of the report. Indicate either "YES" or "NO". A description of of the reason must be provided in the general remarks section if "NO" is entered. It is normal to produce the NIPSSA cover]
page.

20. SPECIAL COVER PAGE. This clause identifies the user's wishes with regard to a special cover page to be printed]
following the standard cover page. Enter either "YES" or "NO". If "YES", describe the cover page in the general comments section.

21. CLASSIFICATION CODEWORD REQD. This clause determines whether the program will produce output which

requires special security handlIng. Enter either "YES" or "NO". If "YES", define the source of the codeword in the general comments area.]

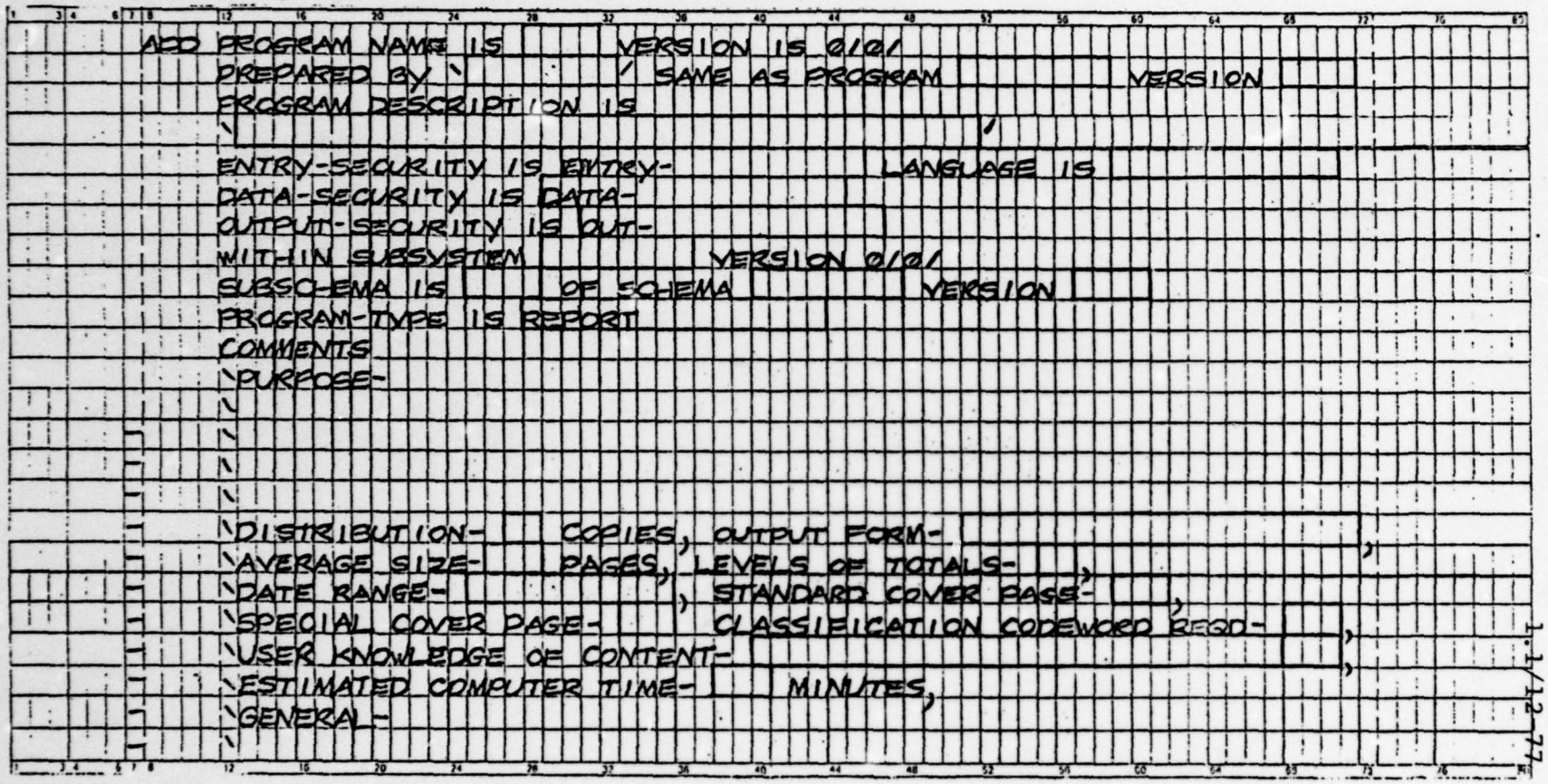
22. USER KNOWLEDGE OF CONTENT. This clause is to be used by analysts in the design of the report. The user described is the end user of the product. This may or may not be the same person/organization as the data base application user. A short statement may be adequate to answer the question. If it is not, expand on the statement in the general comment section.]

23. ESTIMATED COMPUTER TIME nnn MINUTES. This clause defines the estimated number of minutes of processing which will be required to produce the report each time the program is executed. If the estimated time exceeds 60 minutes a special explanation is required in the general comments section.

24. GENERAL. The general comments section may be continued with as many additional lines as required to fully describe and expand on the entries above. Each comment which expands on an entry above should begin with the name of the entry and be indented four spaces to improve readability. The last line of the general comments entry must be terminated by a single quote mark followed by a period ('.).]

ADD IDD REPORT PROGRAM
DATA BASE DEVELOPMENT & DESIGN GUIDE

PAGE 1 OF 1



APPENDIX A.14.

DATA DICTIONARY FILE/AREA DEFINITION

The data base is made up of areas, each containing a portion of the stored data. Each new application is assigned at least one area where its records are stored.

It is possible that applications will define additional special-purpose files for use during processing. Commonly used special files include card data input and printed outputs. When input data from cards is specified, a standard file called "DATA-IN" is used. This file has been previously defined as card-form input. It is not necessary to prepare IDD entries for card input of each new application.

Another standard file has been defined for printer output. This file is called "PRINT-OUT". It is used for all normal single-part 11x14 form printing. Other standard print files have been defined for special purpose forms usage. Refer to Appendix E.9 for the proper file name. If a special print file is required which is not defined, discuss the file requirement with the DBA staff prior to preparing new file definitions.

Figure A.14 illustrates the data dictionary entries for file definition:

1. ADD FILE NAME IS. The ADP name of the file is entered here. The name should be as descriptive as possible

within the 16-character limit indicated.

2. PREPARED BY. The initials of the preparer are entered.

3. SAME AS FILE. This clause is used only when the file is identical to an already existing file defined to the data dictionary. This condition will normally not occur with the definition of new applications since existing files will be used instead of designing or defining a new one.

4. FILE DESCRIPTION IS. The full name of the file is entered in the space provided. An abbreviated name is entered and the full name entered under comments if the full name is too long for the allotted space.]

5. EXTERNAL NAME IS. This name is assigned by the DBA staff. It is the OS DDNAME for the file.

6. LABELS ARE. This clause defines whether internal identifier labels are used by the file. "STANDARD" is entered in the space provided if the file entry describes a data base area or a permanently held file. The word "OMITTED" is entered if the file is a temporary file or used for card input/output or is a print-type file.]

7. RELATED FILE IS. This clause is used only when the file being described is similar to another file or is used in conjunction with another file during processing. When an application using the data base utilizes two or more data

base areas the interrelationship is defined through this clause. VERSION IS identifies the version of the related file.

8. ENTRY-SECURITY IS. This clause defines the security classification of the file description. See Appendix E.1 for a list of applicable classifications.

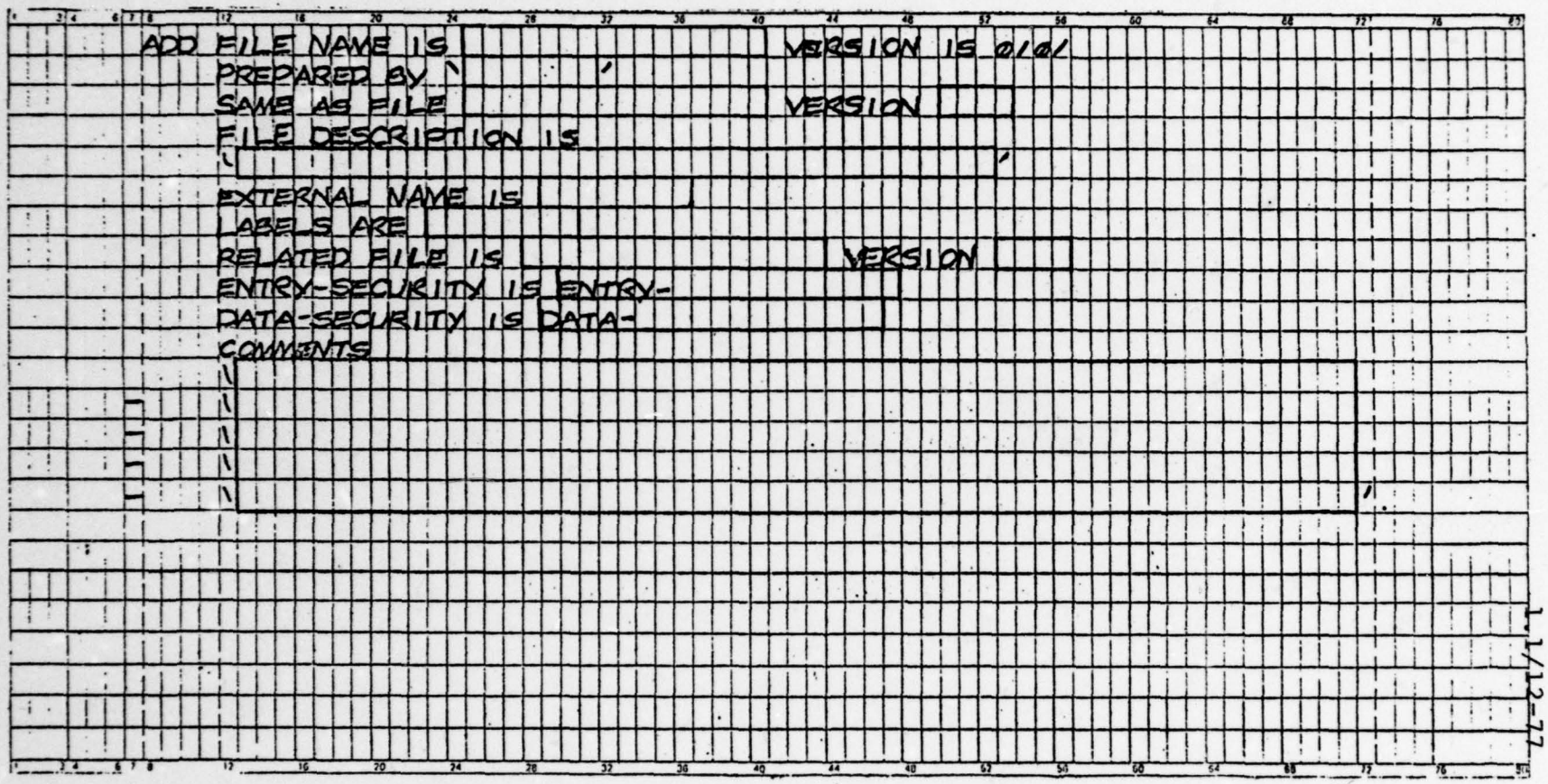
9. DATA-SECURITY IS. This clause defines the security classification of the contents of the file. See Appendix E.1 for a list of applicable classifications.

10. COMMENTS. The comments entry is provided to permit a detailed explanation of the purpose and uses intended for the file. As many comment lines as desired may be coded. The last comment line must terminate with a single quotation mark followed by a period ('.).

ADD IDD FILE/AREA

DATA BASE DEVELOPMENT & DESIGN GUIDE

PAGE 1 OF 1



1.1/12-77

APPENDIX A.15.

DATA BASE DEVICE-MEDIA CONTROL (DMCL) DEFINITION

The data base definiton (schema) defines the logical relationship of data elements, data groups, and records. Each record is logically assigned to an area of the data base. Normally this assignment corresponds to an application which supports a portion of the total data base.

This logical placing of data within areas must also correspond with physical space located on a storage device attached to the computer. The data base system is very flexible in the way it permits the logical areas of the data base to be physically located within the computer storage.

The DMCL module is the map which the data base system uses to identify which data base areas belong to which storage areas within the computer. The ability of the data base user to access data within the data base is controlled by the DMCL module. Access is not permitted if the module does not identify a storage area with a logical data base area.

The statements of the DMCL language module are illustrated in Figure A.15.1. Each statement is a complete sentence and must be terminated by a period as shown.

1. DEVICE-MEDIA NAME IS. This statement identifies the

ADP name of the DMCL module. It is assigned by the DBA staff.

2. OF SCHEMA NAME. This clause identifies the schema (data base description) which is primarily associated with the DMCL module. This is normally the subsystem name assigned to the application. VERSION IS will normally be "0101" but may be different if the schema has been updated during design.

3. AUTHOR. The name of the DBA staff member preparing the module entries is entered.

4. DATE. The date the DMCL module was initially prepared is entered.

5. REMARKS. The subsystem supporting the application is entered in the appropriate box.

6. BUFFER NAME IS BUFFER-. The subsystem name is entered in the space provided.

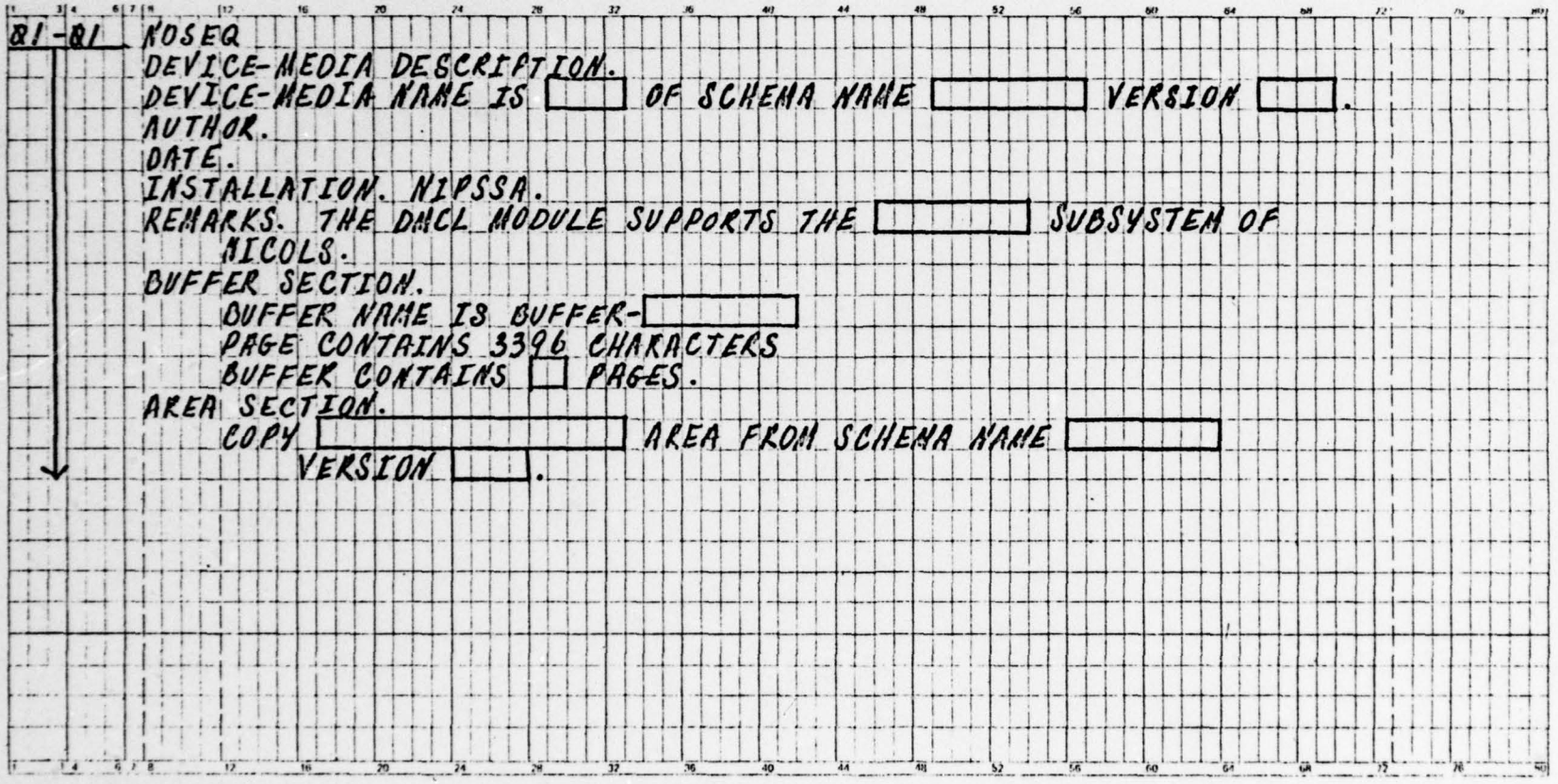
7. BUFFER CONTAINS nn PAGES. The normal number of pages for a buffer is eight. This number may be increased for special purpose DMCL modules. A practical maximum of 12 buffers is recommended. Beyond that number efficient utilization of the computer central processor is impacted.

8. COPY xxxxxx AREA. The name of the logical data base area which will be accessed through the DMCL module is entered. FROM SCHEMA NAME. This clause indicates the

name of the schema where the data base area is described. VERSION IS further defines which version of the schema is to be used to locate the area definition. The FROM SCHEMA NAME xxxx VERSION nnnn clause may be omitted and a period placed immediately following the word AREA if the schema is the same as that defined in the DEVICE-MEDIA NAME statement. As many area statements as desired may be coded. Each is a complete sentence.

Every DMCL module is stored on the Program Support Library (PSL).

DMCL MODULE SOURCE STATEMENTS
DATA BASE DEVELOPMENT & DESIGN GUIDE
PAGE 1 OF 1



APPENDIX A.16.

DATA BASE SUBSCHEMA DEFINITION.

It is seldom necessary for the individual user to see the entire data base structure. Actually it is often desirable that each user be limited to certain records, sets, and areas of the data base to assure proper security. The subschema is the tool which permits the DBA to assign and limit the user's accessability to the data base. The subschema may be looked at as a window through which the user may view a portion of the data base.

Each subschema must be keyed to a particular DMCL module. In this way a two-level security function is implemented. The first level (DMCL) defines which physical data base files may be accessed for processing. The second level (subschema) defines how those areas may be accessed.

Each statement in the subschema definition is a complete sentence and must be terminated by a period. No periods are placed at the end of clauses within the statements. The statements are:

1. SUBSCHEMA NAME IS. The ADP name of the subschema is assigned by the DBA staff. OF SCHEMA NAME. This clause identifies the schema which the subschema supports. This is normally the name of the subsystem application being developed. VERSION. The proper version of the schema is entered.

2. DEVICE-MEDIA NAME IS. The DMCL module which will interface with the subschema is entered. OF SCHEMA NAME. The schema which controls the DMCL module is entered. This is normally the same as the schema named in the subschema name statement. If it is the OF SCHEMA NAME and VERSION clauses may be omitted and a period placed immediately following the DMCL name.

3. AUTHOR. Enter the name of the DBA staff member preparing the subschema entries.

4. DATE. Enter the date when the subschema was originally prepared.

5. REMARKS. This statement provides for free-form text explanation of the purpose of the subschema. The explanation may use as many lines as required. Complete sentences are encouraged to improve readability.

6. COPY xxxx AREA. This clause identifies the name of an area to be included in the subschema view of the data base. As many area statements as required may be coded.

7. PRIVACY LOCK FOR. This clause defines specific limitations, other than the three shown, which the DBA may wish to impose on the use of the area. The first entry space may be left blank or contain the word "PROTECTED". The second entry space may contain either "UPDATE" or

"RETRIEVAL". The privacy locks remaining are "RETRIEVAL", "PROTECTED UPDATE", and "PROTECTED RETRIEVAL".

8. COPY xxxxx RECORD. This statement instructs the subschema compiler to extract the entire named record and make it available to the user. If no privacy locks are required a period must be placed to the right of RECORD. PRIVACY LOCK FOR clause indicates that the user will not be permitted to perform certain data base actions against the specified record. See Appendix E.10 which defines the available privacy locks and their purposes.

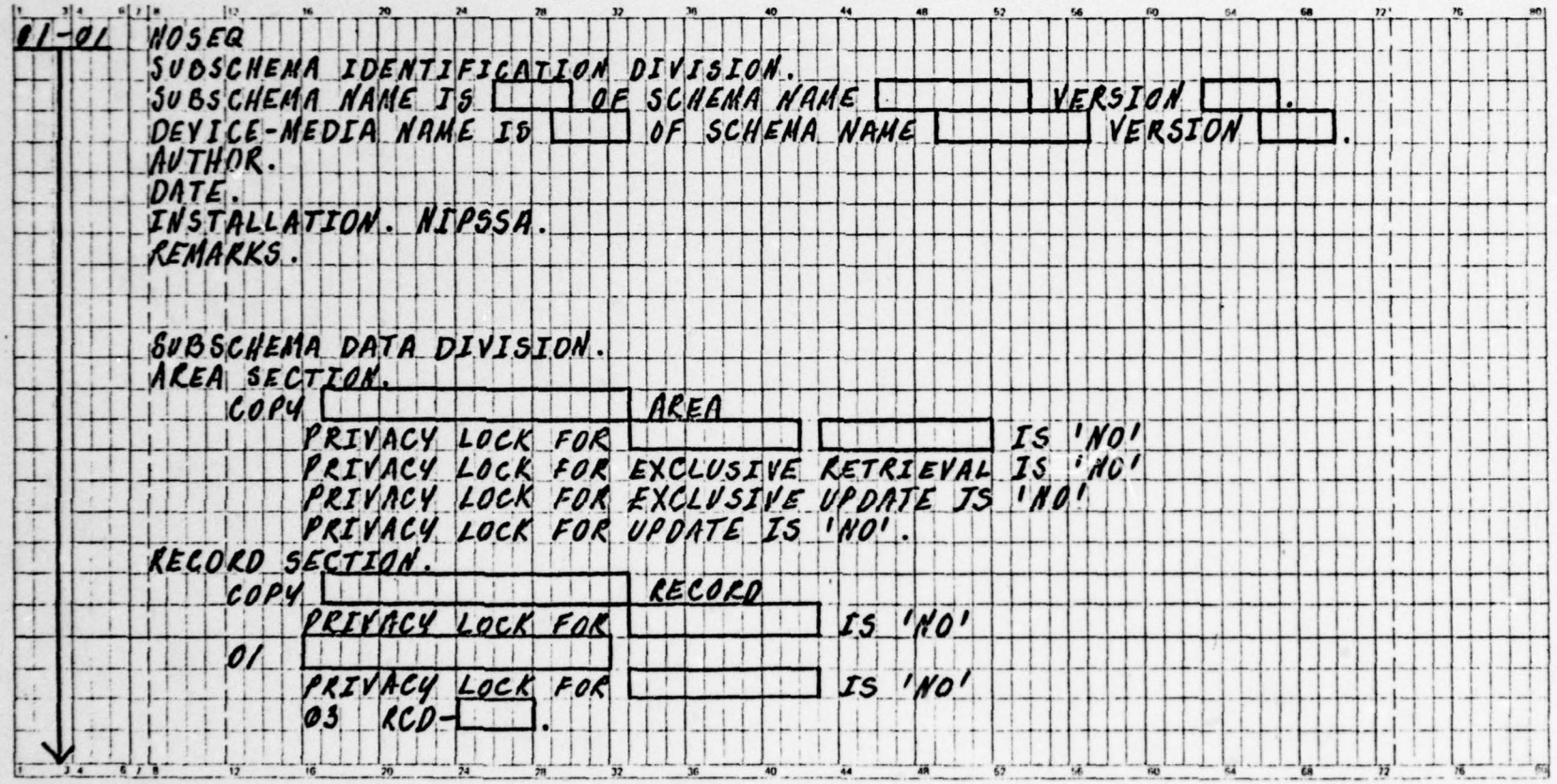
9. 01 xxxxx. This version of the record statement is used when only a portion of a data base record is to be provided to the user. The name of the record follows the "01" indicator. PRIVACY LOCK FOR clauses are written to restrict the user's activity as defined above. See Appendix E.10 for definitions of available privacy locks. This form of the record definition is used for all protected record types in the data base. A period must follow the privacy lock clause.

10. 03 RCD. This clause is used for protected record types. The clause is completed by entering the numeric identifier of the record type being used in the subschema. When other segmentation (data element selection) is being performed, all selected elements or groups of elements must be at the same numeric level (e.g. 03, 05).

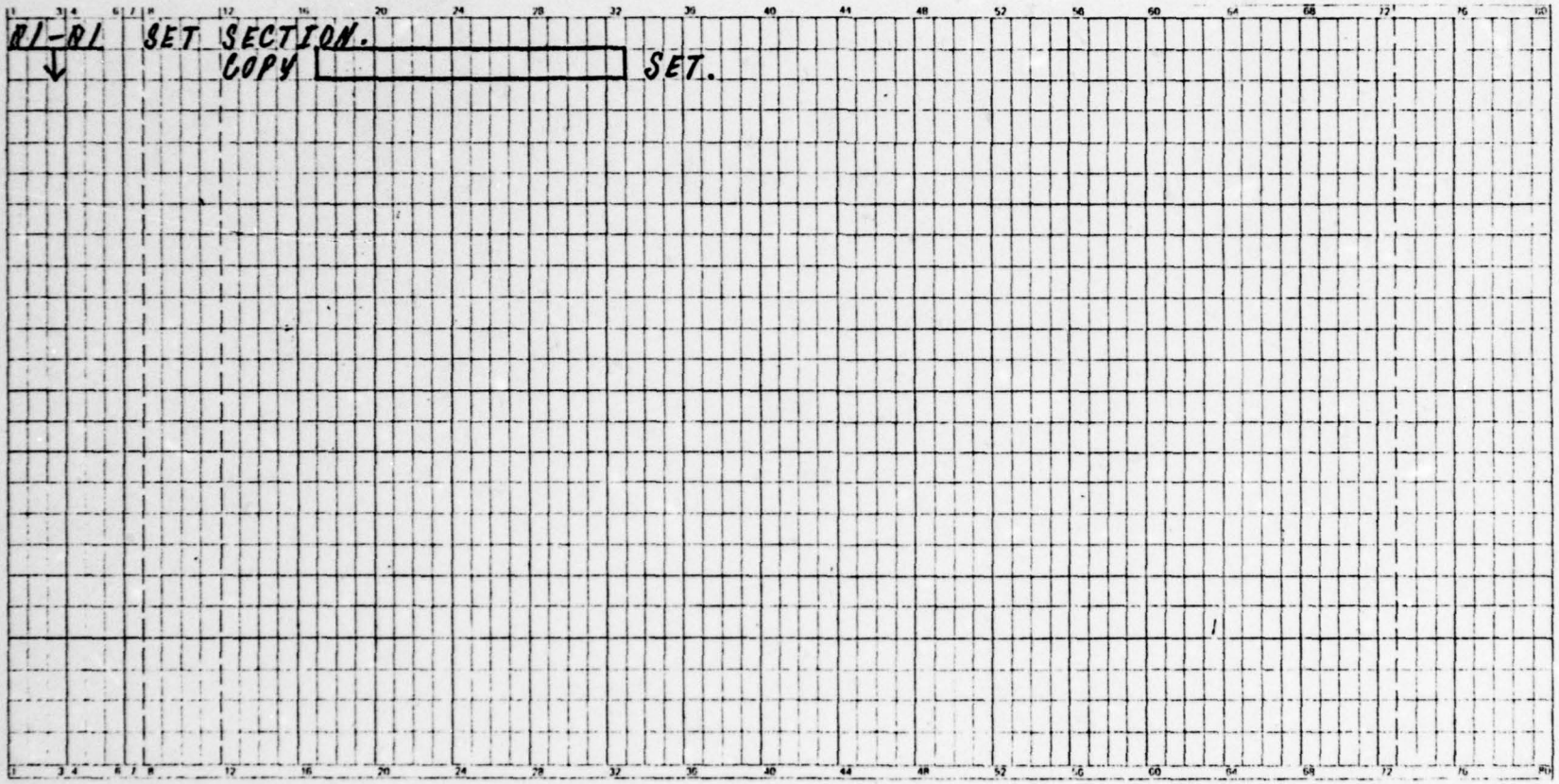
NOTE: All record types copied and defined by the subschema must reside in the areas copied in the previous section.

11. COPY xxxx SET. This clause defines the sets within the schema which will be required to support the user's application. Normally no privacy locks are placed on sets. When a set is specified in this section it is necessary that the owner record type and at least one member record type be specified in the record section above.

SUBSCHEMA SOURCE STATEMENTS
DATA BASE DEVELOPMENT & DESIGN GUIDE
PAGE 1 OF 2



SUBSCHEMA SOURCE STATEMENTS
DATA BASE DEVELOPMENT & DESIGN GUIDE
PAGE 2 OF 2



APPENDIX A.17.

IDD TIE POINT DEFINITION

As new applications are developed associations will be required with the existing data base. This association is established by the definition of what is termed "tie-point records". These records identify relationships with data base information outside of the area of application. The application needs to associate with existing data base information but the level of redundancy must be minimized. Figure A.17.1 illustrates the condition where applications A and B both require information about organizations. This information is stored in a general organization record located in the general reference area of the data base. It is not desirable for each application to duplicate the organization information. It is also not desirable for each application to directly share the information. Such sharing forces each application to be reliant on the other for data base integrity and stability. New applications under development can impact on operational applications and restructuring of the organization record to add new pointers occurs continuously.

The tie-point record approach eliminates inter-application dependency. It permits multiple applications to be developed concurrently with the operation of production applications without fear of interference. Of particular

importance from a security standpoint is the removal of physical pointers connecting diverse applications of different classifications. This separation permits independent operation and protection of data base areas of different classifications.

A certain amount of data redundancy is planned in the use of tie-point records. The value of the reference (CALC) key of the general reference record is duplicated in every associated tie-point record. This permits direct access to the general record during retrieval conditions.

Figure A.17.2 illustrates a second tie-point record use. In this case, an association is established with another application instead of a general reference record. The tie-point record points to a particular record type within the associated application. It is important that the associated record be stored CALC, as a member of a sorted set, or with a secondary index on the reference key. Easy access to the associated record is not possible unless one of these conditions is met. The most desirable is the CALC option.

The design of the tie-point record begins with a determination of the data elements which will be contained in the record and the corresponding elements already resident in the data base. Use instructions in Appendix A.6 to define

the tie-point elements. Use the SAME AS clause to identify the element as identical, except for name, to the corresponding data base element.

Once the tie-point elements have been defined the IDD entries are prepared to define the tie-point record and establish synonym relationships with the corresponding data elements. Two pages of entries are provided. The first page establishes the tie-point record:

1. ADD RECORD NAME IS. This clause defines the name of the tie-point record. It is desirable that the record name illustrate the relationship between the associated data base record and the new application, e.g. PERS-ORGAN, for an organization tie-point record for the personnel subsystem.

2. PREPARED BY. The initials of the preparer are entered.

3. RECORD DESCRIPTION IS. This is a 40-character expansion of the record name.

4. ENTRY-SECURITY IS ENTRY-. This field identifies the security classification of the record description entry. See Appendix E.1 for a list of applicable classifications.

5. DATA-SECURITY IS DATA-. This field identifies the security classification of the record data within the data base. See Appendix E.1 for a list of applicable classifications.

6. RECORD NAME SYNONYM IS. This clause identifies the associated data base record the tie-point record is pointing to.

7. COMMENTS. The comments clause is shown with three lines. As many additional lines may be coded as desired to adequately define the record. The last line of the comments must be terminated with a single quote mark followed by a period ('.).

8. RECORD ELEMENT IS. This sentence defines the data elements which are to be part of the tie-point record in the data base. Each statement must be terminated by a period immediately following the data name without intervening spaces. The elements are to be listed in the order in which they will appear in the record from beginning to end.

The second page provides synonym association between individual data elements of the tie-point record. Four entries are provided. Additional ones may be coded as required. Unused entries should be crossed off to prevent extra punching.

NOTE: Normally a tie-point record only has one associated data element with the corresponding data base record to reduce redundancy. When more than one element is desired or required, logical updating problems occur because the alteration of one element without a corresponding alteration of its associated element creates an integrity question as to which one is correct. SPECIAL DBA APPROVAL IS REQUIRED TO CORRESPOND MORE

THAN THE REFERENCE KEY ELEMENT IN TIE-POINT RECORDS.

1. MODIFY ELEMENT NAME IS. This clause identifies the name of the tie-point record element which is to be associated with a corresponding data base element.

2. INCLUDE ELEMENT NAME SYNONYM IS. This clause identifies the corresponding data base data element.

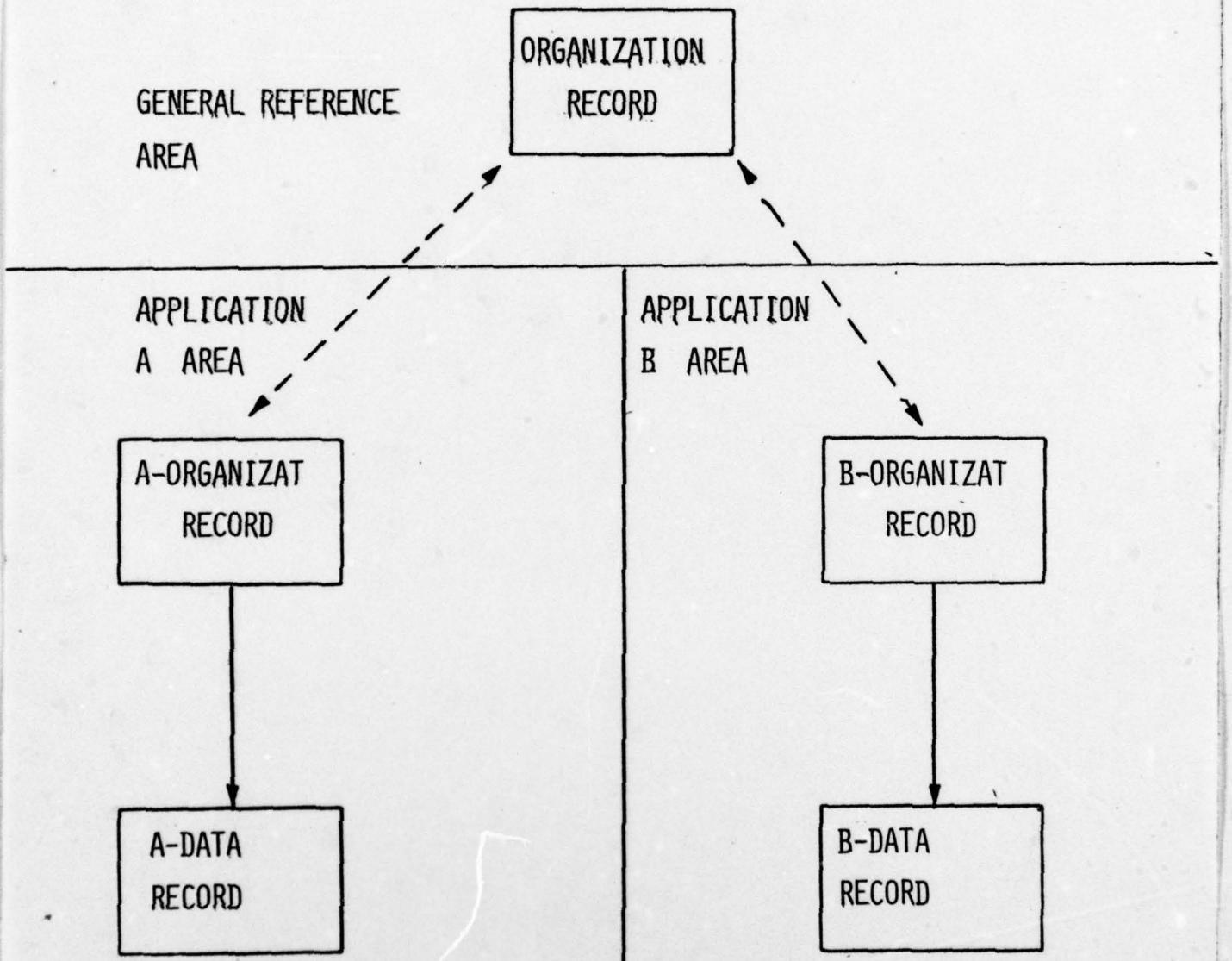

ORGANIZATION RECORD
GENERAL REFERENCE AREA
APPLICATION A AREA
APPLICATION B AREA
A-ORGANIZAT RECORD
B-ORGANIZAT RECORD
A-DATA RECORD
B-DATA RECORD

FIGURE A.17.1

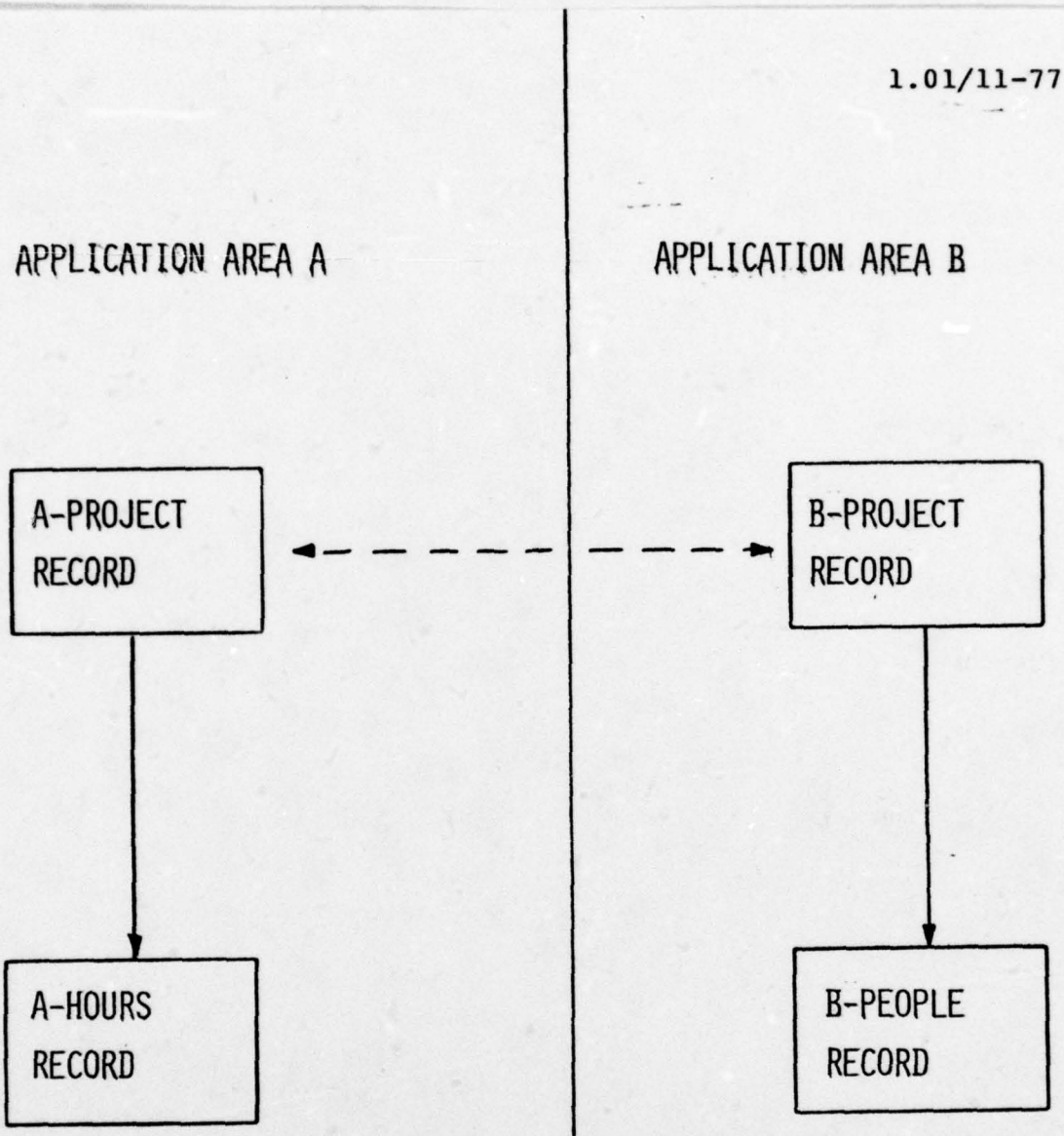
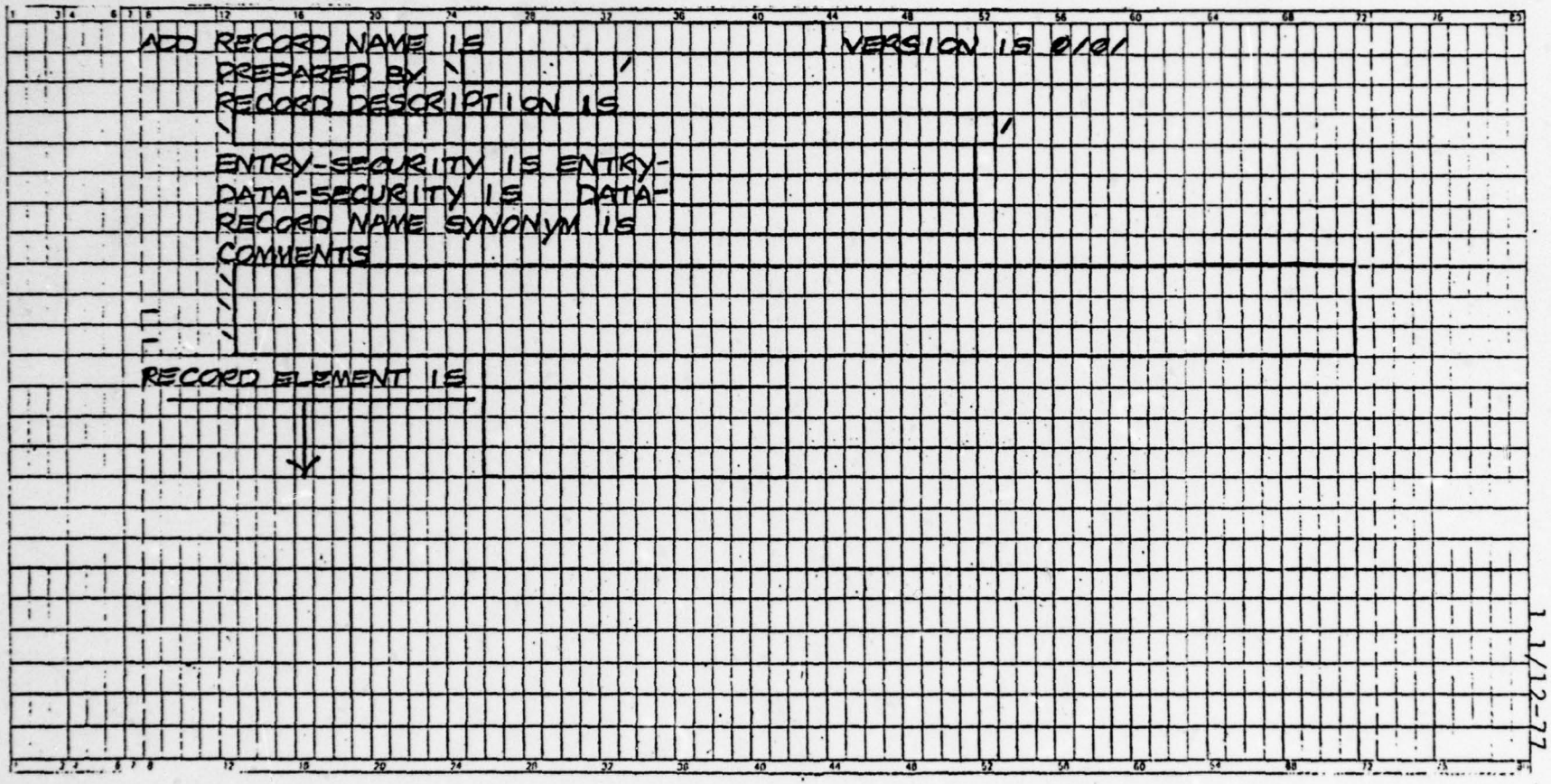
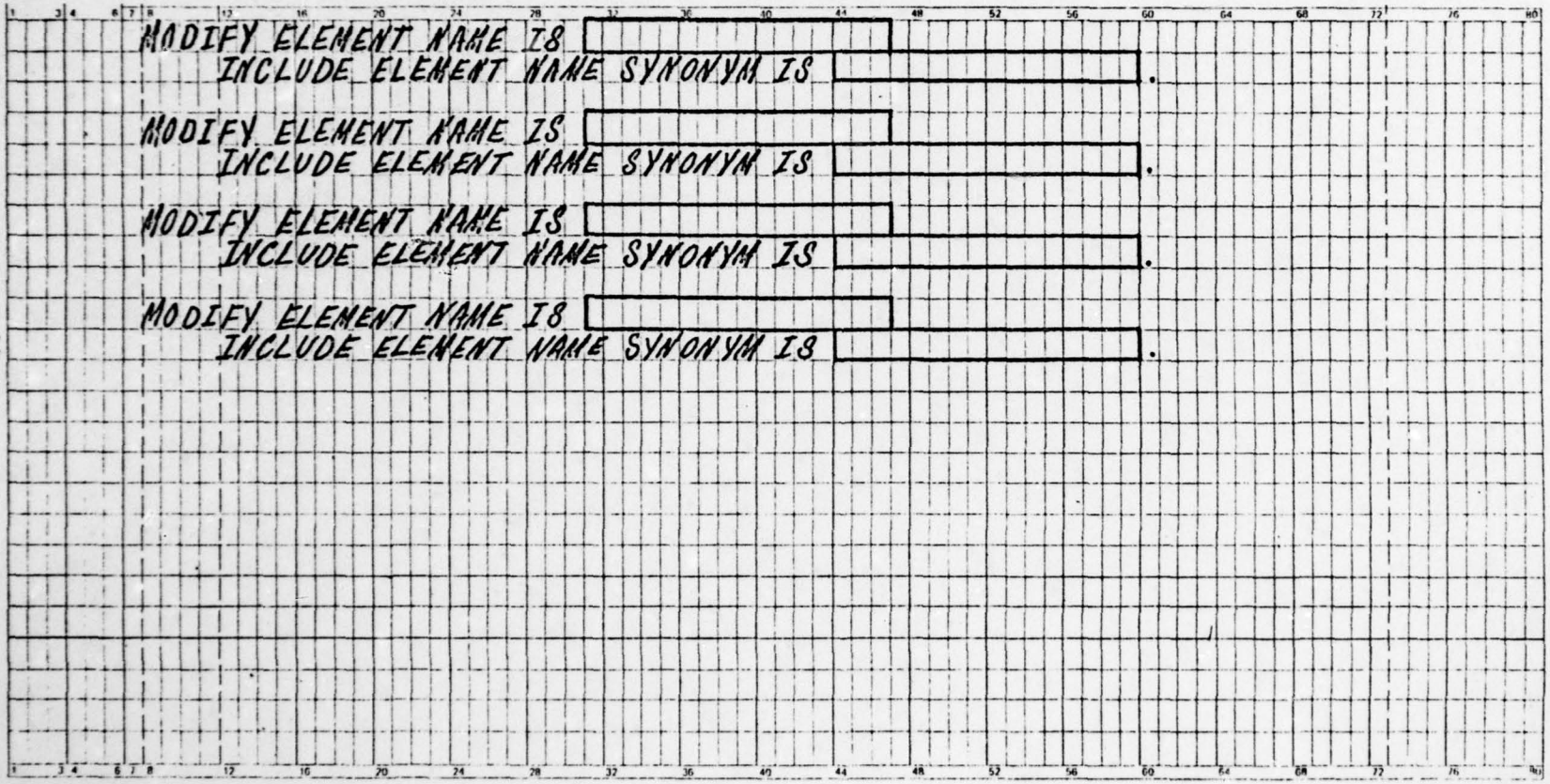

APPLICATION AREA A
APPLICATION AREA B
A-PROJECT RECORD
B-PROJECT RECORD
A-HOURS RECORD
B-PEOPLE RECORD

FIGURE A.17.2

TIE-POINT ESTABLISHMENT
DATA BASE DEVELOPMENT & DESIGN GUIDE



TIE-POINT ESTABLISHMENT
DATA BASE DEVELOPMENT & DESIGN GUIDE
PAGE 2 OF 2



APPENDIX A.18.

INPUT PROCESSING PROGRAM SPECIFICATION DEFINITION]

Data must be loaded to each record defined within the data base. This appendix describes the procedure for preparing specifications for input processing (IP) program modules which will be coded in the COBOL programming language.

Each IP module is designed and coded as an independent entity. The philosophy employed is that all input processing is transaction-driven, each transaction independent of those preceding and following. This approach assures that incoming data can be processed without regard to its environment.

Each IP module is independent of the input medium. The module expects its data in a particular format and processes the data according to the specified statements.

IP modules are designed and programmed using structured programming techniques which groups similar functions to achieve more accurate processing. A facility of the COBOL language compiler permits definition of common program source statements into what is known as a source library "book". These books may be modified when they are requested by the program, tailoring the statements to the specific data element] being serviced. This approach assures consistent common language logic and eliminates large volumes of statement

coding and punching. Appendix E.11 defines the source library books in detail.]

IP specifications are written as COBOL language comment statements. This permits placing the specification information directly into the program, eliminating the need for a separate program specification document. Maintenance of the specification and program consistency is also improved.]

Five pages of formatted coding sheets are provided for specification definition. Statements are:

1. IP IDENTIFIER. This field contains the ADP name assigned to the IP module by the DBA staff. See Appendix B.8.

2. PROCESSING OPTIONS. Three single-position fields are provided. The options permitted are:

a. S - Store a new record into the data base using information in the input data.

b. M - Modify an existing record in the data base using information in the input data. Blank fields are not modified.

c. D - Delete an existing record from the data base using information in the input data. The only data required in the input is that which defines the record to be deleted.

d. I - Insert an existing record into a set that

the record participates in as a member.]

e. R - Remove an existing record from a set in which it participates as an optional member.

3. DATA BASE RECORD. This entry contains the ADP name of the data base record which the IP will service. The record name must correspond to a record defined for use in the data base.]

4. DB RECORD IDENT. This field contains a numeric value corresponding to the data base record definition. The number is assigned by the DBA staff and must correspond to conventions defined in Appendix B.9.

5. SUBSCHEMA. This field contains the DBA staff assigned identifier of the subschema which will service the IP module.]

6. IP DESIGNED BY. The name of the person who prepares the IP specification is entered.

7. IP PROGRAMMED BY. The name of the person who performs the programming of the IP is entered when that person has been identified.

8. SET MEMBERSHIP. These statements identify the data base sets which are used by the IP during data base accessing.

a. O/M. This field defines if the owner or member of the set is used by the program.

b. SET NAME. This field contains the name of the

set.

c. NPO. This field defines the pointer arrangement used by the set. "N" will always be present. "P" is used when the set contains prior pointers, a normal occurrence. "O" is shown when owner pointers are also present.

d. MAOM. This field defines the contingency of relationship between owner and member record. The valid entries are either "M" or "O" and "M" or "A". These indicate mandatory, optional, manual, or automatic, respectively.

e. ORD. The storing order of members of the set is indicated. Enter the appropriate set order. If the set is sorted, enter either "ASC" or "DESC".

f. SORT FIELD/DUP/ASSOC RCD. This field is used to provide information on sorted fields and associated set records. If the set is sorted the field name of the sort key is entered, followed by "FIRST" or "LAST" if duplicates are allowed. If no duplicates are allowed, enter "NO". Enter the name of the associated record in the set (usually the owner record).

9. FUNCTIONS TO BE ACHIEVED BY IP. This page is used to describe in detail the individual functions which the IP will be expected to perform other than those of data element validation and the specified data base functions basic to the IP.

10. SPECIAL PROCECESSING REQUIREMENTS. This page is used to define special processing which must be performed to achieve the functions assigned to the IP. This includes calling subroutines, transformations, computations, and table lookups.

11. CURRENCY REQUIREMENTS. This page defines the currencies which must be established to permit IP processing to continue. The currency record type is indicated followed by the type of record, the field upon which selection is based and the type of IDMS command required to access the record. Identify the set if the record is accessed through a set.

12. IP DATA AREA NAME. The COBOL working storage data area which will provide the input structure for the IP is described. The data area name must begin with the three-character IP identifier, followed by a hyphen. The remaining portion of the data name should describe the IP function.

13. DATA ELEMENT VALIDATION. This group of statements defines each data element within the IP and the validation functions required. The five clauses are ordered to expedite the coding of source library books:

a. BOOK. This field contains the name of the source library book used for performing the primary function

of the IP. If the IP permits store, modify, and delete, the book field contains the source library book name of the store book. Modify and delete book names are identified in the remarks area, if different from the store book.]

b. F-NO. This clause identifies the field number of the data element within the input data definition. The fields are numbered from left to right with the format identifier and option fields as one and two, respectively.

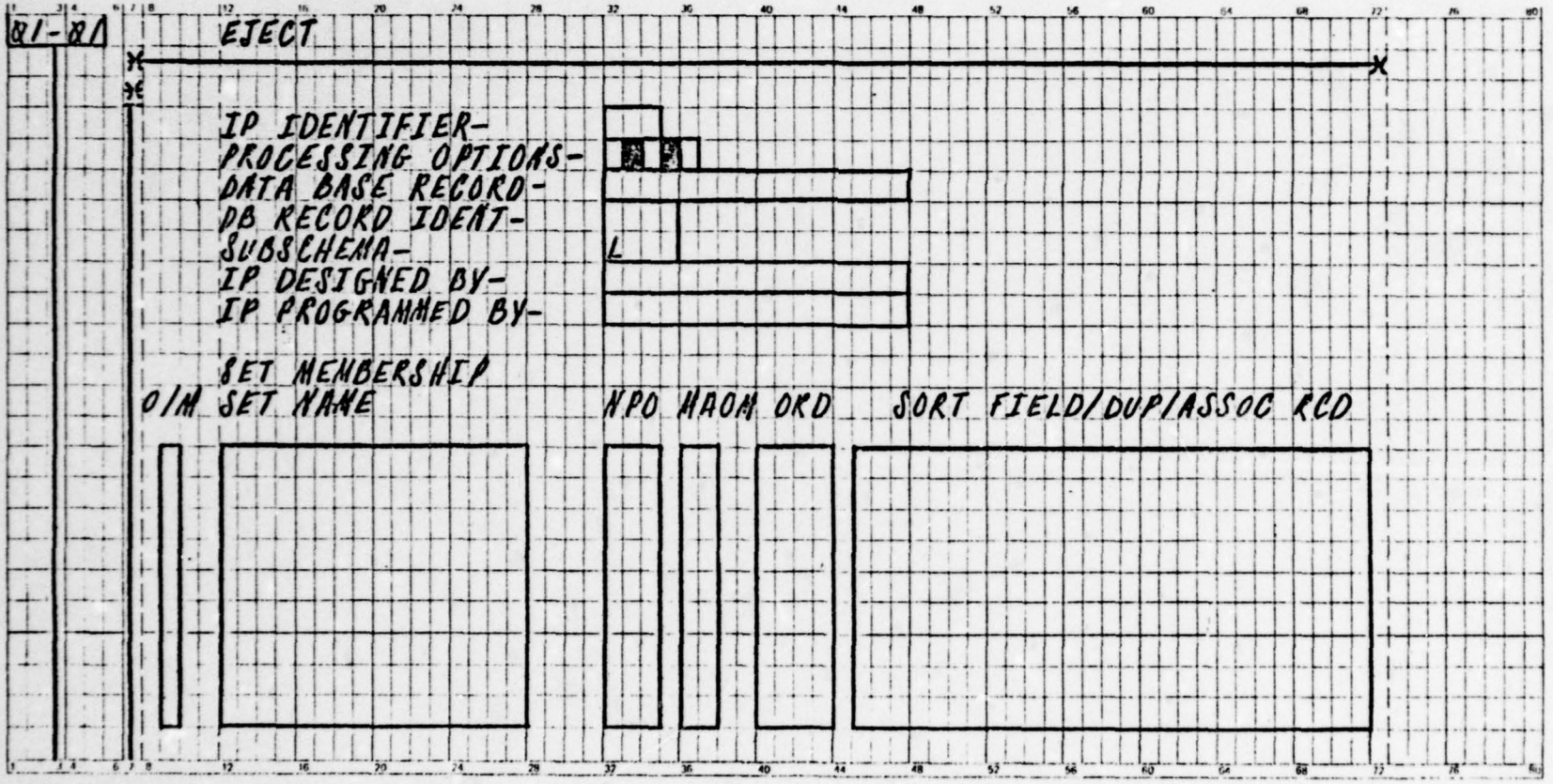
c. F-FLD. This clause identifies the name of the data field in the working storage area assigned for IP input.

d. R-FLD. This clause identifies the name of the corresponding field in the data base record which will receive the processed data.

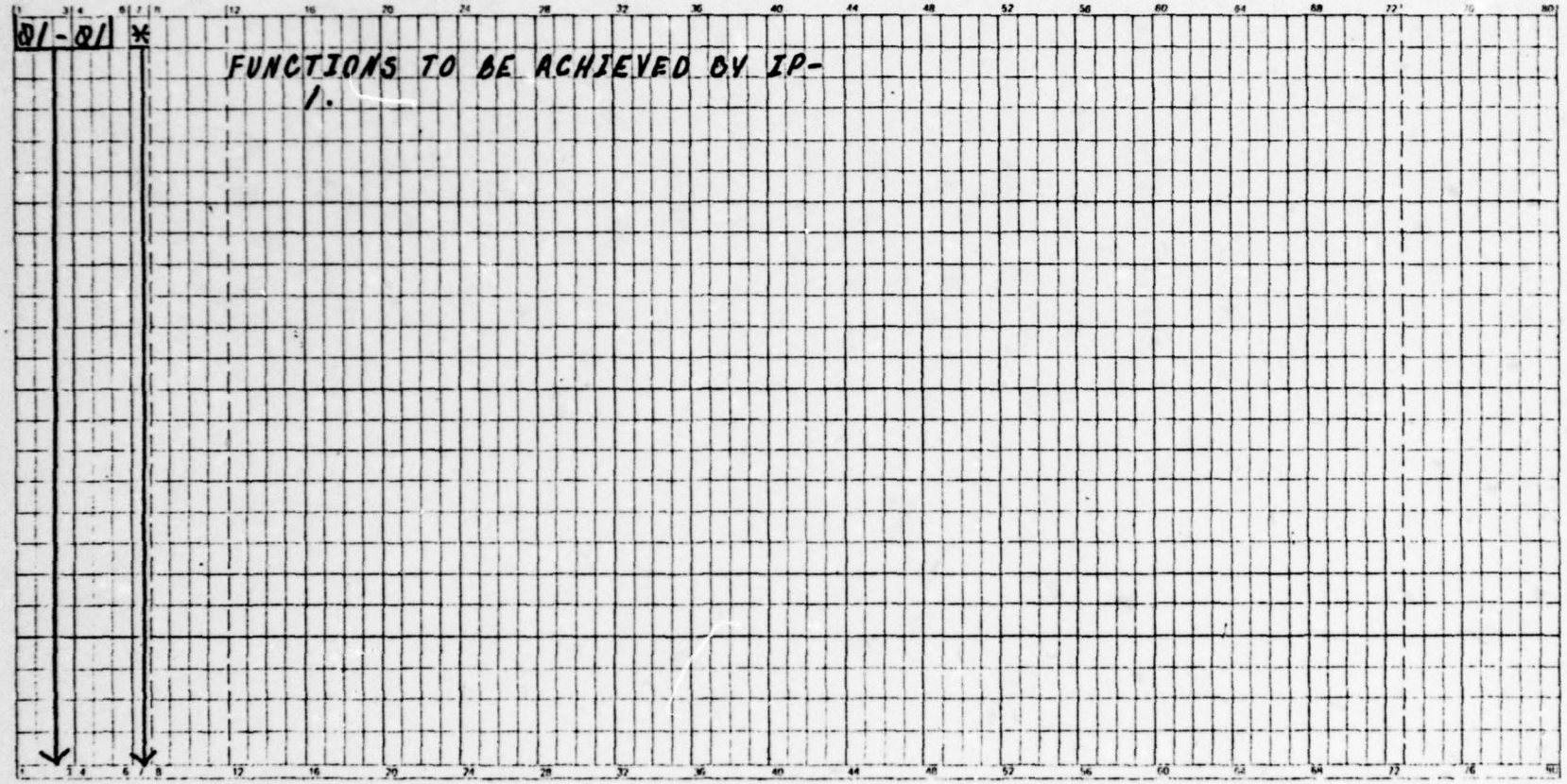
e. REMARKS. This clause contains such other information as necessary to code the data element validation. More than one line may be used if desired to completely explain the validation process. If a subroutine is called it is identified here. Range values are entered in this field.]

NOTE: Appendix E.11 identifies the source library books used to perform data element validation.

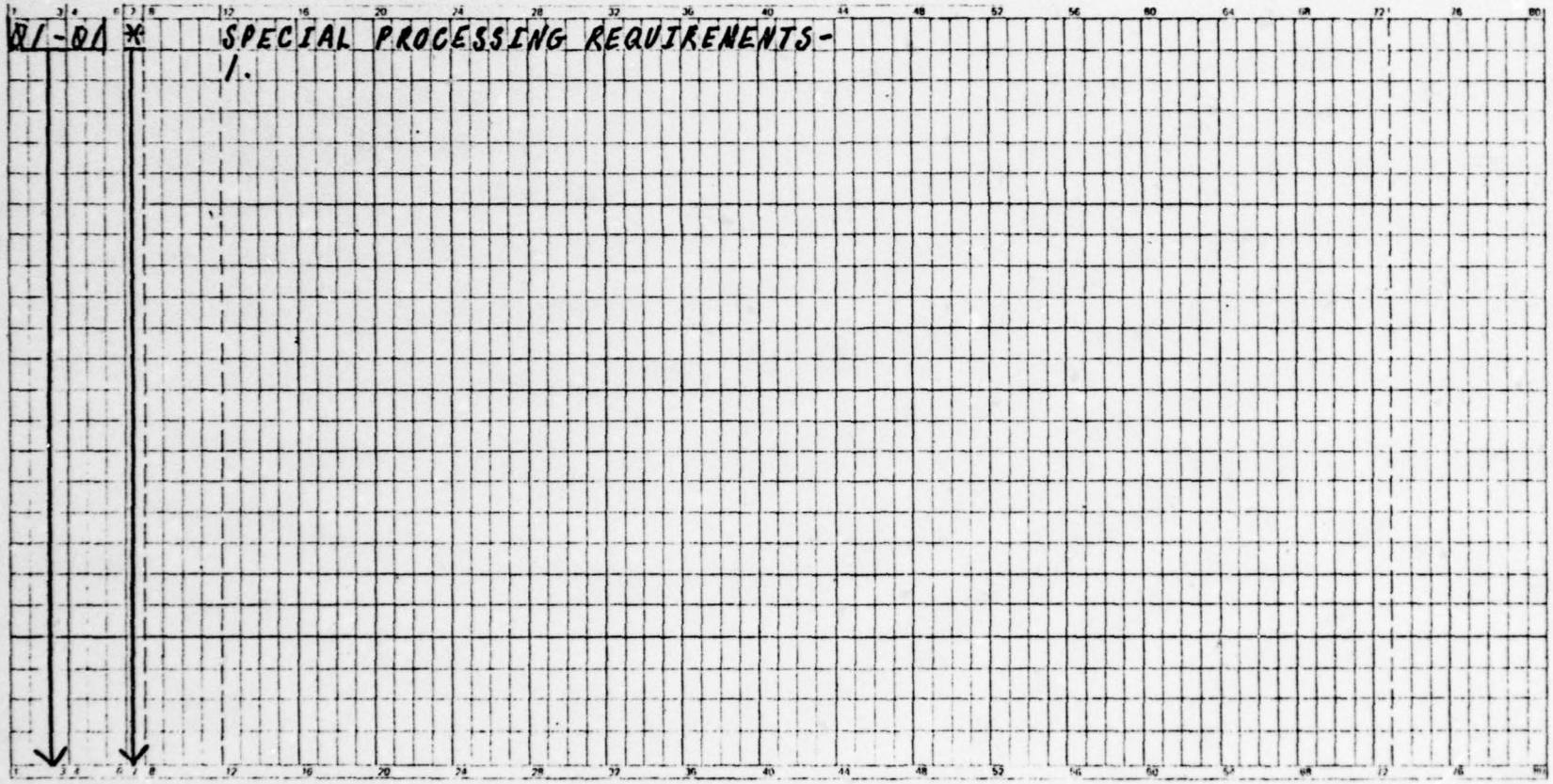
IP DEFINITION
PAGE 1



IP DESCRIPTION
PAGE-2
90-10-01



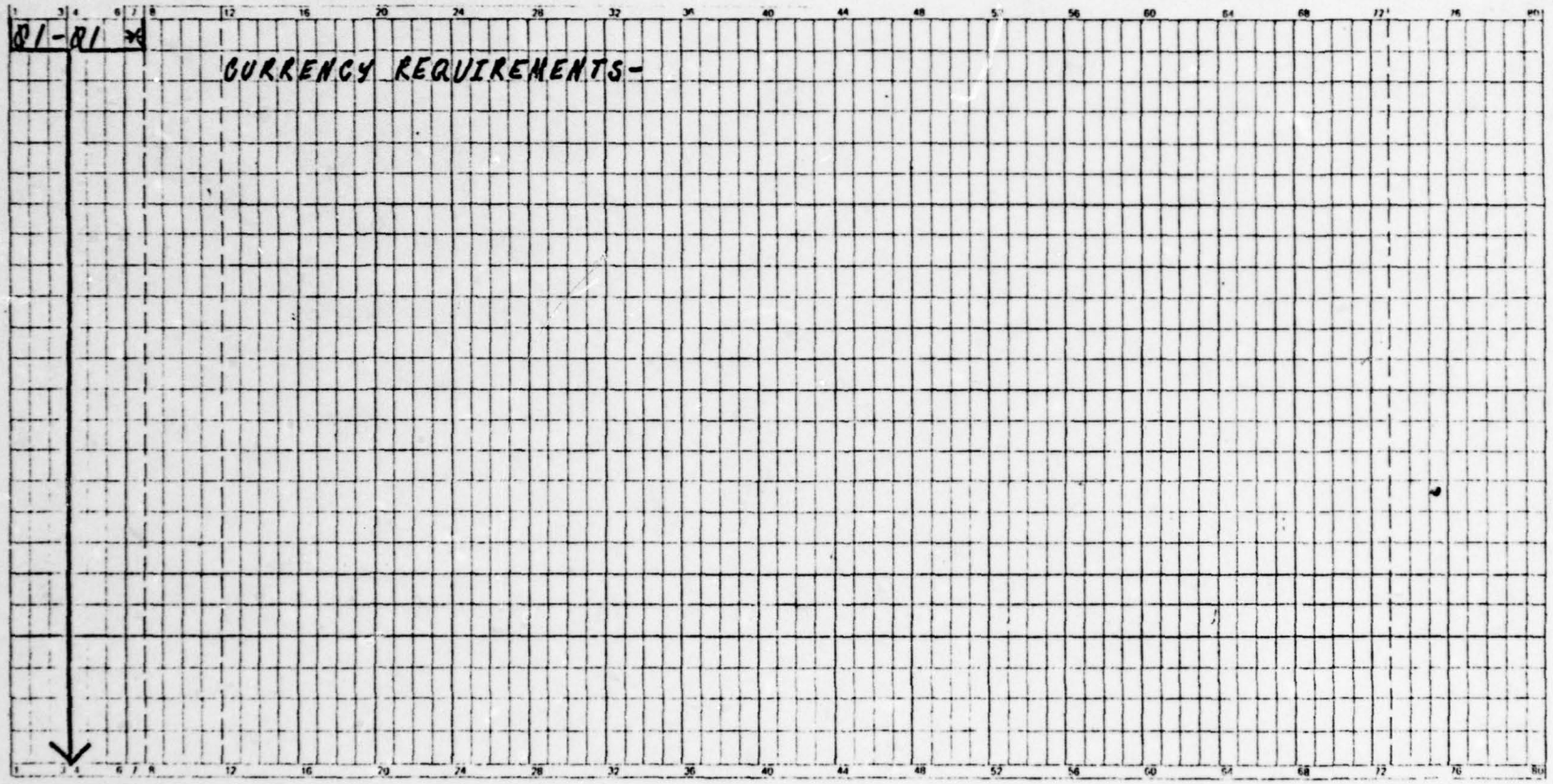
IP DESCRIPTION
PAGE-3
90-10-01



A.100

1.01/11-77

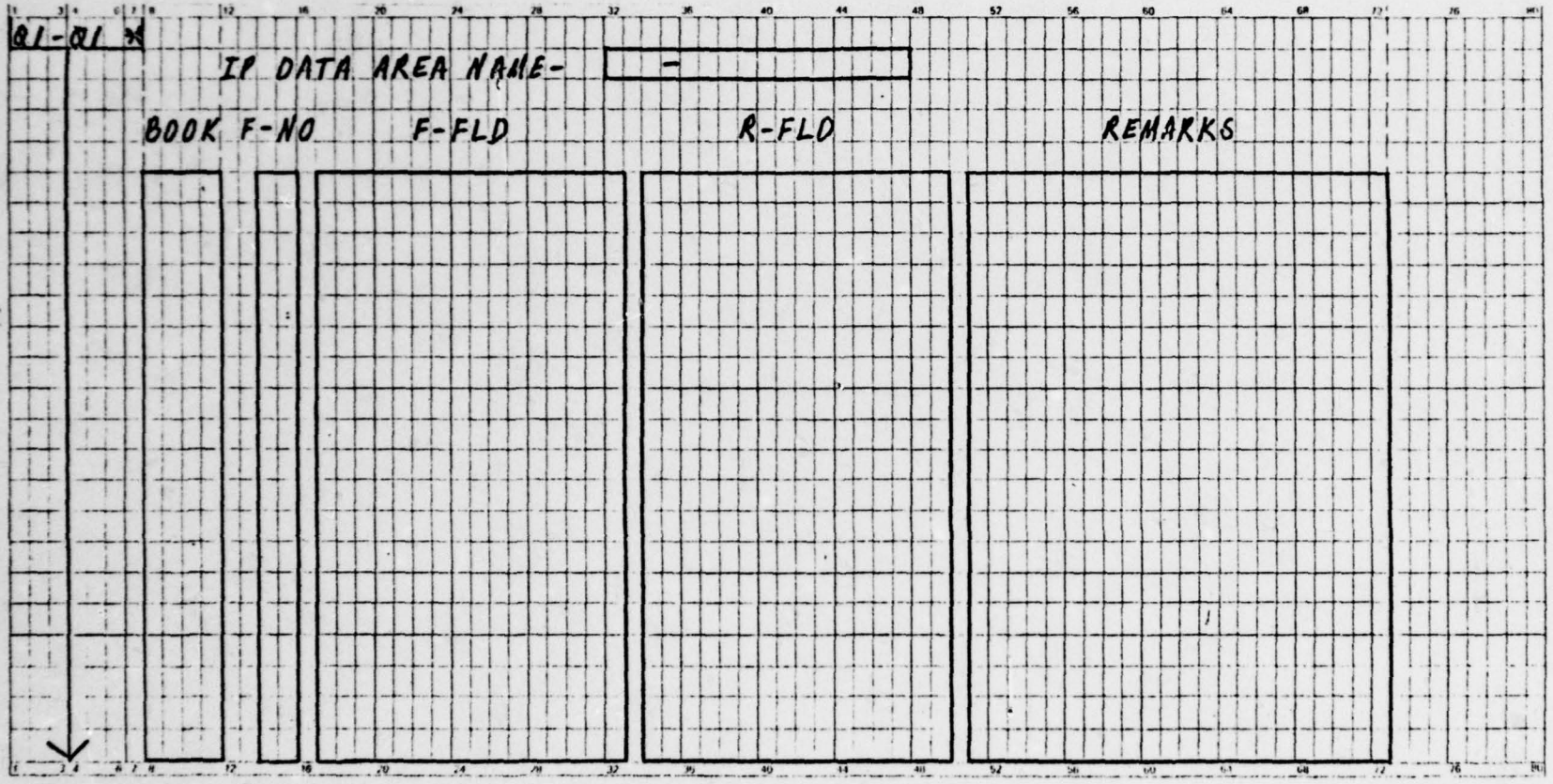
IP DESCRIPTION
PAGE-4
90-10-01



A.101

1.01/11-77

IP DESCRIPTION
PAGE-5
90-10-01



A.102

1.01/11-77

Once the IP specification data has been prepared IDD entries must be prepared to define the IP structure to the data dictionary. This is done in two stages: (1) the definition of individual data elements and data element groups; and (2) the definition of the IP input data record.

Three coding forms are provided to assist data element definition to the data dictionary. Each will be described in detail.

The first page provides entries for constant data elements which occur at the beginning of every IP data definition.

1. ADD ELEMENT NAME IS xxx-FORMAT. The IP identifier is inserted in place of xxx. This field contains the IP identifier which is used by the processing system to identify the data definition.

2. VALUE IS. The IP identifier is inserted.

3. ADD ELEMENT NAME IS xxx-OPTION. The IP identifier is inserted in place of xxx. This field contains the processing option to be used by the system to process the IP data.

4. VALUE IS. Three occurences of this clause are provided. If less than three are required, place a period following the last one used and cross out the remaining to prevent punching. The values in this field must correspond

to the options defined earlier for the IP.

5. ADD ELEMENT NAME IS xxx-RCD-CTL. This is a group data element which contains the previously defined individual elements. xxx is replaced by the IP identifier.

6. SUBORDINATE ELEMENTS ARE xxx-FORMAT xxx-OPTION. xxx is replaced by the IP identifier.

The second page contains eight occurrences of the data element definition used by IDD. It identifies the name of the IP data element and relates it to the corresponding data base data element. In some cases the data element does not relate to a data base data element and is used solely for IP processing control. A full data element description should be prepared as described in Appendix A.6 in this case.

1. ADD ELEMENT NAME IS. The IP identifier is entered as a prefix, followed by a hyphen. The remaining 12 characters is a descriptive data name.

2. PREPARED BY. The initials of the preparer are entered.

3. SAME AS ELEMENT. The name of the corresponding data base data element is entered.

The third page contains two occurrences of a group element definition. This definition is used to associate groups of data elements together that have been previously

defined.

1. ADD ELEMENT NAME IS. The IP identifier is entered as a prefix, followed by a hyphen. The remaining 12 characters is a descriptive data name.

2. PREPARED BY. The initials of the preparer are entered.

3. SUBORDINATE ELEMENTS ARE. The names of IP data elements are entered, line by line, in the order in which they appear in the IP data definition.

The IP record definition page is completed following data element definition.

1. ADD RECORD NAME IS. This entry defines the name of the IP data definition. The name is prefixed by the IP identifier and joined to a descriptive name by a hyphen.

2. PREPARED BY. The initials of the preparer are entered.

3. RECORD DESCRIPTION IS. An expanded descriptive record name is entered. A single quote mark (') is entered at the end of the name.

4. ENTRY-SECURITY IS ENTRY-. This field defines the security classification of the IP record entry. See Appendix E.1 for a list of applicable classifications.

5. DATA-SECURITY IS DATA-. This field defines the

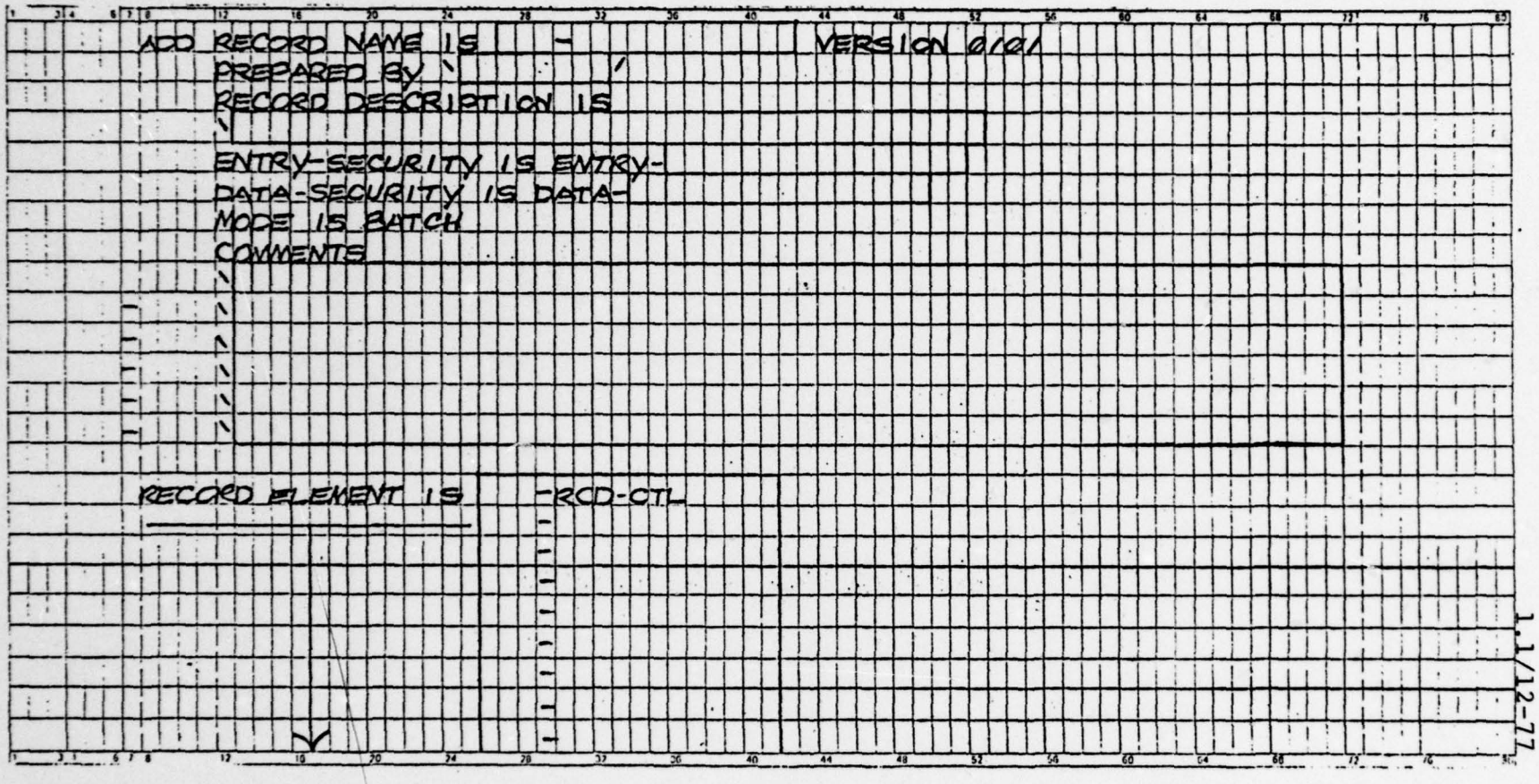
security classification of the IP data described by the definition. See Appendix E.1 for a list of applicable classifications.

6. COMMENTS. Six comment lines are provided. As many additional comment lines may be added as are required to describe the IP data definition in non-ADP terms which may be readily understood by the user. A single quote mark followed by a period ('.) is placed at the end of the last comment line.

7. RECORD ELEMENT IS. As many of these clauses as necessary may be coded. The first is identified and is required. The IP identifier is the first three characters of each element name. The elements previously defined to the data dictionary are entered in the order in which they appear in the IP data definition, proceeding from left to right. A period is placed immediately following each element name.

DEFINE INPUT PROCESSING RECORD
DATA BASE DEVELOPMENT & DESIGN GUIDE

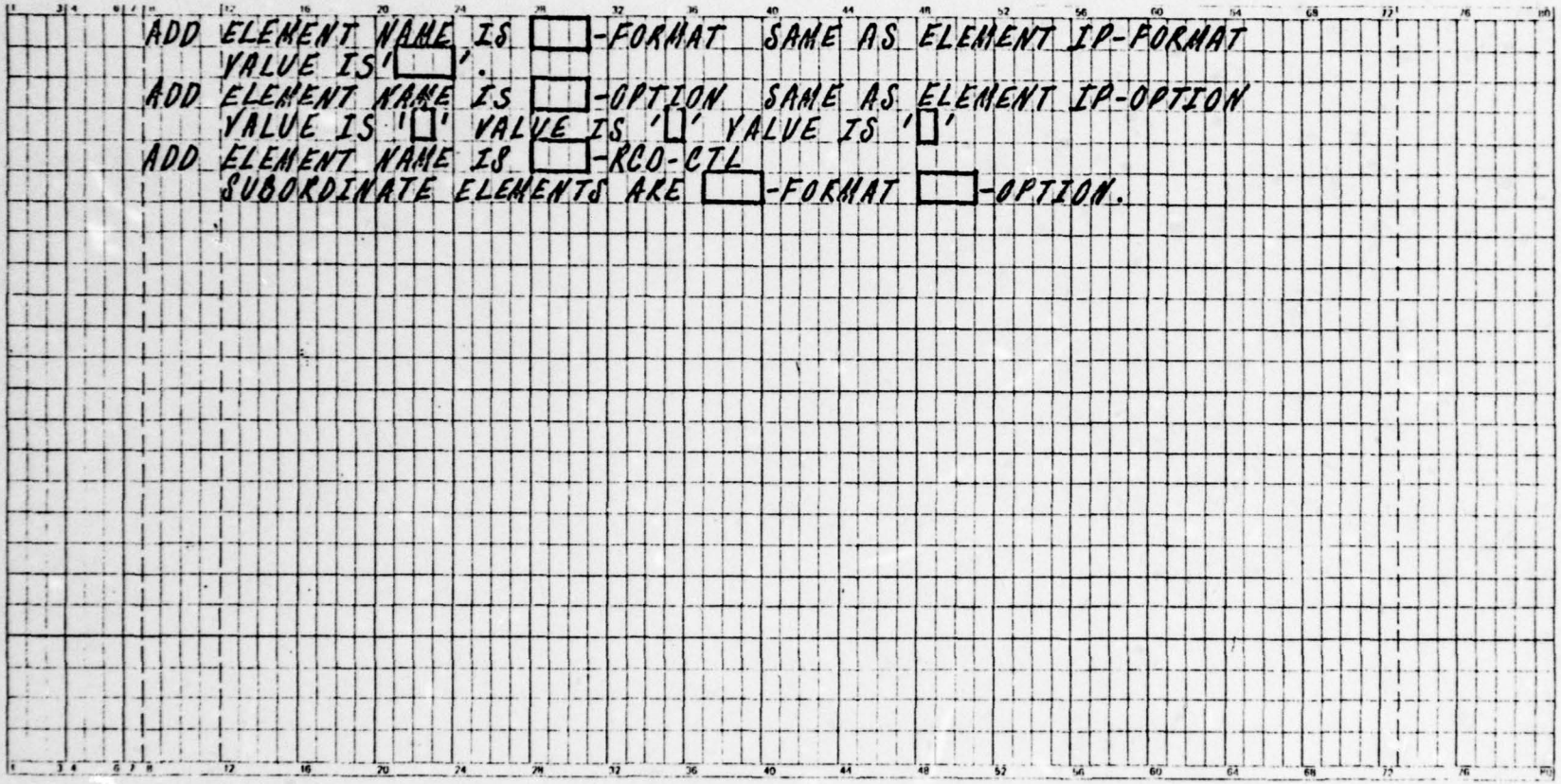
PAGE 1 OF 1



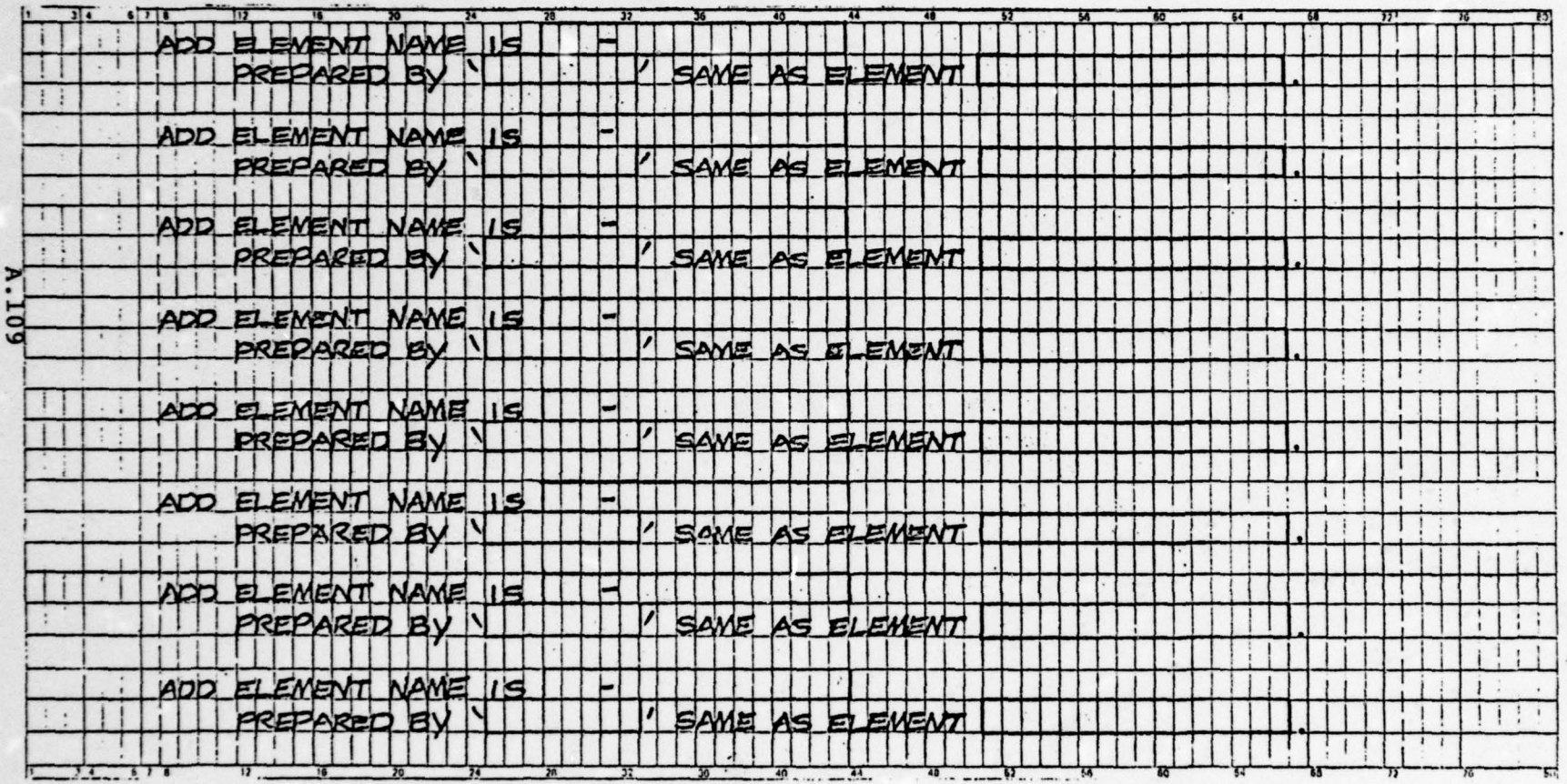
A.107

1.1/12-77

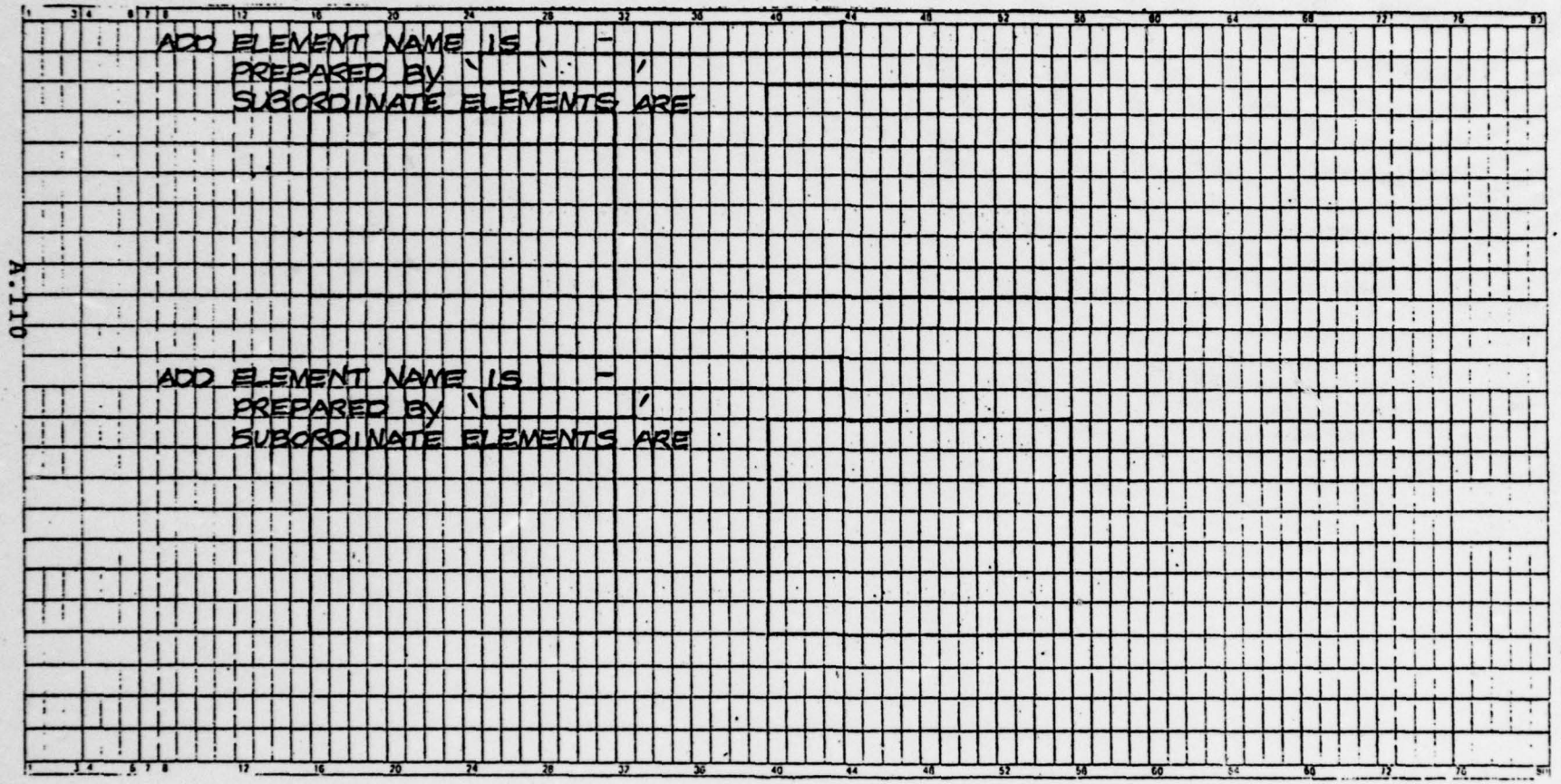
DEFINE INPUT PROCESSING ELEMENTS
DATA BASE DEVELOPMENT & DESIGN GUIDE
PAGE 1 OF 3



DEFINE INPUT PROCESSING ELEMENTS
DATA BASE DEVELOPMENT & DESIGN GUIDE



DEFINE INPUT PROCESSING ELEMENTS
DATA BASE DEVELOPMENT & DESIGN GUIDE



APPENDIX A.19.

USER CODING INSTRUCTIONS

One of the most important aspects of the implementation of a new ADP capability is communicating to the user how to use the facility. The capability is useless unless the means for its use are communicated in a form and language that is readily understood. This section describes the basic form of instructions the user must follow to prepare batch input to the data base.

The information about each input processing data line and its individual elements is stored within the data dictionary. Appendix A.18 describes the definition of an input processing module for batch input. It also defines the manner in which the data elements are associated within the dictionary to form input processing data records.

The non-ADP definition of all data elements defined to the integrated data base is stored in the data dictionary as part of the element definition defined in Appendix A.6. As each input processing data record is defined the associated data base elements are linked to the IP and the non-ADP definitions of those elements are available for reporting purposes.

An introduction to each IP entry is provided as part of the documentation. This introduction is extracted from

the comments associated with the IP record definition comments.

The complete user coding instruction for each IP data entry consists of the introduction information extracted from the IP record definition, followed by each data element definition, beginning with the left-most. The format of the instruction is illustrated in Figure A.19.1.

The user information is extracted from the data dictionary but a special program will create an entry into the programming support library (PSL). The library entry may be massaged to achieve the desired user communications.

Referring to Figure A.19.1:

1. The field number is a sequential numbering of data elements from left to right within an IP line. This number is used by the IP programs to identify errors detected when validating raw data.

2. The position defines the place on the input card or other entry which contains the data element.

3. The "A/N" entry defines whether the entry is alphabetic (ALPH), numeric (NUM), or either (AN). If the entry is numeric, the values should be justified to the right end of the entry field and zero filled to the left. The other entry forms should be justified to the left of the field.

4. The "REQ/OPT" entry defines whether the field is required or optional. Processing terminates if a required element is omitted.

5. The "* DEL" entry defines whether the corresponding data field within the data base may be deleted by inserting an asterisk in the left-most position of the IP data field.

The description is begun with the name of the data element as the user knows it. The element name is in capitals to make visual identification and location easier. The name is followed by the non-ADP description extracted from the data dictionary description of the data base data element.

APPENDIX A.20 TO BE ADDED

XXX INPUT PROCESSING DATA ENTRY DESCRIPTION

----------description of input processing entry----------

DETAIL DATA ELEMENT DEFINITIONS

FLD NO	POSIT	A/N	REQ/ OPT	* DEL	DESCRIPTION
01	01-03	ALPH	REQ		NAME OF ELEMENT. Element description.
02	04-04	ALPH	REQ		NAME OF ELEMENT. Element description.
03	05-16	ALPH	REQ		NAME OF ELEMENT. Element description.
04	17-20	NUM	OPT	*	NAME OF ELEMENT. Element description.
05	21-27	AN	OPT	*	NAME OF ELEMENT. Element description.

Figure A. 19.1

APPENDIX A.21.

USER DATA BASE AUTHORIZATION.

Each user must be overtly associated with those elements within the data base which may be accessed by the user. Without such an authorization input and retrieval errors will occur during processing. While a user may be permitted access to certain data base records, access to data elements within the records is dependent upon specific authorization.

The IDD authorization entries define four interface modes which the user may assume: access, store, modify, and deletion of data elements.

1. MODIFY ELEMENT NAME IS. This clause identifies the data element which the user will be permitted to act upon.

2. INCLUDE USER IS. This clause, when used, permits the user to retrieve the data element from the data base.

3. INCLUDE USER IS xxxxx RESPONSIBLE FOR CREATION. This clause, when used permits the user to store the data element in the data base.

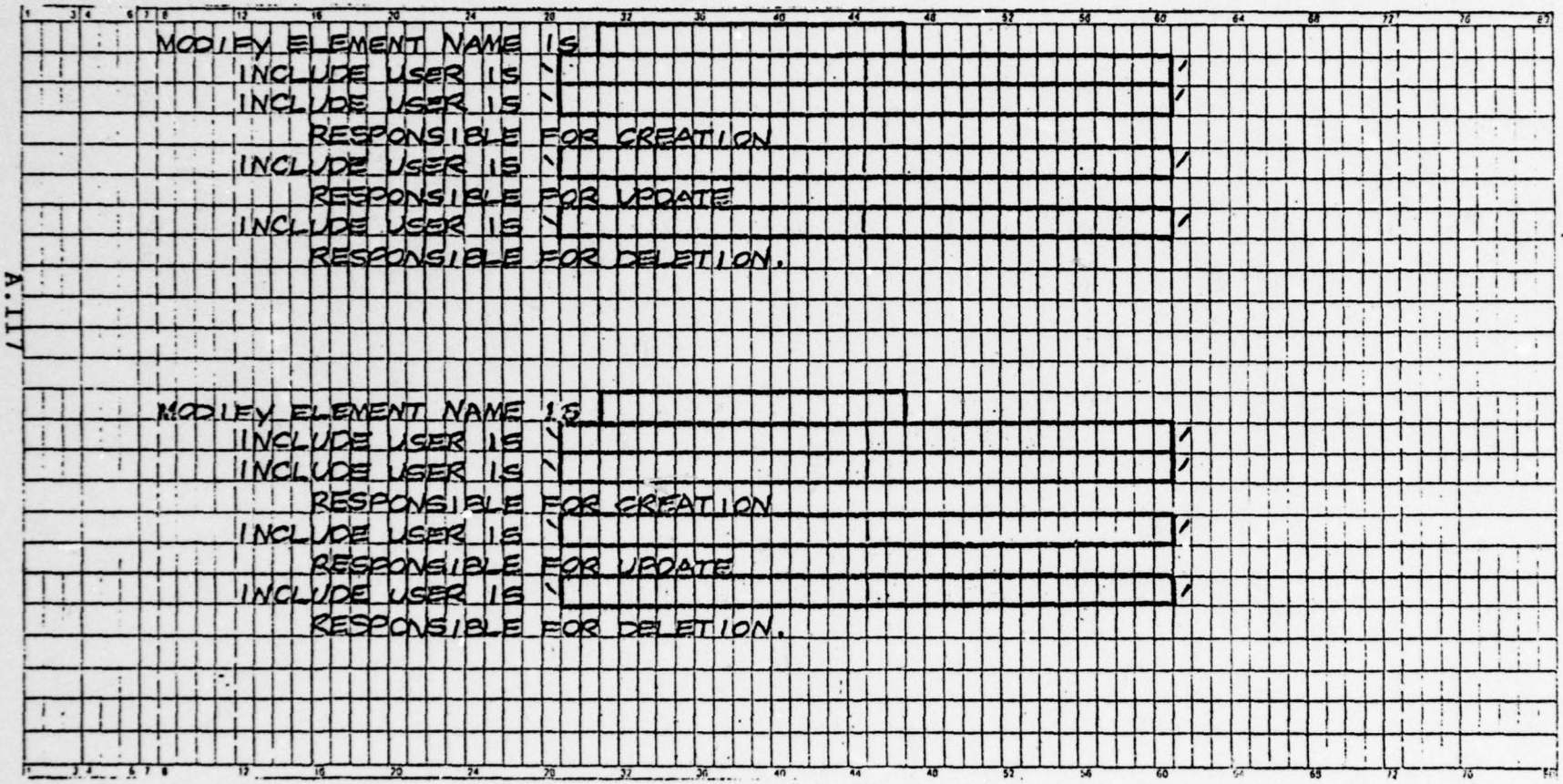
4. INCLUDE USER IS xxxxx RESPONSIBLE FOR UPDATE. This clause, when used, permits the user to modify the data element in the data base.

5. INCLUDE USER IS xxxxx RESPONSIBLE FOR DELETION.

This clause, when used, permits the user to delete the data element from the data base.

NOTE: Any combination of these entries may be used. A period must follow the last entry. If the user is permitted all three update options (creation, update, and deletion), a single clause INCLUDE USER IS xxxxx RESPONSIBLE may be used in place of the three individual entries. Cross out any unused lines to prevent extra punching.

ESTABLISH USER RESPONSIBILITY TO DATA
DATA BASE DEVELOPMENT & DESIGN GUIDE



APPENDIX A.22.

ON-LINE DISPLAY SCREEN SPECIFICATION.

Each application which will process data in an on-line mode will require that displays be developed to support on-line input processing.

This appendix defines the methodology of documenting the display screen and the data elements which it will support.

The initial definition is that of the display screen template. This template illustrates the literals and data elements as they will appear on the screen. The statements described below are IDD entries for a dictionary module:

1. ADD MODULE NAME IS. This clause defines the name of the display module. The name is normally a four-character identifier assigned by the DBA staff, followed by the letter "M" as a suffix.

2. PREPARED BY. The initials of the preparer are entered.

3. MODULE DESCRIPTION IS. This clause identifies the module with an expanded name. The name will appear on the first line of the display screen.

4. ENTRY-SECURITY IS ENTRY-. This clause identifies the security classification of the module entry itself.

See Appendix E.1 for a list of acceptable classifications.

5. DATA-SECURITY IS DATA-. This clause identifies the security classification of the data to be displayed by the module display. See Appendix E. 1 for a list of acceptable classifications.

6. SCREEN-FUNCTION IS. This clause defines the input proocessing functions to be performed by the processing screen. See Appendix E.12 for a list of valid functions.

7. SYSTEM-USER IS. Multiple occurrences of this clause may be coded. The name of the user personnel for the screen are entered.

8. COMMENTS. This multiple line clause describes the screen display and its purposes in detail. Eight lines are provided on the coding form. As many additional lines as necessary may be added. The last line must terminate in a single quote mark followed by a period ('.).

The second page of the module definition contains the display as it will appear on the CRT screen. Line one of the display is pre-formatted and may not be altered. Lines 23 and 24 are reserved and may not be formatted.

The screen identifier is entered in the four-character

area beginning in position 2.

Positions 6 through 15 are used to hold the processing option for the screen. The default option is coded. Options are:

1. Store. Bulld and store a record in the data base.

2. Modify. Retrieve an existing record from the data base and modify it. Return the modified record to the data base.

3. Delete. Retrieve an existing record from the data base and delete it from the data base.

4. Insert. Associate two records resident in the data base.

5. Remove. Break the association of two records in the data base.

6. Relate. Same as Insert.

7. Separate. Same as Remove.

Position 17 contains the error cycle count. This position is suppressed and not seen by the user. It is initialized at a value of zero and is incremented by the error controller routine to control the number of attempts a user is permitted to correct errors.

Positions 20 through 59 contain the name of the screen. This may be coded directly into the screen when the layout is prepared or copied from the MODULE DESCIPTTION entry in

the screen module.

Literals to be displayed are coded exactly as they will appear on the screen. All data fields are shown as a series of plus signs (+). The data field must include the preceding attribute byte in its length.

The third page of the definition consists of definitions of the data fields in the form of comments lines. These will be placed behind the COMMENTS header statement and before the main comments associated with the screen. The field definitions identify the specific data base elements which are to be supported by the screen. The elements are listed in the order they are found on the screen, beginning in position two of line two and reading left to right, top to bottom.

Prior to storing the display screen module within the data dictionary, it will be stored in the program support library (PSL) to facilitate easier modification.

The fields in the data element definition comments are:

1. A sixteen-character field to contain the name of the data base data element being supported, followed by

2. The line number and line position of the attribute byte which begins the data element display space on the screen, followed by

3. The length of the display space on the screen, including the attribute byte, followed by

4. The character "Y" if the field is numeric, followed by

5. The attribute description. The attribute description consists of one or more entries as defined in Appendix E.13. A right parenthesis ')' must terminate the list. Each entry is separated from the next by a comma.

6. The required/optional flag. A "R" is entered if the field must contain data when used by the screen.

The fourth page of the module description defines additional information formatted as comments. These comment lines should immediately follow those on the third page (data element descriptions).

The "FOLLOWING SCREENS" section of the comments provides screen transfer information:

1. Positions 12 through 15 identify the transaction of the system which is to be executed next depending on the condition defined.

2. Positions 17 and 18 define the condition of execution to be performed. The condition codes are:

a. Numeric identifiers "01" through "16" define

the terminal function keys.

b. Identifies "A1" through "A6" define terminal attention keys.

c. The code "EN" defines the enter key.

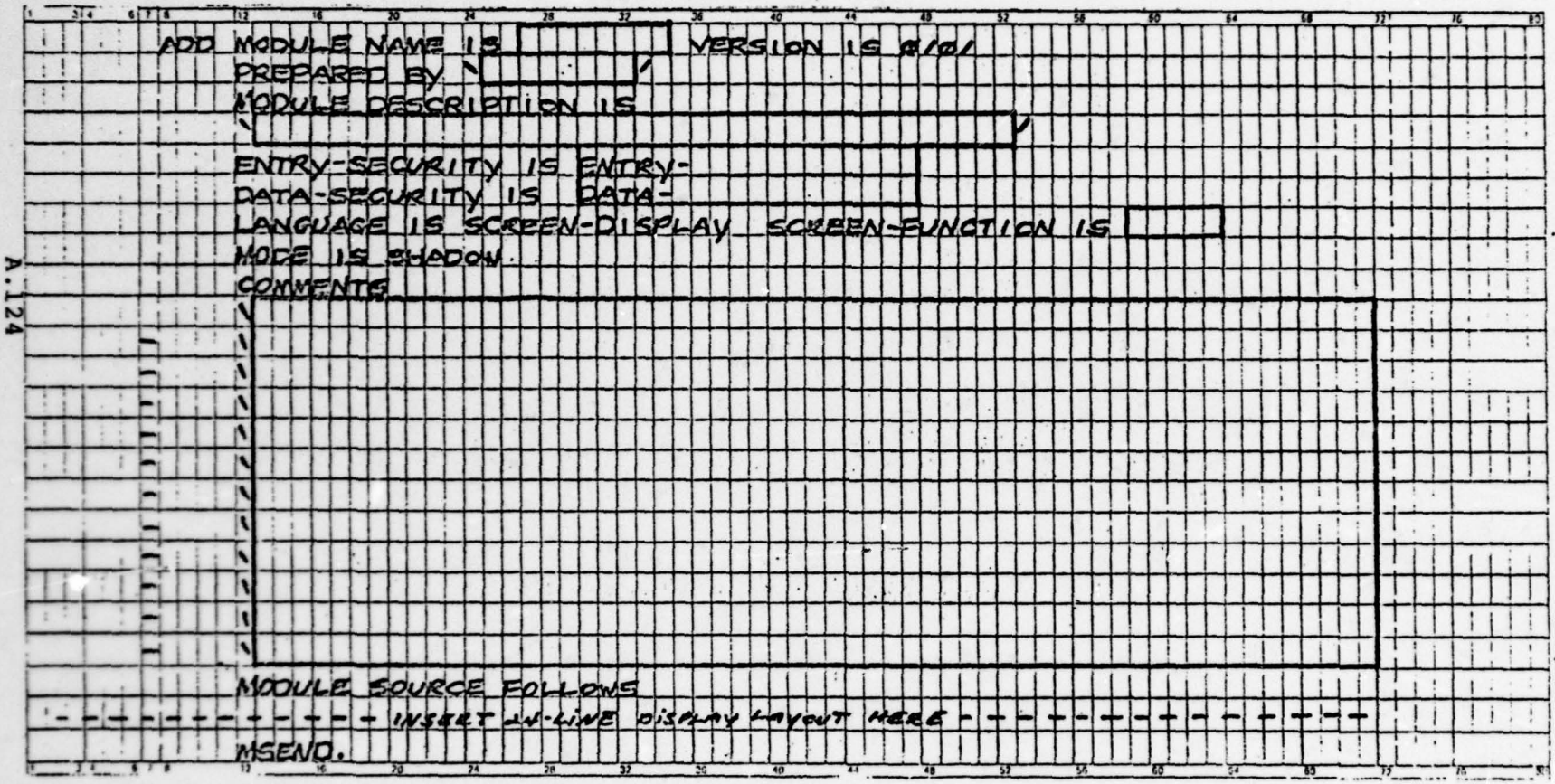
d. The codes "IX", "WX", "CX", "EX", and "DX" define levels of error within the input processing program.

3. Positions 21 through 59 contain a literal value to be placed on the screen instructing the operator to use certain terminal keys to achieve end results. The literal, as entered, is displayed on the second and subsequent lines following the last normal display line.

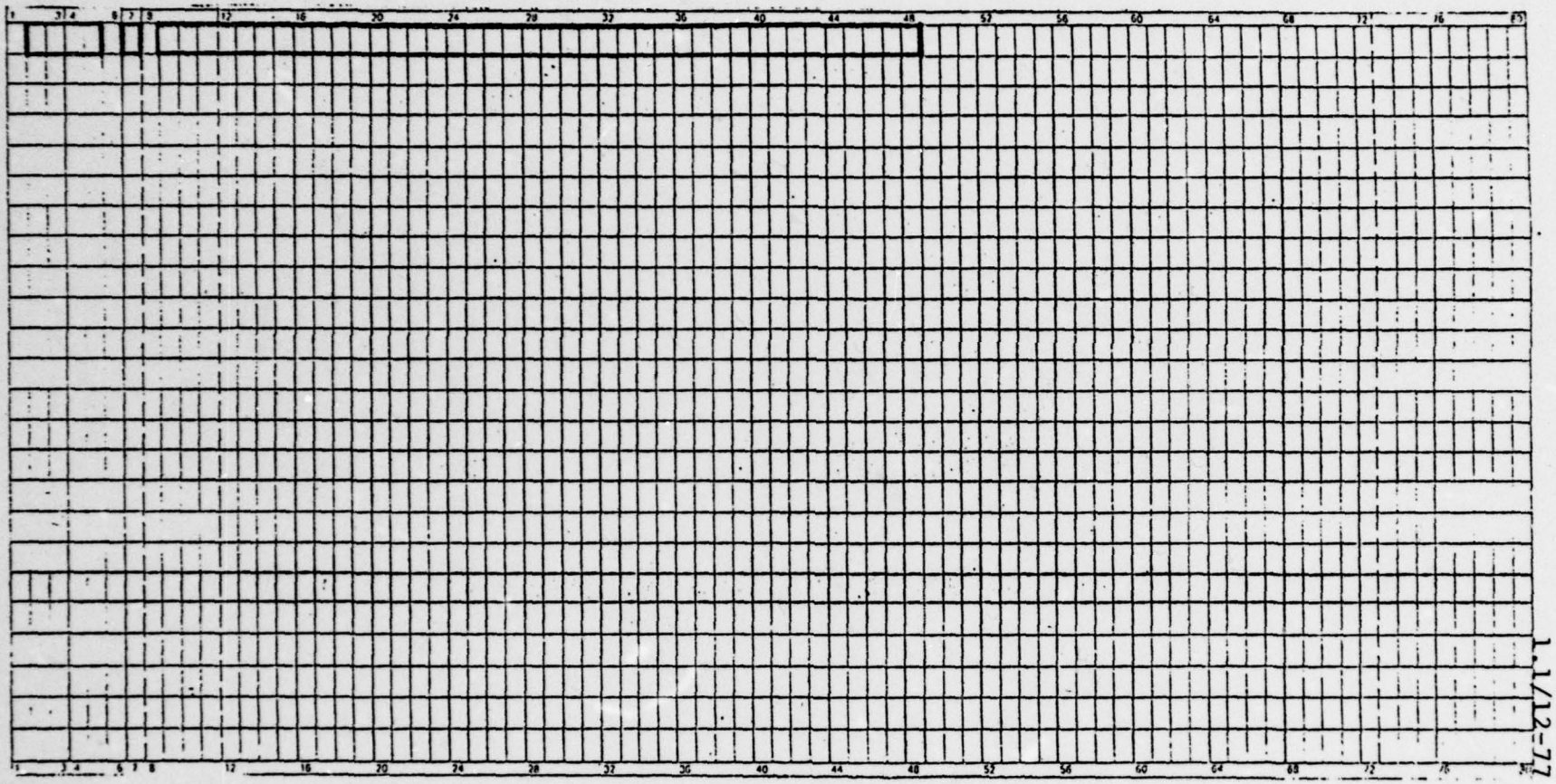
The "OBJECT RECORD TYPE" clause identifies the data base record type which is to be acted upon by the screen where that record type is not overtly identified by data elements. The name of the object record type must correspond to a valid record in the data base definition.

ADD DISPLAY SCREEN DESCRIPTION
DATA BASE DEVELOPMENT # DESIGN GUIDE

PAGE 1 OF 3



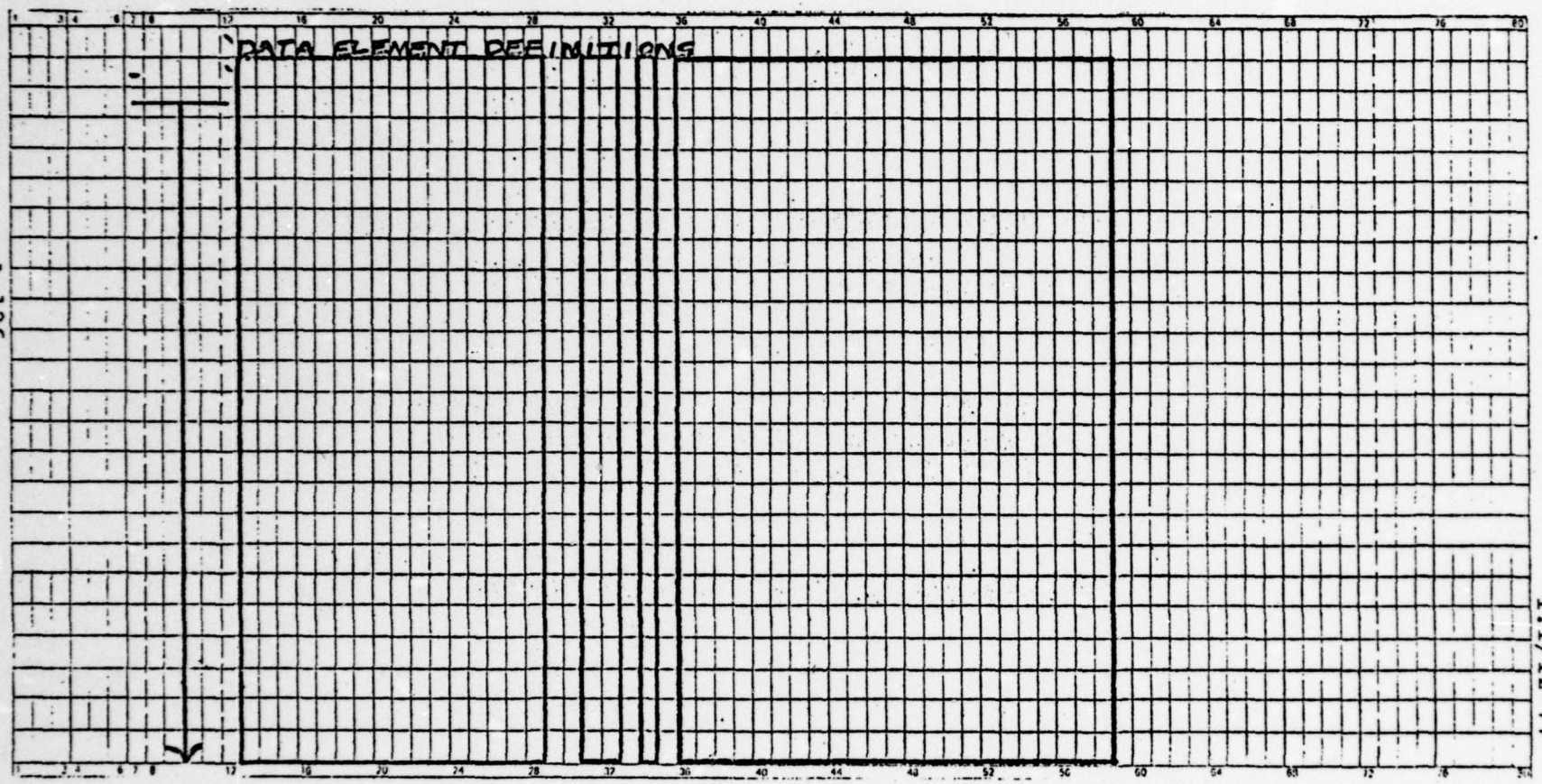
ADD DISPLAY SCREEN DESCRIPTION
DATA BASE DEVELOPMENT & DESIGN GUIDE
PAGE 2 OF 3



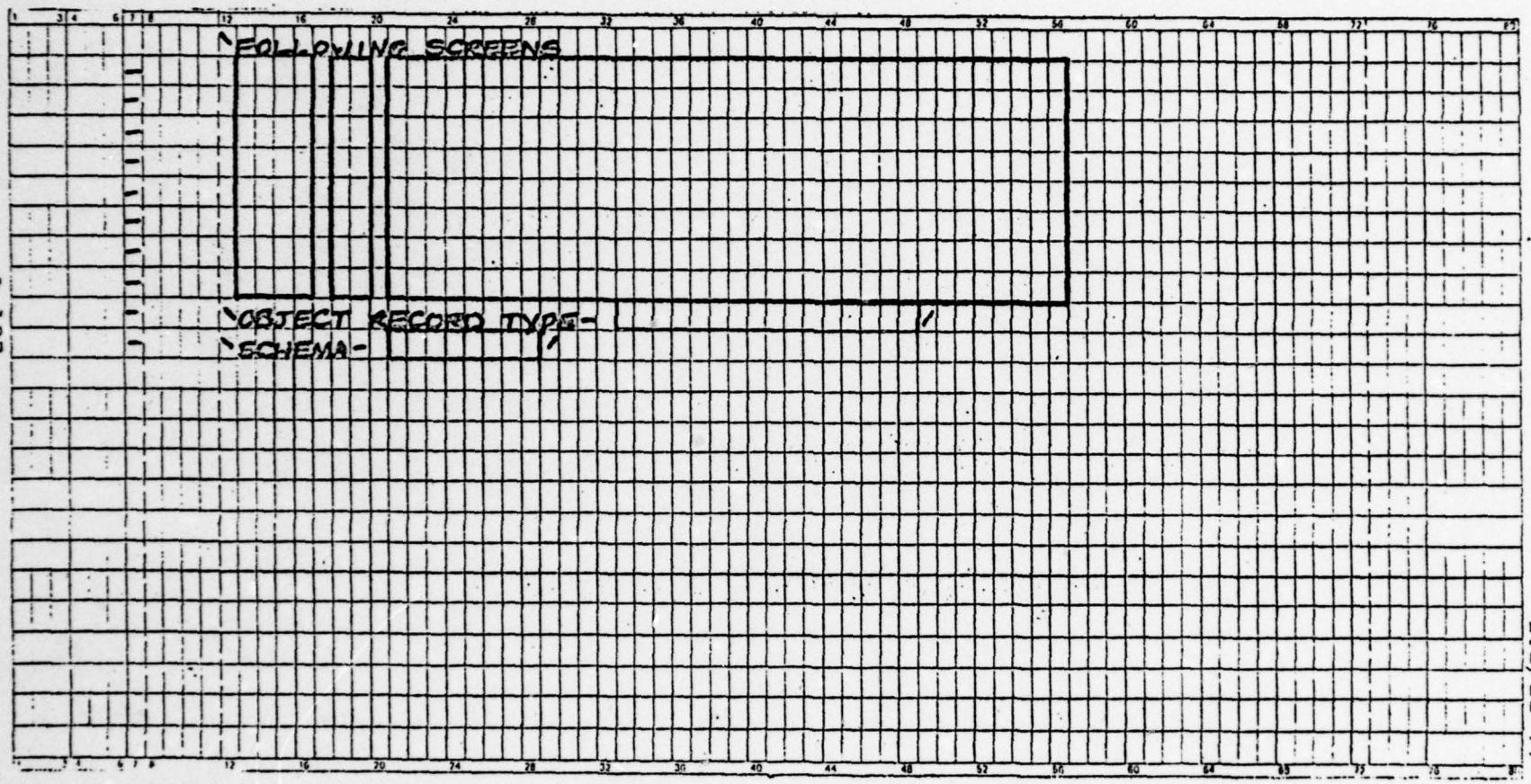
1.1/12-77

FOD DISPLAY SCREEN DESCRIPTION
DATA BASE DEVELOPMENT & DESIGN GUIDE

PAGE 3 of 3



ADD DISPLAY SCREEN DESCRIPTION
DATA BASE DEVELOPMENT & DESIGN GUIDE
PAGE 4 OF 4



APPENDIX A.23 TO BE ADDED

APPENDIX A.24 TO BE ADDED

APPENDIX B

CONVENTIONS

APPENDIX B.1.

DATA ELEMENT NAME CONVENTIONS, GENERAL.

It is important to consistency within the structure and representation of the contents of the data base that data elements be named and addressed in a uniform mannner. The data element naming conventions described here are established to meet this basic requirement. Where an unusual condition makes naming a data element without regard for these conventions, written approval of the Data Base Administrator must be obtained.

Conventions:

1. The data element name should reflect, as clearly as possible, the purpose or contents of the data element.

2. The data element name will not exceed 16 characters in length, including hyphens.

3. Where the element is one of the common type listed in Appendix B.2, the appropriate prefix will be used.

4. All data element names must begin with an alphabetic character.

APPENDIX B.2.

COMMON DATA ELEMENT NAME PREFIXES

Certain data elements within the data base are basically identical to other elements except for their individual usage. For example: NAME may be the name of a person, the name of a publication, the name of a company, or the name of a position, etc. All, however, are NAMEs. The same is true of DATE, where the type of date may vary widely.

These common data types are identified by a one-to-four character prefix which is appended to the data element name to identify its generic group. When used as part of a data element name the prefix is separated from the main portion of the name by a hyphen (-).

The prefixes listed in this appendix are broken into two groups: primary and secondary. Primary prefixes are used wherever applicable. Secondary prefixes are used, whenever applicable, if a primary prefix is not also used. Use of the secondary previx is optional if a primary prefix is also used. Where both primary and secondary prefixes are used, the hyphen between the prefixes may be omitted.

Primary prefixes are:

1. CTL. This prefix is used only when a data base record is to be protected from unauthorized use. The data element contains a numeric value corresponding to the

protection key of the user who stores the record in the data base.

2. RCD. This prefix is used only when a data base record is to be protected from unauthorized use. It is used in conjunction with CTL above.

3. OCRS. This prefix is used to describe the control field of a record with an OCCURS-DEPENDING-ON group.

4. DATE. This prefix is used for all data elements and groups defining dates. The form of date: calendar, computed, or Julian, is optionally indicated as J (Julian), C (computed), or G (Georgian calendar) as an additional prefix character.

5. MO. Month of the year.

6. DY. Day of the month.

7. YR. Year of the century.

8. NAME. This prefix is used for all data elements of the generic group defined as names.

9. ADDR. Addresses.

10. CITY. A city, town, borough.

11. STAT. A state or province.

12. CTRY. Country.

13. CTY. County.

14. DATA. Data identifier.

15. DBKY. Data base key information.

16. DTG. Date-time-group.

17. HR. Hour of the day.

18. MIN. Minutes of an hour.

19. SEC. Seconds within a minute.

20. ZIP. Postal zip code.

Secondary Prefixes are:

1. SW. A switch or flag element.

2. CTR. A counter or accumulated data element.

APPENDIX B.3.

DATA ELEMENT PICTURE DEFINITION CONVENTIONS

The size and composition of data elements are commonly defined through a short-hand form called a "picture". This short-hand form may be directly translated into computer-readable format. Picture definitions are used within the data dictionary to define the structure of the data element within the data base.

The picture definition is composed of three basic element segments: (1) a qualifier; (2) the data type; and (3) data length.

The qualifier which may be used applies to numeric type data only. The letter 'S' may precede the data type to identify a numeric data type as signed. The use of a signed numeric data type permits the data element to contain negative values as well as positive values. When used the qualifier is immediately followed by the numeric data type without intervening spaces or special characters.

Three basic data types are permitted:

1. Numeric. All data within the element will always be numeric (zero through nine). Alphabetic or spectial characters entered in this field will result in processing errors. The picture symbol for a numeric data type is a nine (9).

2. Alphabetic. All data within the element will

always be alphabetic (the alphabet in upper-case and normal punctuation characters). Numeric data entered in this field will result in processing errors. The picture symbol for an alphabetic data type is the letter "A". This option is seldom used except where numeric data is specifically excluded.

3. Alphanumeric. Data within the element may consist of any of the permissable characters or digits defined by the computer. Alphabetic and numeric data may be intermixed at will without processing errors. Numeric data stored in this type of field should be used for information purposes only. Computation processing of numeric data in this data type will require extra handling by the computer and will result in slower processing. The picture symbol for an alphanumeric data type is the letter "X". This is the most commonly used data element definition.

The data element length is defined following the data type segment. The length is subject to certain rules:

1. Numeric data elements may not exceed 18 digits in length.

2. Alphabetic or alphanumeric data elements may not exceed 2000 characters in length without approval of the DBA.

Data element length is identified by a numeric value enclosed in parenthesis. This value may range from one

through the limits defined above, e.g., "(12), (8)". The size segment immediately follows the data type segment without intervening spaces or special characters.

The complete picture definition for the various types of data are illustrated below:

1. Numeric. S9(8), 9(8).
2. Alphabetic. A(10).
3. Alphanumeric. X(10).

APPENDIX B.4.

DATA ELEMENT USAGE CONVENTIONS

The usage clause defines the intended use and composition of a data element within the data base. The usage clause is a qualifier for the PICTURE clause.

Available usage clauses are:

1. DISPLAY. This clause is the default whenever the usage clause is omitted. If this usage is desired, omit the clause to reduce punching volume.

2. COMP. This clause is used to define numeric data types which will use a special form of numeric representation within the data base. This usage type should be used by the DBA staff only.

3. CONDITION-NAME. It is sometimes useful to define a phrase which identifies a condition or value of the data element. This condition is met whenever the data element contains the value within the data base record which matches the value associated with the condition name. The DBA staff should be consulted prior to defining any condition names.

APPENDIX B5.

INITIAL VALUE DEFINITION

It is desirable that an initial value be provided for all data elements. This assures that garbage data values will not be loaded to the data base. Initial values are typically either SPACES (blanks) for alphabetic or alphanumeric data elements or (0) for numeric data elements. If the numeric item is signed (a 'S' in the qualifier segment) the initial value should be "+0".

The length of the value data must not exceed the length of the field as defined in the *PICTURE clause*. *Therefore*, if the field is ten characters long, the value literal must not exceed ten characters. If an alphabetic or alphanumeric literal contains spaces or special characters (characters other than A-Z, 0-9, and hyphen, period, comma, it must be enclosed in single quote marks ('). In no case may the value literal exceed 34 characters in length.

APPENDIX B.6.

DATA BASE RECORD L0CATION MODE CONVENTIONS

The manner in which records are located within the data base is important to the effectiveness of their use. The intended use and speed of retrieval is the key to the location mode used.

Two location modes are permitted with the integrated data base approach in use:

1. CALC. This location mode identifies the record type as being randomly located within the data base. This mode is used primarily for owner records which are to be used as entry points into the data bsse. While any record type, regardless of whether it is an owner or member, may be defined as CALC, it is dangerous to define both an owner and its member(s) as CALC. This type of assignment can result in slower than normal updating speed.

When defining a record type as CALC it is particularly important to select a data element/group within the record which will readily identify the record, preferably uniquely, to both the data base system and the user. It is not desirable to define a data element as the record key if it is subject to frequent changes in value within the record occurrence. While the data base system permits the value of the key field to be modified such a modification often alters the random location

where the record is to be found in the data base. The data base system does not physically relocate the record. It simply assigns a pointer from where the record should be located in the data base to where it is actually located. This will increase the amount of searching required to locate the record and degrade retrieval performance. A small number of key value changes is not harmful. The danger comes when the percentage of key changes anticipated exceeds 10 percent.

2. VIA. This location mode is used primarily to define the storage approach for member records. The VIA mode tells the data base system to store a member record as close as possible to its owner record occurrence in the data base storage area. This has the effect of reducing the retrieval time required to obtain an owner record occurrence and one or more of its members. This mode is not usable if the member record is not stored as an automatic member of the owner occurrence. That condition confuses the data base system because it does not know which owner occurrence to place the member beside.

APPENDIX B.7.

DATA BASE SET NAMING CONVENTIONS.

Each association between two record types within the data base is known as a 'set'. A set may contain one and only one owner record type. The set may contain as many member record types as the user desires. The number is limited only by the practicality of data retrieval speed. Speed decreases as the number of record occurrences, regardless of type, within a set.

The set naming convention adopted breaks the set name into three segements, each separated from the others by a hyphen (-):

1. The application acronym. This field segment is the left-most portion of the set name. It may contain from one to nine characters, beginning with an alphabetic character. The acronym should identify the application as definitively as possible. Sets are physically placed in the data base definition (schema) in alphabetic order. This approach groups sets by application.

2. The owner identifier. This segment consists of the four-digit identifier assigned to the owner record.

3. Sequence. This segment consists of the alphabet from A through Z identifying each separate set associated with the application and owner record. For example: the first set

within an application and associated with a specific record type as owner is given the sequence "A"; the second "B"; etc.

An example of a set definition is:

ACCTNG-1012-A

where

ACCTNG identifies the application which the set supports, in this case the accounting application.

1012 identifies the owner record type of the set.

A defines the set as the first one for record type 1012 within the accounting application.

APPENDIX B.8.

IP NAMING CONVENTIONS

Each input processing (IP) module is identified as a unique entity through a defined name. This name is composed of three primary characters and, optionally, one secondary character.

The three primary characters are assigned by the DBA staff. They are normally alphabetic. The second and third characters may be numeric if approved by the DBA. The first character must always be alphabetic and will conform to the following basic scheme:

1. A - Management-oriented processing
2. B - Scientific and technical data base processing
3. Y - Support services processing.
4. Z - System-oriented processing.

The fourth optional character identifies the processing option to be performed by the IP. This optional character is used when the size of the IP module, including several options, exceeds allowable maximums. Allowable characters are:

1. S - Store a record in the data base.
2. M - Modify an existing record in the data base.
3. D - Delete an existing record from the data base.
4. I - Insert/associate two records in the data base.
5. R - Remove/disassociate two records in the data base.

6. Q - Retrieve a record from the data base.

APPENDIX B.9.

RECORD NUMBERING CONVENTIONS.

Data base records are identified both by name and by number. The number identifier ranges from 0001 through 9999. Each number has significance with respect to the pupose and security classification of the record to which it refers.

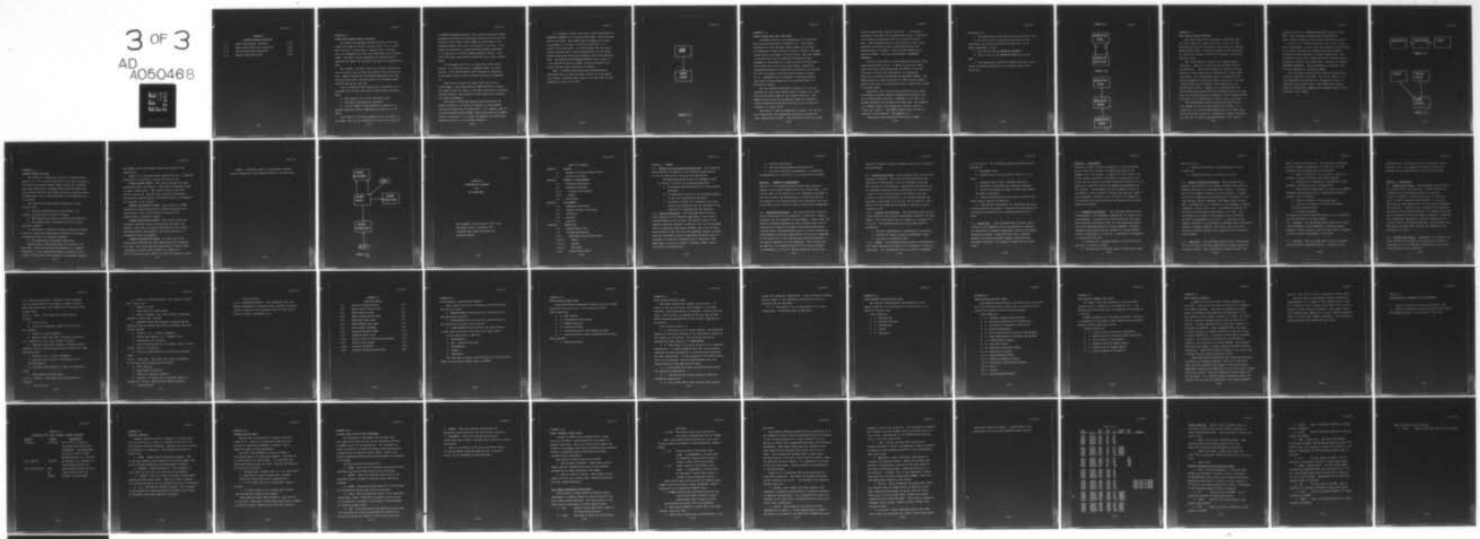
Number conventions are:

1. 0001-0999 System reserved records.
2. 1000-3999 Unclassified application records.
3. 4000-4999 Confidential application records.
4. 5000-7999 Secret application records.
5. 8000-8999 Top Secret non-compartmented application records.
6. 9000-9999 Top Secret compartmented application records.

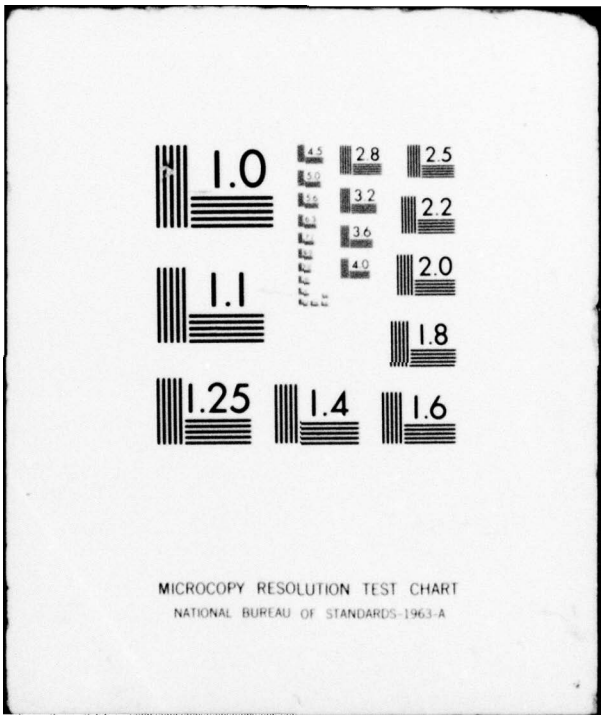
AD-A050 468 NAVAL INTELLIGENCE PROCESSING SYSTEM SUPPORT ACTIVITY--ETC F/G 5/2
INTEGRATED DATA BASE DEVELOPMENT AND DESIGN GUIDE. VERSION 1.1.(U)
DEC 77 L E TOWNER

UNCLASSIFIED NL

3 OF 3
AD A050468



END
DATE FILMED
3-78
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

APPENDIX C

STANDARD RECORD STRUCTURES

APPENDIX C.1.

SINGLE OWNER/MEMBER RECORD STRUCTURE

The single owner/member record structure is the most common and simplest record structure within the data base. This structure is used where a group of data, occuring once, is accompanied by other data which may occur multiple times. The owner record represents the single group of data and the member the multiple occurrences of accompanying data.

For example: An owner record may contain information about a person, such as name and address, which occurs only once. Member records may be defined identifying the jobs the person has held. One occurrence of the member record would be stored for each job.

Prior to defining this structure as illustrated it is necessary that certain questions be asked about the member data:

1. How large is it in total character size?
2. How many occurrences are possible?
3. Can the number of occurrences be limited to a specific quantity without impairing the usability of the data?

The answers to the above questions will determine if the member data can be included within the owner record as

an OCCURS DEPENDING ON group. This option eliminates pointer overhead and additional processing time during retrieval but places a finite limit on the number of occurrences of the repeating member data which is practical to include. If the number of characters of repeating data exceeds 1000 bytes, it is preferable to define separate member records. Very large data base records make inefficient use of data storage space.

If the member data is to be shared by another owner record, it is necessary that a separate member record be defined. This shared member record becomes an adaptation of the many-to-many record structure described in Appendix C.3.

Most member records are stored "VIA" their owner record types. This means that the DBMS physically locates the member record as close to its owner occurrence as possible in data storage. This results in less operational overhead and faster processing.

The need to access the member record determines the manner in which it is stored in relation to other member record occurrences for the same owner. If a logical order is to be maintained between member record occurrences the SORTED option is selected. If no order is required the NEXT option should be used as it is the most efficient.

It is common to define the member record association as MANDATORY AUTOMATIC if a simple owner/member record relationship is involved. This means that the member will always be attached to its owner record occurrence when it is stored in the data base. It further means that the association with the owner record occurrence may not be broken without physically deleting the member record from the data base. The association OPTIONAL AUTOMATIC may be used if it is desired to permit a member record occurrence to be moved from one owner occurrence to another.

NOTE: A member *record* type must be automatically associated with at least one owner record if it is stored in VIA mode. Access to the record in the data base is very difficult if this is not done.

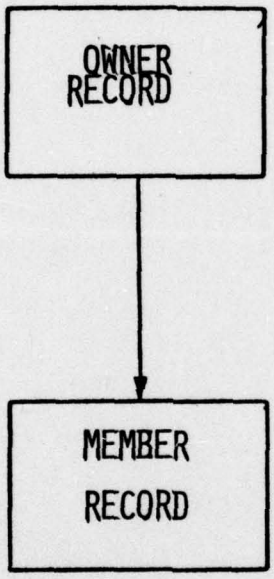


FIGURE C.1.1

APPENDIX C.2.

DBOMP (RECORD OWNS SELF) STRUCTURE.

The DBOMP structure is an adaptation of the many-to-many structure described in Appendix C.3. The primary difference is that the owner record types are the same. The advantage of this structure is that it permits the same record type to own others of its own kind. The term "DBOMP" is derived from "bill of materials processing" where equipment is described as a series of parts and each part may be made up of one or more other parts. Applied to business processing it is possible to describe an organization chart by using a structure such as shown in Figure C.2.1. Organizations are composed of sub-organizations which may continue downward for an unlimited number of hierarchial levels.

The two records illustrated in Figure C.2.1 are the equivalent of the three records shown in Figure C.2.2 with the top and bottom types of this figure identical. The DBOMP structure permits an infinite number of subsidiary levels of definition whereas the Figure C.2.2 structure is limited to a single level.

Two sets are used to connect the structure: "A" and "B". Set A identifies the organization record as the owner of other organization records. The association record is stored

VIA the organization record on this set. It contains, internally, the name of the organization which is subsidiary to the owning organization. The set is sorted on this name. This technique permits rapid access to owned organizations, alteration of the relationship between two organizations, or simply, a list of those organizations owned by another. The set relationship is MANDATORY AUTOMATIC.

The B set identifies, to the organization record, other organizations which own it. A search of this set identifies all other organizations who are higher in the hierarchy and to which this organization is responsible. This set is normally stored NEXT and MANDATORY MANUAL. The MANUAL option is used to satisfy the DBMS currency requirements at the time the association record is physically placed in the data base.

Basically, then, this record structure is very useful in defining infinite levels of repeating hierarchies, such as an organization structure. Secondly, it is useful to simply associate two records of the same type. For example: In a keyword index, two keywords are related, such as "ship" and "tanker." The DBOMP structure may be used to associate these keywords. See Appendix C.4.

Conventions associated with definition of DBOMP

structures are:

1. The association record will be store VIA the "A" set.

2. The "A" set will be sorted on the field in the association record which contains the identifier of the owned (subsidiary) record.

3. The "A" set will be MANDATORY AUTOMATIC.

4. The "B" set will be MANDATORY MANUAL and stored NEXT.

5. The association record will contain the name of the record occurrence subsidiary to the primary owner record occurrence.

FIGURE C.2.1

1.0/11-77

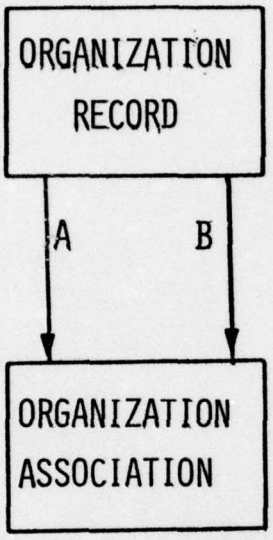


FIGURE C.2.2

APPENDIX C.3.

MANY-TO-MANY RECORD STRUCTURE

Data base design often encounters a condition where two record types are closely related in such a way that each may be related to the other in more than one instance. Stated another way: A single occurrence of one record may be related to multiple occurrences of another record type and vice versa.

This relationship is known as the "many-to-many" structure. Figure C.3.1 illustrates the basic form of the structure. Three record types are required, the two which are interrelated and an "intersection" record which joins them. The intersection record is created for the express purpose of joining two owner occurrences. As such it may consist of the minimum record definition, four characters of filler. However, it is possible that the two records being joined have data elements which occur only when the joining condition is present. These elements should be placed in the intersection record if this is the case.

The intersection record is normally stored VIA the most common of the two record types to be joined. If a sorted set is to be included between one of the owners and the intersection record, it is advisable to store the record via that set to improve processing speed. The record is

normally defined as a MANDATORY AUTOMATIC member of both sets. This technique automatically associates the owner records together by storing the intersection record.

Figure C.3.2 illustrates another form of the many-to-many relationship. In this case, the purpose of the structure is to reduce redundancy of commonly used data elements instead of relate multiple to multiple occurrences. Here the member record type is associated with the owner which is primarily relates to the member data. This set is normally MANDATORY AUTOMATIC. The relationship to the other owner is often OPTIONAL and frequently MANUAL.

The two forms of many-to-many record structures described are contrasts in one way. The first, Figure C.3.1, uses additional storage space to accomplish its goal of associating two record types. The second saves storage space by eliminating redundany data elements common to the two owner record types.

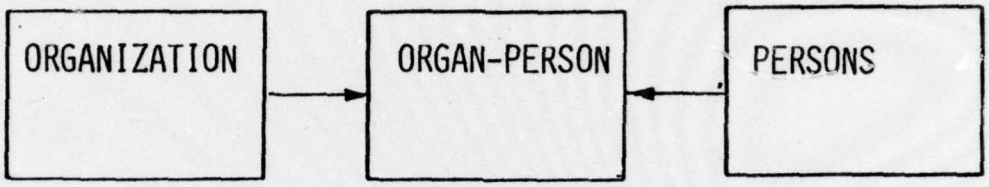


FIGURE C.3.1

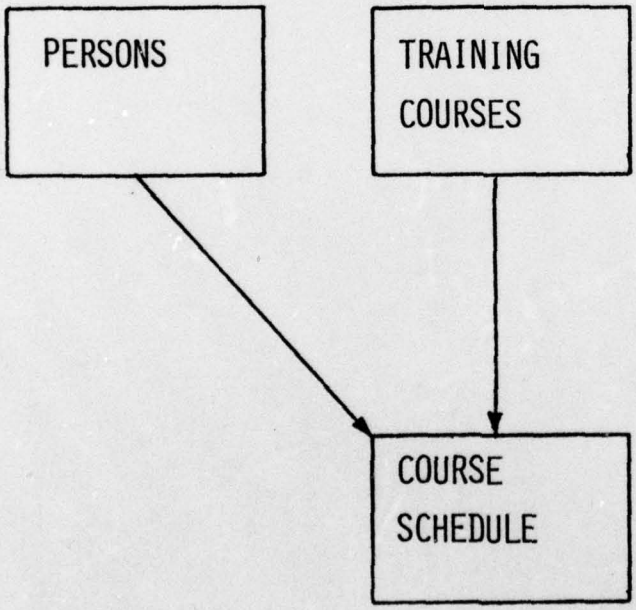


FIGURE C.3.2

APPENDIX C.4.

KEYWORD RECORD STRUCTURE.

The ability to reference a data base through defined keywords is a very valuable addition to the system capability. The feature has become common enough within the integrated data base system that a standard data record structure has been defined. Whenever the keyword facility is desired within an application, the structure defined in this appendix will be utilized.

The capabilities established through use of this structure are:

1. Multiple keywords may be interrelated. For example, airplane may be related to Boeing.

2. Each keyword may contain an associated definition.

3. The keyword may be associated with one or more data base records.

4. The phrase or sentence extract in which the keyword occurs in a data base record may be stored in the keyword intersection record for context reference.

5. The keyword may be searched generically.

Keyword master record definitions are stored in the data base. These definitions make addition of a keyword capability to a new application easier by permitting the essential information and parameters to be copied, changing

the element and record names to make them unique to the application.

Figure C.4.1 illustrates the record structure. A detailed description of individual records and sets follows.

KEYWORD MASTER RECORD. This record contains the basic information about the keyword. The record is randomly stored to permit rapid access. The keyword itself is the CALC key for the record. The date the record was established, the security classification, and handling code of the keyword are included in this record.

KEYWORD RELATIONSHIP RECORD. This record is a DBOMP intersection record which permits multiple keywords to be interrelated. The record contains the name of the keyword which is owned by another.

KEYWORD DESCRIPTIONS RECORD. This record allows the storage of multiple lines of description data about the keyword. More than one type of description may be stored. Each description record permits up to 600 characters of information about the keyword.

KEYWORD INTERSECTION RECORD. At least one of this record type is defined for each application where keywords are used. The record contains the ability to extract from a sentence or clause which contains the keyword to aid in establishing the context in which the keyword is used.

INDEX. A secondary index is established to permit generic searching of the keywords stored in the data base.

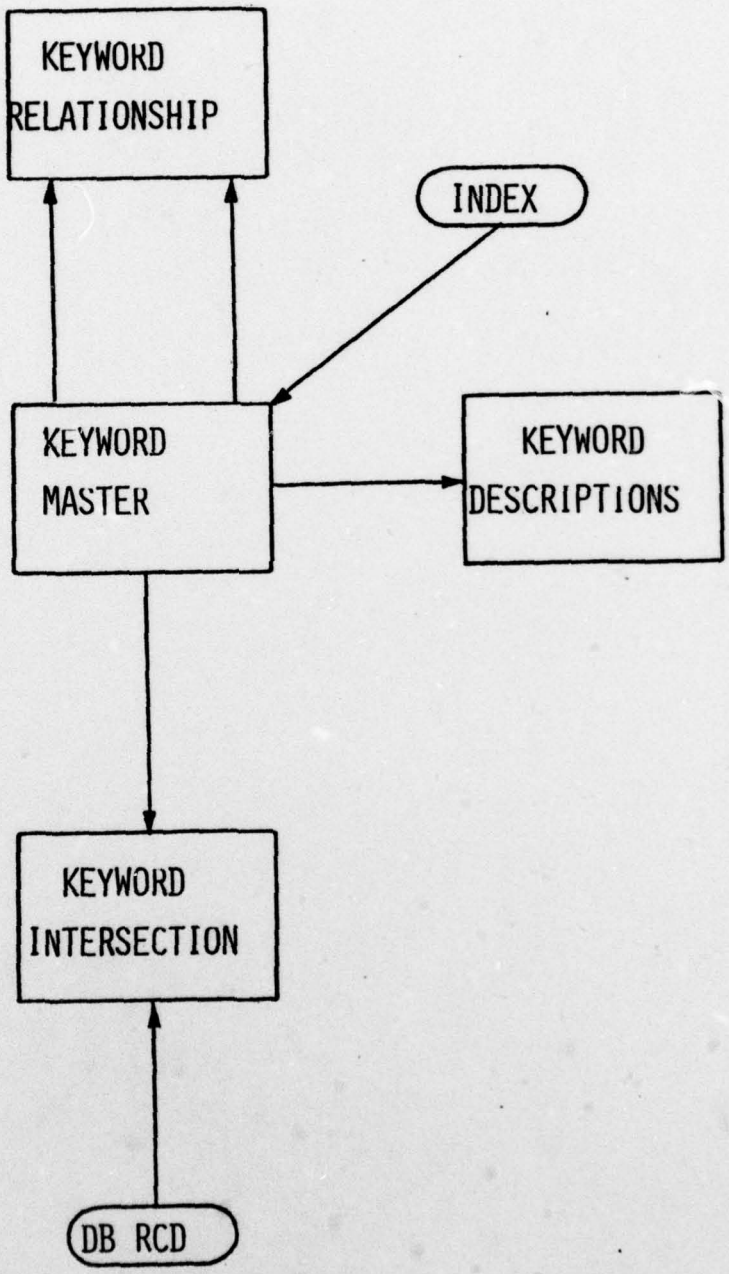

KEYWORD
RELATIONSHIP
INDEX
KEYWORD
MASTER
KEYWORD
DESCRIPTIONS
KEYWORD
INTERSECTION
DB RCD

FIGURE C.4.1

APPENDIX D

DOCUMENTATION STANDARD

FOR

ADP SUBSYSTEMS

This appendix is extracted in total from

DOD Manual 4120.17, December 1972

Automated Data System Documentation

Standards Manual

TABLE OF CONTENTS

SECTION 1. GENERAL.

1.1. Purpose of the Subsystem Specification. This paragraph shall describe the purpose of the subsystem specification in the following words, modified when appropriate:

The Subsystem Specification for (subsystem name) is written to fulfill the following objectives:

a. To provide detailed definition of the subsystem functions.

b. To communicate details of the on-going analysis to the user's operational personnel.

c. To define in detail the interfaces with other systems and subsystems and the facilities to be utilized for accomplishing the interface.

1.2. Project References. This paragraph shall provide a brief summary of the references applicable to the history and development of the project. The general nature of the computer programs (tactical, inventory, etc.) to be developed shall be specified. The project sponsor, user, and the operating center(s) that will run the completed computer programs shall be indicated. A list of applicable documents shall be included. At least the following shall be specified, when applicable, by source or author, reference number, title, and security classification:

a. Function description.

b. Related system/subsystem specifications.

c. Any other pertinent documentation or significant correspondence not specified in the Functional Description.

SECTION 2. SUMMARY OF REQUIREMENTS.

Section 2 of the Subsystem Specification shall provide a summary of the system characteristics and requirements. This section shall be an expansion of the information published in the FD to reflect the determination of additional details. Any changes to the characteristics and requirements set forth in Sections 2 and 3 of the FD must be specifically identified.

2.1. Subsystem Description. This paragraph shall provide a general description of the subsystem to establish a frame of reference for the remainder of the document. Higher order and parallel systems/subsystems and their documentation will be referenced as required to enhance this general description. Included within this description shall be a chart showing the relationship of the user organizations to the major components of the system and a chart showing the interrelationships of the system components for the subsystem. These charts shall be based on or be updated versions of the charts included in paragraph 2.4 of the FD. The more detailed charts to be

included in Section 4 shall be based on the charts included in this paragraph.

2.2. Subsystem Functions. This paragraph shall describe the subsystem functions. There will be both qualitative and quantitative descriptions of how the subsystem functions will satisfy the requirements. Although the descriptions of the subsystem functions may be refined and more detailed as a result of the on-going analysis and design, they must maintain a direct relationship to the system functions established in paragraph 2.3 of the FD, and be stated in such a manner that the subsystem environment in Section 3 can be related to them.

2.2.1. Accuracy and Validiity. This paragraph shall provide a description of the accuracy requirements imposed on the subsystem. The requirements will be related to paragraph 3.3.1 of the FD. The following accuracy requirements must be considered:

a. Accuracy requirements of mathematical calculations.

b. Logical and legal accuracy of alphanumeric data.

c. Accuracy of transmitted data.

2.2.2. Timing. This paragraph shall provide a description of the timing requirements placed on the subsystem, if they are applicable. The requirements will be related to paragraph

3.1.2 of the FD. The following timing requirements may be considered:

a. Throughput time.

b. Response time to queries and to updates of data files.

c. Response time of major subsystem functions.

d. Sequential relationship of subsystem functions.

e. Priorities imposed by types of inputs and changes in modes of operation.

f. Timing requirements for the range of traffic load under varying operating conditions.

g. Interleaving requirements for sequencing and interleaving programs and systems (including the requirements for interrupting the operation of a program without loss of data).

2.3. Flexibility. This paragraph shall provide a description of the capability to be incorporated for adapting the subsystem to changing requirements, such as anticipated operational changes, interaction with new or improved systems, and planned periodic changes. Components and procedures designed to be subject to change will be identified.

SECTION 3. ENVIRONMENT.

This section shall provide an expansion of the environment given in the FD to reflect the additional analysis and changes to the environment. Changes in the environment that do not affect the scope of the project as described in the FD and are the result of on-going analysis and design will be explicitly identified within the appropriate paragraphs of this section. These changes will be discussed in terms of the impacts on the currently available environmental components (equipment, software, etc.) as well as the impacts on the estimates and functions which were based on the original planned environment.

3.1. Equipment Environment. This paragraph shall provide a description of the equipment required for the operation of the subsystem. Included will be descriptions of the equipment presently available as well as a more detailed discussion of the characteristics of any new equipment necessary. Equipment requirements will be related to the requirements stated in paragraph 4.1 of the FD. A guideline for equipment to be described follows:

a. Processor(s), including number of each on/off line and size of internal storage.

b. Storage media, including number of disk units, tape

tape units, etc.

c. Input/output devices, including number of each on-off line.

d. Communications net, including line speeds.

3.2. Support Software Environment. This paragraph shall provide a description of the support software with which the computer programs to be developed must interact. Included will be both support software and test software, if needed. The correct nomenclature and documentation references of each such software system, subsystem, and program shall be provided. Included must be a reference to the languages (compiler, assembler, program, query, etc.), the operating system, and any data base management system (DBMS) to be used. This description must relate to and expand on the information provided in paragraph 4.2 of the FD. If operation of the computer programs to be developed is dependent upon forthcoming changes to support software, the nature, status, and anticipated availability date of such changes must be identified and discussed.

3.3. Interfaces. This paragraph shall provide a description of the interfaces with other applications computer programs, including those of other operational capabilities and from

other military organizations. The individual interfaces will be related to paragraph 4.3 of the FD. For each interface, the following shall be specified:

a. Type of interface, such as operator control of a terminal, program interfaces with other programs.

b. Description of operational implications of data transfer, including security considerations.

c. Data transfer requirements to and from the subject subsystem and characteristics of communications media/ systems used for transfer.

d. Current formats of interchanged data.

e. Interface procedures, including telecommunications considerations.

f. Interface equipment.

Interfaces with other subsystems which are to be developed will be described in the same manner.

3.4. Security. This paragraph shall describe the classified components of the subsystem, including computer programs, inputs, outputs, and data bases. These components will be related to paragraph 4.4 of the FD.

3.5. Controls. This paragraph shall provide a presentation of overall subsystem control. Included in this

paragraph will be controls such as record counts, batch controls, etc. If no specific controls are to be established at the subsystem level, this will be stated.

SECTION 4. DESIGN DATA.

4.1. System Logical Flow. This paragraph shall describe the logical flow of the subsystem. Logical flow of the subsystem will be presented primarily in the form of Macro flowcharts. Flowcharts will be in sufficient detail to permit computer program design. A narrative presentation, when appropriate, will be used to supplement the flowchart. Flowcharts will provide an integrated presentation of the subsystem dynamics, of entrances and exits, and of interfaces with other computer programs. Flowcharts will effectively represent all modes of operations, priorities, cycles, and special handling. The flowcharts will show data flow from input, through the subsystem to the generation of output.

4.2. Program Descriptions. Paragraphs 4.2.1 through 4.2.n shall provide descriptions of the functions (related to paragraph 2.2 of the subsystem specification) of the computer programs in the subsystem.

4.2.1 Program Description. Included in this paragraph will be a description of the inputs, outputs, and data used. Each description will include the information below, as applicable.

4.2.1.1. Inputs. Each input will be described as follows:

a. Title and tag.

b. Format and acceptable range of value of each data element.

c. Number of items (elements).

d. Means of entry and input initiation procedures; e.g., typewriter, card, tape, sensor, internal.

e. Expected volume and frequency, including special handling (such as queueing and priority handling) for high density periods.

f. Priority, e.g., routine, emergency.

g. Sources, form at source, and disposition of source document.

h. Security classification of input and individual items.

i. Requirements for timeliness.

4.2.1.2. Outputs. Each output will be described as follows:

a. Title and tag.

b. Format to include headings, line spacing, arrangement, totals, etc.

c. Number of items.

d. Preprinted form requirements.

e. Means of display, e.g., CRT, printer, typewriter, projector, alarm type, internal.

f. Expected volume and frequency including special handling (such as queueing and priority handling) for high density periods.

g. Priority; e.g., routine, emergency.

h. Timing requirements; e.g., response time.

i. Requirements for accuracy.

j. User recipients and use of displays, such as notification, trends, or briefings.

k. Security classification of output and individual items.

4.2.1.3. Data Base. Each data file, table, dictionary, or directory will be described as follows:

a. Title and tag.

b. Description of content.

c. Number of records or entries.

d. Storage, to include type of storage, amount of storage and, if know, beginning and ending addresses.

e. Classification.

f. Data retention.

4.2.2. Program Description. This paragraph (with subsequent paragraphs if required) shall describe the second computer program in the subsystem using the same organization as shown in paragraph 4.2.1.

APPENDIX E

CODES AND TABLES

APPENDIX E.1.

VALID SECURITY CLASSIFICATION ENTRIES

Three types of security information are defined within the data dictionary:

1. ENTRY-SECURITY which defines the classification of data dictionary entries.

2. DATA-SECURITY which defines the classification of data which the dictionary entry defines.

3. OUTPUT-SECURITY which defines the classification of data when presented in the form of an output report.

Valid classification codes are:

1. UNCLASSIFIED.

2. OUO. (Official Use Only)

3. CONFIDENTIAL.

4. SECRET.

5. TOP-SECRET.

For each type of security defined above the classification codes are prefixed by ENTRY, DATA, or OUTPUT.

APPENDIX E.2.

NIMIS PROJECT STATUS CODES

Each NIMIS project progresses through a series of status conditions during its life. The codes below define these conditions:

1. A - Open Project
2. B - Rescheduled, Open Project
3. C - Complete Project
4. F - Cancelled Project
5. G - Continuing Project (No defined due date)
6. X - Inactive Project (Dates suspended and no effort being applied.)
7. Z - Overhead Project

APPENDIX E.3.

NIMIS PROJECT PRIORITY CODES

Each NIMIS project must compete for resources. To assist in the prioritization and assignment of available resources, each NIMIS project is assigned a relative priority level. This priority is assigned by the user and the DBA staff following determination of the relative importance of the project.

Valid priority codes are:

1. A - The project is of utmost urgency. The successful support of the primary mission of the organization depends on the results of the project. This priority code may be assigned only upon direction of COMNAVINTCOM.

2. B - The project is of great urgency to the requesting organization. It takes precedence over all other projects requested by that organization or currently being performed for that organization. If other projects of the same priority level are in progress, they are placed behind this, the latest project of the same priority level.

3. C - The project has above average priority within the requesting organization.

4. D - The project has average priority within the requesting organization.

5. E - The project has a lower priority than average

within the requesting organization. It may be deferred without serious impact to the requesting organization's goals if resources are not available.

6. F - The project is to be performed on an as-time-allows basis. No delivery date is specified.

APPENDIX E.4.

NIMIS SECURITY CLASSIFICATION CODES

The security classification codes defined in this appendix identify the classification of the entries to be stored in the data base.

Valid codes are:

1. U - Unclassified
2. O - Official Use Only
3. C - Confidential
4. S - Secret
5. T - Top Secret

APPENDIX E.5.

NIMIS PROJECT WORK TYPE CODES

Each NIMIS project performs a particular type of service. The codes described below assist users in preparing reports based on the types of service being performed.

Valid codes are:

1. A - Overhead, Support Administration.
2. B - Briefing Preparation and Presentation.
3. C - Conversion of Programs to IBM 360 OS.
4. D - Program Development.
5. E - Minor modification of Existing ADP Systems.
6. F - Major Modification of Existing ADP Systems.
7. G - Graphics/photo Support.
8. H - Data Base Design.
9. M - Operation System Discrepancy.
10. Q - Query/Ad hoc Request.
11. R - Research/Report/Study.
12. S - Technical Support to Users.
13. T - Non-project Miscellaneous Support.
14. V - Visit.
15. Y - Travel.
16. Z - Miscellaneous Overhead.

APPENDIX E.6.

NIMIS PROJECT COMMENT TYPE CODES

The comment type code provided for use with NIMIS comment entries is mandatory for NIPSSA use only. Other organizations may assign their own meanings to the codes if so desired.

NIPSSA assignments are intended to provide a hierarchy of importance to the codes so that reports may be produced based on various importance levels.

Assigned codes are:

1. 0 - Initial project definition information.
2. 1 - Modifications to original project definition.
3. 2 - Action taken on the project.
4. 5 - Monthly Acitivity Summary Report.
5. 6 - Weekly Activity Summary Report.
6. 8 - Detail Comments by Personnel.

APPENDIX E.7

DATA ELEMENT STANDARDS

A number of sources of data element standards are available. These may be divided into descriptive standards and content standards. Descriptive standards are those which are primarily concerned with the definition of the data element and the manner in which it is used. The format used for internal representation of the data may also be defined. Content standards add an additional dimension to descriptive standards. These standards define the legal or valid contents of standard elements, in addition to the structural definition. An example of descriptive standards is dates. The internal representation identifies the element as six characters, in format YYMMDD. This standard definition may become a content standard by adding validity information: month value range 1 through 12; day value range 1 through 31 with conditions based on month.

Every data element defined for the integrated data base is subject to existing standards. Table E.7.1 identifies those existing standards which apply to data base design. Applicable standards are identified as part of the data element definition process. Where more than one standard apply, all are indicated. If a conflict between standards exists, the conflict must be clearly documented in the element comments

section. The conflict will be resolved by the DBA staff.

The DBA staff has established internal standards of representation for a number of data element types. Table E.7.2 identifies those elements and element types which are subject to these internal standards. The primary reason such standards were adopted is to assure uniform representation of similar data elements internally. This simplifies query processing and program decision logic.

TABLE E.7.1

APPLICABLE ADP STANDARDS FOR DATA ELEMENTS

1. Federal Information Processing Standards (FIPS)

2. Defense Intelligence Agency Authorized Data Elements and Related Features SO-730-110-76, 1 July 1976

TABLE E.7.2

INTEGRATED DATA BASE INTERNAL ELEMENT STANDARDS

Element	Format	Description
Frequency	9(15)V999	This field is primarily associated with radio/radar frequencies. The measurement is in hertz (Hz) to three decimal places, i.e., milli-Hz
Time (seonds)	9(7)V999	The measure is in seconds, to three decimal places, MS.
Time (hrs,min,sec)	9999 99 99V999	Hours measured using the 24-hr clock, minutes 0-59, and seconds to milliseconds.

APPENDIX E.8.

COMPUTER LANGUAGES.

Computer programs written in support of the data base will be produced in a number of languages depending on the best and most efficient approach. Language use may be limited by convention or regulation. The languages which may be defined are:

1. COBOL. Common Business Oriented Language. This is the most commonly used language for data base support and is the standard NIPSSA data base development language.

2. *FORTRAN. Formula Translator.* This language is often used for scientific and statistical programming.

3. CULPRIT. This is a report and query package which supports the data base system. While not truly a language, it is defined as one as far as the data dictionary is concerned.

4. ALC. IBM 360/370 assembly language. This language is used primarily to develop difficult programs which cannot be developed using other available languages.

APPENDIX E.9

STANDARD OUTPUT FORMS

Reports may be produced on a variety of printed output forms. Special or preprinted forms should be avoided as turnaround (response to request) time is longer for special forms processing.

The basic form available is used to produce a large percentage of the data base reports. This form is 11 inches deep by 14 inches wide. It is printed with guide lines spaced every two lines. Several variances of this form are available:

1. Multiple part (carbon) forms in 2, 3, and 4 parts.
2. Reproduction quality single part, unlined.

Additional forms which may be requested are:

1. 8 1/2 inches wide by 11 inches deep, lined or unlined.
2. 14 inches wide by 8 1/2 inches deep, unlined (limited quantity, special order paper).

Users desiring other than standard 11 inch high by 14 inch wide, single part, lined forms should plan to supply the volume of paper required to process their reports.

APPENDIX E.10

PRIVACY LOCKS FOR DATA BASE SUBSCHEMAS

The subschema is the window into the data base. Each application may have one or more subschemas defined to support users of the application. It is possible to limit the access to certain data base record types within an application by applying privacy locks. These locks prevent unauthorized data base activity against the selected record types.

Privacy locks may be defined for:

1. STORE. This lock will prevent occurrences of the specified record from being stored in the data base.

2. MODIFY. This lock prevents occurrences of the specified record, resident in the data base, from being modified.

3. ERASE. This lock prevents deletion of occurrences of the specified record type from the data base.

4. FIND. This lock prevents access to the specified record type. NOTE: This lock is normally used only when it is necessary to include a record type in the subschema for control purposes only.

5. GET. This lock permits the specified record type to be accessed and currency estalished but prevents the user from viewing the contents of the record occurrence.

6. CONNECT. This lock prevents associating the designated record type with any other record occurrence.

7. DISCONNECT. This lock prevents the selected record type from breaking its association with other record occurrences.

When it is desired to set any of the above locks, the clause PRIVACY LOCK FOR xxxxx IS 'NO' is entered as part of the subschema record definition.

APPENDIX E.11

SOURCE PROGRAM LIBRARY BOOKS

A series of COBOL source program library 'books' have been defined to speed preparation of programs to process input data. The use of these books reduces the volume of programming which must be prepared and eliminates errors in processing logic by providing consistent, reusable source statements.

Two groups of source books are provided:

1. Data element validation. These books provide common code for validating contents of data elements processed by an input processing (IP) module.

2. Special control routines. These books provide common code for error status tests, function decisions, and other common processes.

Data Element Validation Source Books

Source books for data element validation require replacement of symbolic names to tailor the code to the data element being validated. All source books in this group require replacement of these symbolic names:

1. F-NO Numeric literal positional number of IP field being processed
2. F-FLD Data name of input data field being

processed

3. R-FLD Data name of data base record field into which validated data will be loaded.

Those source books which provide data value range testing require replacement of these additional symbolic names:

1. LO Literal value of the lowest value range. If alphanumeric, literal must be enclosed in quotes and be equal in length to data field being tested.
2. HI Literal value of the highest value range. If alphanumeric, literal must be enclosed in quotes and be equal in length to data field being tested.

Those source books which provide for specific data element value verification (table validation) require replacement of an additional symbolic name:

1. R-NAME Literal value containing the data base record field name enclosed in single quote marks (name is same as R-FLD).

Source books are broken into two categories:

1. Those used primarily to store data in the data base for the first time.
2. Those used to modify data already present in the

data base.

The primary difference between the two categories is in the handling of blank fields and asterisk-delete (an asterisk in the left-most position of a field indicates the user desires to delete the corresponding data base field contents) processing. Store books always move the contents of the input data field to the data base field, even if it is blank. This insures that garbage data, or data from a previous record, is not left in the field and stored. Where the data base field is numeric, it is important to use one of the zero fill books. Asterisk-delete is not permitted in any store books.

Table E.11.1 illustrates each source book used for input data validation. The books are grouped by function so that reference is easier. The meaning of the function options shown are:

1. Op/Req - Data element is either optional (not essential to complete IP processing) or required (essential to complete IP processing). If a required data element is missing, an "E" level error is generated which rejects the entire input transaction.

2. AN/Num - Data element to be tested is either alphanumeric or numeric. If the element must be numeric and tests as non-numeric, the ZERO FILL IF BLANK books will

attempt to correct the situation. If a non-numeric condition still exists, the data element value is rejected with a "C" level error. Existing data is not modified and zeros are stored in the data base field.

3. * Del - If yes, the data base field may be cleared to blanks or zeros, as appropriate, when an asterisk is punched in the left-most position of the corresponding input data field.

4. Range/Date - Where indicated, range tests for inclusive low and high values. Ranges for AN books must be shown as alphanumeric literals of the same length as the input field being tested. DATE tests year for 00 - 99 inclusive. A numeric test is also performed on dates. Dates must be six digits using format YYMMDD. Month and day combination ranges are also tested.

5. Tbl Val - Where indicated, the source book loads the literal value of R-NAME (which is identical to the name replacing R-FLD except in quotes) and the IP data value into a work area and issues a FIND to the table validation area of the data base. This feature eliminates embedded literal values, subject to later change, from the source code.

6. Zero Fill - Where indicated, source book code tests input data field data for numeric status and inserts

zeros where spaces are found. A second numeric test determines if zero filling satisfied numeric status.

Table E.11.1

Book Ident	Funct	Opt/ Req	AN/ Num	* Del	Range/ Date	Tbl Val	Comments
SSZA	STORE	REQ	AN	NO			
SMZB	MODIFY	REQ	AN	NO			
SSZG	STORE	OPT	AN	NO			
SMZB	MODIFY	OPT	AN	NO			
SMZA	MODIFY	OPT	AN	YES			
SSZC	STORE	REQ	AN	NO	RANGE		
SMZF	MODIFY	REQ	AN	NO	RANGE		
SSZQ	STORE	OPT	AN	NO	RANGE		
SMZF	MODIFY	OPT	AN	NO	RANGE		
SMZE	MODIFY	OPT	AN	YES	RANGE		
SSZI	STORE	REQ	AN	NO		YES	
SMZI	MODIFY	REQ	AN	NO		YES	
SSZJ	STORE	OPT	AN	NO		YES	
SMZI	MODIFY	OPT	AN	NO		YES	
SMZH	MODIFY	OPT	AN	YES		YES	
SSZB	STORE	REQ	NUM	NO			
SMZV	MODIFY	REQ	NUM	NO			
SSZH	STORE	OPT	NUM	NO			
SMZV	MODIFY	OPT	NUM	NO			
SMZW	MODIFY	OPT	NUM	YES			
SSZN	STORE	REQ	NUM	NO			ZERO FILL IF BLANK
SMZM	MODIFY	REQ	NUM	NO			ZERO FILL IF BLANK
SSZM	STORE	OPT	NUM	NO			ZERO FILL IF BLANK
SMZM	MODIFY	OPT	NUM	NO			ZERO FILL IF BLANK
SMZO	MODIFY	OPT	NUM	YES			ZERO FILL IF BLANK
SSZP	STORE	REQ	NUM	NO	RANGE		
SMZA	MODIFY	REQ	NUM	NO	RANGE		
SSZO	STORE	OPT	NUM	NO	RANGE		
SMZU	MODIFY	0PT	NUM	NO	RANGE		
SMZY	MODIFY	OPT	NUM	YES	RANGE		
SSZF	STORE	REQ	NUM	NO	DATE		
SMZC	MODIFY	REQ	NUM	NO	DATE		
SSZE	STORE	OPT	NUM	NO	DATE		
SMZC	MODIFY	OPT	NUM	NO	DATE		
SMZX	MODIFY	OPT	NUM	YES	DATE		

Special Routines. Special source statement books are provided to execute logic which repeats frequently in input processing programs. These routines are broken into three groups:

1. Option verification and distribution. These routines test the processing option code in the input data and perform the designated routine(s).

2. IDMS status tests. These tests check the error status field after IDMS executions and set the error level.

3. Other tests.

Option Verification and Distribution Books.

1. SSZD - Store/modify/delete. The book loads UE-MOD (the IP module identifier field), initializes UE-LEVEL/UE-MAX (error status hold fields) and performs appropriate sub-section or issues an error message at "E" level if the proper option is not found. Replacement of the following symbolic names is required:

a. FID Literal value of UE-MOD. Use IP identifier enclosed in single quote marks.

b. F-OPT Name of the IP area option field, usually (xxx-OPTION).

c. F-ST Name of procedure SECTION for store, usually xxx-STORE.

d. F-MOD Name of procedure SECTION for modify, usually xxx-MODIFY.

e. F-DEL Name of procedure SECTION for delete, usually xxx-DELETE.

2. SMZL - Modify only. The book loads UE-MOD, initializes UE-LEVEL, and UE-MAX and tests for option "M". An "E" level error message is issued if the option is not modify. Replacement of the following symbolic name is required:

a. FID Literal value of UE-MOD. The IP identifier is entered, enclosed in single quote marks.

3. SMZR - Insert/remove. The book loads UE-MOD, initializes UE-LEVEL/UE-MAX, tests the option for "I" or "R", and performs appropriate sub-section or issues an "E" level error message. Replacement of the following symbolic names is required:

a. FID Literal value of UE-MOD. The IP identifier is entered, enclosed in single quote marks.

b. F-INS Name of procedure SECTION for insert, normally xxx-INSERT.

c. F-REM Name of procedure SECTION for remove, normally xxx-REMOVE.

input data field being processed.

b. F-FLD Name of input data field being processed.

APPENDIX E.12

IP FUNCTION OPTIONS

Batch input processing (IP) data contains a data item identifier and a functional option code. This code, the fourth position of the data item format, determines what processing will be performed using the data on the input item.

Five basic input processing function codes are permitted:

1. S - Store a new record occurrence in the data base.

2. M - Modify the contents of an existing record occurrence in the data base.

3. D - Delete an existing record occurrence from the data base.

4. I - Insert an existing data base record occurrence into a set, establishing a relationship with another record occurrence.

5. R - Remove an existing data base record occurrence from its association with another record occurrence.