Taylor & Francis
Taylor & Francis Group

EDPACS: The EDP Audit, Control, and Security Newsletter

ISSN: 0736-6981 (Print) 1936-1009 (Online) Journal homepage: www.tandfonline.com/journals/uedp20

System & Audit Aspects of the Data Dictionary

Donald L. Adams

To cite this article: Donald L. Adams (1976) System & Audit Aspects of the Data Dictionary, EDPACS: The EDP Audit, Control, and Security Newsletter, 3:11, 1-14, DOI: 10.1080/07366987609450193

To link to this article: https://doi.org/10.1080/07366987609450193

Published online: 05 Jan 2010.

Volume III, No. 11 May 1976

edpacs®

The EDP Audit, Control and Security Newsletter

System & Audit Aspects of The Data Dictionary

(NOTE: This is an expanded version of a presentation given at the Sixth Conference on Computer Audit, Control and Security on March 29, 1976, in San Francisco.)

THE DATA DICTIONARY DEFINED

The data dictionary is a documentation tool designed to provide a standard definition for all data elements (fields), segments (records), and data bases (files). It contains narrative and technical descriptions of the data as well as information about security, edit considerations, structure, and application usage.

Normally, a data dictionary is considered to be a documentation aid used to support a data base system. However, one of the earliest data dictionary projects was undertaken by IBM in 1969 to support an internal study of data flows within their Data Processing Group. The intent was to design a uniform representation for data structure and content and at the same time to reduce the cost and confusion caused by data redundancy. IBM's early involvement in developing a data dictionary stood them in good stead as they began to implement data base applications in the early 1970's.

Thus, the data dictionary can be considered a basic tool for the management of data. It can be particularly useful when data is to be shared by a number of different applications or systems. In such systems, one consistent definition of data can serve the needs of all users. While filling this function, the data dictionary will help reduce the errors associated with the misinterpretation of data definitions; e.g., what is meant by "date".

ADVANTAGES

The data dictionary provides a number of concrete advantages:

1. Simple, effective control over all data definitions. Since each element is only defined once, control is much easier.
2. Tracing the path of data through the system is simplified. Each unique element can be directly referenced to all programs that create, use, update, or delete that particular element. This referencing also includes all files that contain the data element.
3. Standard edit routines can be developed and related, as required, to individual data elements. In this way programmers no

Publisher: Harold Weiss Editor: Donald L. Adams
Production Editor: Sylvia M. Geller

longer have to develop their own edit coding. The use of standard validation routines will help reduce the overall error rate within the system.

4. Whenever a data element must be modified, it will be easier to evaluate the impact of the change on all aspects of the total system. At the same time, making the change within the one data element will automatically mean the same change has been effected in regard to all programs or files that use that data element.

5. Since all data is defined on a system-wide basis, individual programmers and analysts do not have to devote time to designing and implementing file designs. This will serve to reduce greatly the time required to develop new applications.

6. Establishing a data dictionary is the initial step in the design and implementation of a data base system. Any organization that has any plans to move to a data base management system (DBMS) should give serious consideration to investing the time and effort required to implement a data dictionary in support of the existing systems. This initial investment will be more than justified on the basis of time it will save in developing the design of the first data base.

PROBLEMS

As a partial offset to the many advantages of the data dictionary, there are some disadvantages involved:

1. It will be a time-consuming and complex effort to develop the data dictionary. In a medium-scale installation (e.g., IBM S/370, Model 145) with a number of existing major applications, the first cut at a data dictionary could easily take one to two man-years. From the cost standpoint, such a project is a significant undertaking.

2. Once it has been installed, it will take a considerable amount of effort to maintain the data dictionary. In a typical installation two or three people may have to be assigned on a full-time basis to the care and feeding of the data dictionary.

3. Once the data dictionary is operational, access to existing files and the setting up of new ones will have to be on a controlled, formal basis. Many programmers and analysts may resent this formality. They are likely to feel that being kept away from the design of files cramps their artistic style. The effect on morale must influence any decision to implement a data dictionary.

So, the data dictionary project has its positive and negative aspects. However, on an overall basis, the good points will outweigh the bad for most data processing installations. The implementation of a data dictionary deserves serious consideration by any EDP operation.

PLANNING FOR A DATA DICTIONARY

A number of factors must receive special consideration during the planning phase of a data dictionary implementation project. They include:

Naming Conventions. Standard names should be assigned to each data element. Usually, two names will be involved. One will be the name used to refer to the data element within the data processing system (e.g., a COBOL data name). The other name is a fully descriptive name used for documentation purposes. (It is not always possible to be fully descriptive within the 30 character maximum name allowed by COBOL.) For example:

- Data Name - "VENDR-LAST-PRICE"
- Documentation Name - "Last Price Charged by a Vendor".

Numbering Conventions. For ease of processing, it is often necessary to refer to a program, file, or data element by use of a reference number. This can be particularly useful in building cross-reference listings (e.g., programs that access a particular data element). Some typical numbering conventions might be:

- Program Number - INP391. The first two characters identify a system (IN = inventory), the third character (P) indicates a program, and the last three characters (391) identify the specific program.
- File Number - PSD293. The first two characters again identify a system (PS = parts), the third character shows what kind of file is involved (D = disc, T = tape, C = card, R = printed report, V = visual display), and the last three characters (293) identify a specific file.
- Data Element Number - PS300030. The first two characters identify the data

base (PS = parts data base); the next two indicate segment (00 = root, 30 = third segment within the hierarchy; the fifth character indicates hierarchy level (0 = highest, 9 = lowest); and the last three establish the data element within the segment. In both the segment and data element numbers, the zero in the last character provides room for insertion in the event of future modifications.

Use of a Table-Driven Editor. Since the data dictionary will provide for the non-redundant definition of each data element, its installation presents a good opportunity to implement an edit monitor concept. Rather than have each application program perform its own edit on all the data it processes, all input will pass through a standard edit facility (the edit monitor). Once it has passed the standard edit, data will be recorded on the appropriate file or data base. (See Figure 1.) From then on, any application that handles the data can assume the data is correct and will not be required to perform any edits.

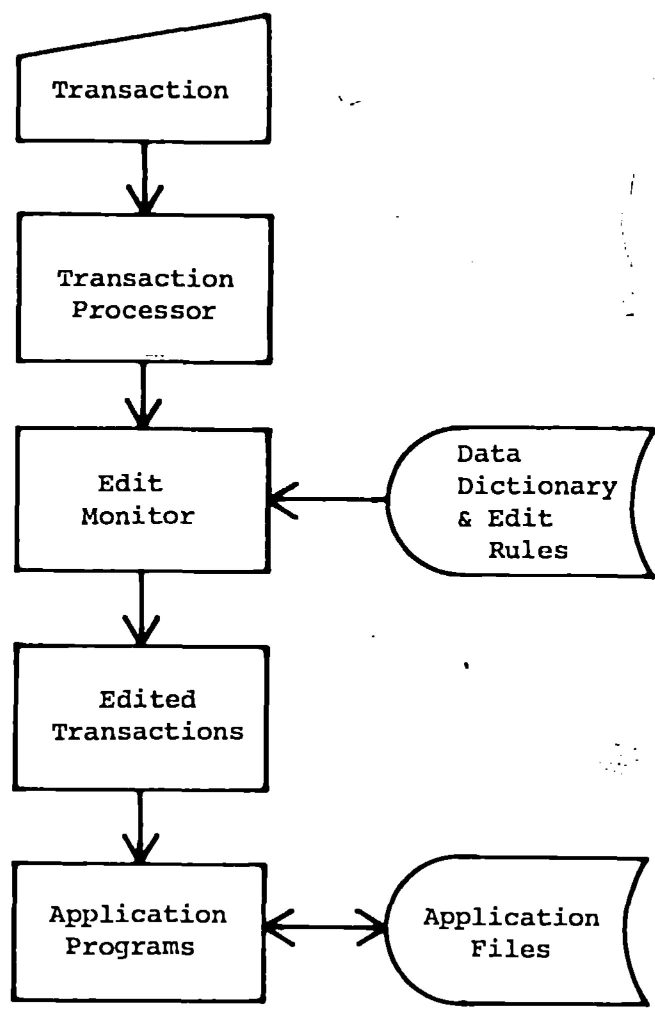
To implement this approach, standard edit requirements must be established for each data element. A table of these requirements is then stored within the computer system. When a transaction enters the system, the edit monitor will identify transaction type, determine what data elements will be present, access the edit requirements table, and apply the appropriate edits.

This approach has three basic advantages:

- All data will be subject to uniform edits.
- Since only one set of edit tests is applied to each data element, it is much easier to modify the tests in the light of changed requirements.
- Application programmers do not have to code any edit processing.

Content of the Data Dictionary. Fairly early in the design effort, the planners must develop a preliminary idea of the data dictionary's content. As work continues, modifications to the planned content will probably take place, but a preliminary design provides a starting point. Later in this article the contents of a fairly complete data dictionary will be presented and discussed. This illustration will serve as a good basis for creating a preliminary design of data dictionary content.

Reports to be Produced. Another design factor that must be considered is the reports that will be produced by the data dictionary system. Requirements will vary from installation to installation. This article will present some of the types of reports that should be considered.



STANDARD EDIT FACILITY

Figure 1

IMPLEMENTATION STEPS

While it is difficult to set up a standard approach that is applicable to any systems effort, the following steps might be considered fairly representative:

1. Obtain or prepare copies of the layout of all files. In this instance, printed reports should be considered to be files.
2. Verify the accuracy of these layouts. This involves comparison with file dumps.
3. Develop a definition of the purpose of each file and the content of each data element (field). It may be easier to start with the output and work back to the input.
4. Develop the required standard names for each data element.
5. Develop a numbering convention for programs, files, and data elements.
6. Implement standard data names in all existing programs and files. The global replace function provided in many librarian systems can prove quite helpful in accomplishing this step. (See EDPACS, July 1973, page 4 and February 1974, page 7).
7. Develop a standard format for all file definitions.
8. Redefine all files in accordance with the standard format.
9. Store the new file formats in a system library so they can be copied directly into all newly developed programs.
10. Establish an individual (or group) who will have the sole authority to set up or define new files or data elements.
11. Inform all programmers and analysts that they are no longer allowed to create new files or data elements. Further, they are not allowed to modify any files or data elements that have been defined.
12. To the maximum possible extent, eliminate redundant files in existing systems.
13. Develop preliminary edit requirements for each data element.
14. Implement the centralized validation and updating concept (edit monitor) for all new programs, files, and applications.
15. Determine who among the users should be assigned ownership of each individual data element.
16. Review with the owner the preliminary edit requirements for each data element. Make revisions as needed.
17. To the extent practical, implement the edit monitor concept for existing applications.
18. Design the final format of the data dictionary.
19. Determine the output reports to be produced by the data dictionary system.
20. Implement the computer and clerical processing required to perform all data dictionary functions.
21. Prepare and distribute the formal data dictionary documentation.

A TYPICAL DATA DICTIONARY

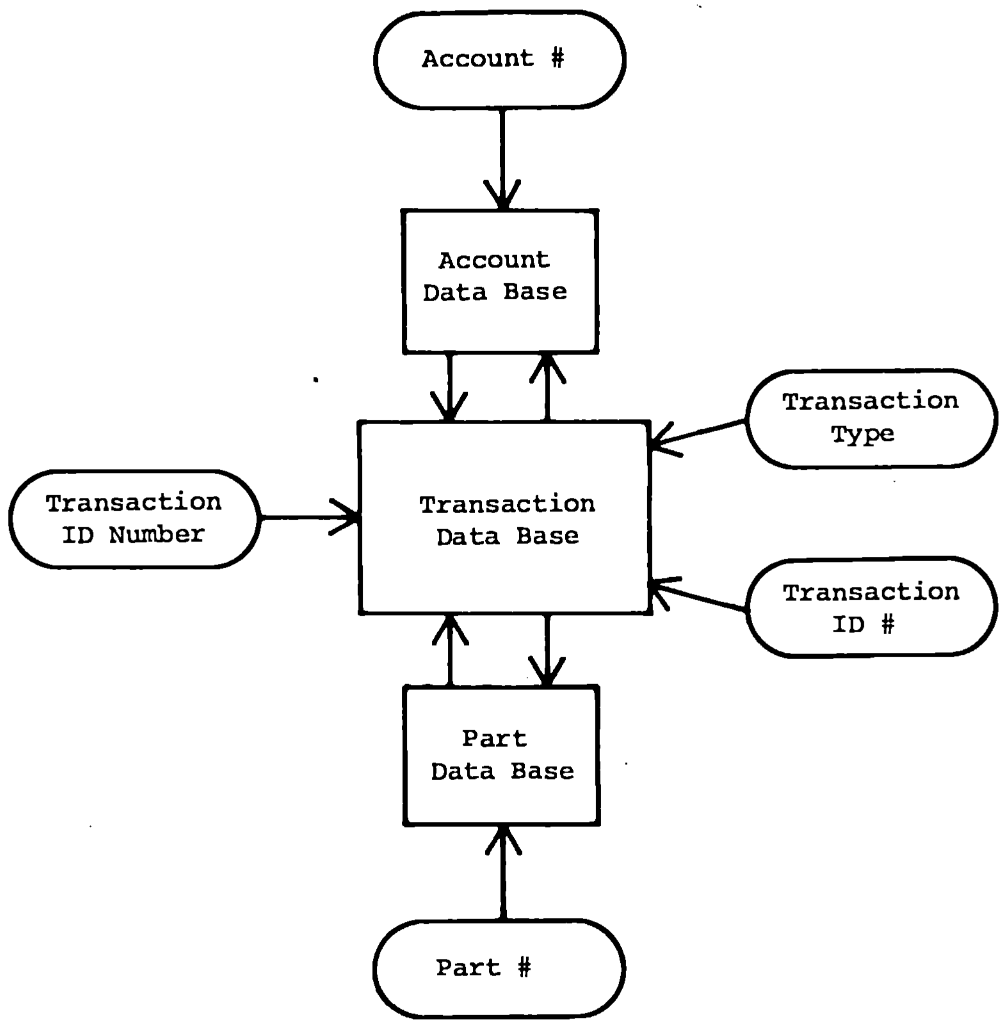
While there is probably no such thing as a truly "typical" data dictionary, it should prove helpful to review in some detail the elements of a comprehensive dictionary system. In this example the data dictionary supports a data base management system. However, with minor modifications, this same type of information could be developed to document a non-data-base system.

Hierarchy Diagrams. Two types of hierarchy diagrams are involved. On the highest level is a diagram that illustrates the entrances into and relationships between the various data bases within the system. (See Figure 2.) At the second level of detail is a hierarchy diagram that shows the structure and segment network within each data base. (See Figure 3.) On these diagrams a segment that is marked "R" is a repeating segment. That means it may be present one or more times as required by the design of the system. These hierarchy diagrams are manually prepared and maintained.

Segment Libraries. Assuming the system is using COBOL, a standard Data Division entry would be developed for each segment (see Figure 4) and stored in a system library. For other programming languages, an equivalent capability is usually available. As needed, these library entries can be copied into an application program that will use the particular segment or segments. Besides ensuring uniformity and thus reducing errors, this approach reduces the amount of effort required to code an application program.

Detailed Documentation. In support of each segment and its individual data elements, the following information would be provided (all items may not be required in all cases):

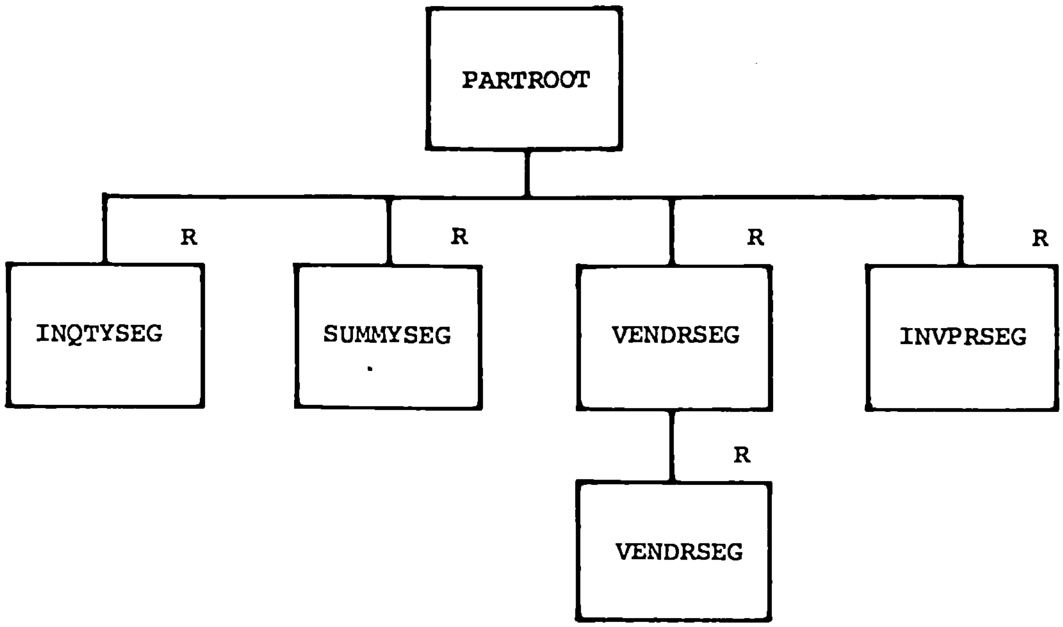
1. Name of the data base



A DATA BASE SYSTEM

Figure 2

2. Name of the segment (COBOL name)
3. Date prepared
4. Data element name (COBOL name)
5. Segment or data element identification number
6. Fully descriptive name
7. Description of the contents of the segment or data element. This description is as detailed as possible.
8. Formula or other source for the derivation of data within this element
9. Physical attributes of the contents of the element. In a COBOL environment, the PICTURE clause would serve to provide the required information.
10. Specific value, values, or expected range of values associated with this element
11. Edit tests that can be applied to this data element. For each test, a description of how exceptions are to be handled should also be given.
12. Owner of the data element or segment. Normally, this will be a specific user or user department.
13. Security precautions or the level of security associated with a



DATA BASE HIERARCHY DIAGRAM

Figure 3

```
01   VENDRSEG.
     05   VENDR-ID-CODE                    PIC 9(8).
     05   VENDR-NAME                       PIC X(30).
     05   VENDR-LAST-PRICE                 PIC 9(4)V9(2).
     05   VENDR-LAST-QTY-ORDERED           PIC 9(6).
     05   VENDR-LAST-ORDER-DATE.
          10  VENDR-LAST-ORDER-YR          PIC 9(2).
          10  VENDR-LAST-ORDER-MONTH       PIC 9(2).
          10  VENDR-LAST-ORDER-DAY         PIC 9(2).
     05   VENDR-LAST-ORDER-PO-NO           PIC X(8).
     05   FILLER                           PIC X(8).
```

DATA DICTIONARY SEGMENT DEFINITION

Figure 4

segment or data element
14. Standard report heading to be used whenever this data element is printed on an output report and the format used to print the data element. Once again, in a COBOL environment, the PICTURE clause would probably be used for the latter.
15. Output reports or files that contain this data element
16. Programs that produce the outputs listed in item 15
17. Input documents, files, or other sources that are used to create this data element
18. Programs that process the inputs listed in item 17
19. COBOL name of parent segment (if any)
20. COBOL name of dependent segment(s), often called "children" (if any)
21. If a segment is being documented, a list of the COBOL names of its data elements should be provided.

(See Figures 5 and 6 for samples of documentation.)

Because of the volume of information contained within the data dictionary (500-1,000 pages is not unusual), the preparation of this documentation is usually automated in some way. Some of the currently available data dictionary software packages will be briefly reviewed in this article. While on the subject of the size of the data dictionary, some statistics from a dictionary project within Xerox are of interest:

- Data elements defined - 3,000
- Estimated additional elements to be defined - 7,000

Data Base: PART

Segment: VENDRSEG

Element: n/a

ID Code: PS30000

Name: Vendor Information Segment

Description: This segment contains information about vendors who are possible sources of supply for a particular part. If a part is not purchased from the outside (e.g., internally manufactured), this segment will not be present. When it is present, this may be a repeating segment.

Parent Segment: PARTROOT

Dependent Segment(s): VADDRSEG

Data Elements: VENDR-ID-CODE
VENDR-NAME
VENDR-LAST-PRICE
VENDR-LAST-QTY-ORDERED
VENDR-LAST-ORDER-DATE
VENDR-LAST-ORDER-PO-NO

DATA DICTIONARY - DETAILED DOCUMENTATION (SEGMENT)

Figure 5

Data Base: PART

Segment: VENDRSEG

Element: VENDR-LAST-PRICE

ID Code: PS300030

Name: Last Price Charged by a Vendor

Description: This element contains the most recent price a particular vendor charged, per unit of measure, for a specific part.

Physical Attributes: PICTURE 9(4)V9(2)

Values: Range from 0000.01 to 9999.99 - positive values only

Edit Considerations: The element should always be all numeric. In actual practice, no known value exceeds 2000.00, but it is not likely that this limit would ever be exceeded. An updated value placed in this element should not vary from the prior value by more than 10%. This element should not vary by more than 20% from the value of MASTER-UNIT-PRICE in the PARTROOT. A transaction with a non-numeric value in this element should be rejected. All other transactions that fail to meet the edit criteria should be held on an edit suspense file, reported on an edit exception report, and processed in accordance with instructions issued by someone who knows the proper supervisory override password.

Owner: Authorized users in the Purchasing Department

Security: Access restricted to authorized users in Accounting, Auditing, and Purchasing. This element is considered to be company confidential.

Standard Report Heading and Output Format: bbVENDORbb
LAST PRICE
bZ,ZZ9.99b

Outputs:	Report/File	INR237	INF393	INF407	INF632
	Program	INP391	INP391	INP277	INP892

Parent Segment: PARTROOT

Dependent Segment: VADDRSEG

DATA DICTIONARY - DETAILED DOCUMENTATION (DATA ELEMENT)

Figure 6

- Physical size of dictionary - 24,000,000 characters
- Estimated size of "final" dictionary - 56,000,000 characters.

Indexing. To provide easy access to the information within the data dictionary, three different indexes are provided:

- Hierarchical index - In the same sequence as the hierarchy within each data base.
- Alphabetic Index - based on the alphabetic sequence of the COBOL data names assigned to each segment and data element. In some cases, two alphabetic indexes may be given — one based on the COBOL data name and the other on the fully descriptive name.
- KWIC Index - KWIC stands for key-word-in-context. Each word in the COBOL data name is extracted, and a full index of the individual words in alphabetic sequence is prepared. The data name from which each word was extracted is shown next to that word to indicate the "context" in which the word was used.

A particular installation might develop additional index requirements to provide for a special need within a given system. However, the three indexes outlined above will usually prove to be quite workable. (See Figure 7 for examples of indexes.)

	Page
HIERARCHICAL INDEX	
PART Data Base (continued)	
VENDRSEG	62
VENDR-ID-CODE	63
VENDR-NAME	64
VENDR-LAST-PRICE	65
VENDR-LAST-QTY-ORDERED	66
VENDR-LAST-ORDER-DATE	67
VENDR-LAST-ORDER-YR	68
VENDR-LAST-ORDER-MONTH	69
VENDR-LAST-ORDER-DAY	70
VENDR-LAST-ORDER-PO-NO	71
ALPHABETIC INDEX	
VENDR-ID-CODE	63
VENDR-LAST-ORDER-DATE	67
VENDR-LAST-ORDER-DAY	70
VENDR-LAST-ORDER-MONTH	69
VENDR-LAST-ORDER-PO-NO	71
VENDR-LAST-ORDER-YR	68
VENDR-LAST-PRICE	65
VENDR-LAST-QTY-ORDERED	66
VENDR-NAME	64
VENDRSEG	62
KWIC INDEX	
ORDER - (VENDR-LAST-ORDER-DATE)	67
ORDER - (VENDR-LAST-ORDER-DAY)	70
ORDER - (VENDR-LAST-ORDER-MONTH)	69
ORDER - (VENDR-LAST-ORDER-PO-NO)	71
ORDER - (VENDR-LAST-ORDER-YR)	68
ORDERED - (VENDR-LAST-QTY-ORDERED)	66

DATA DICTIONARY INDEXES

Figure 7

DATA DICTIONARY MAINTENANCE

Like any other system, the data dictionary processing must be subject to maintenance. Since it is a documentation tool, the data dictionary is useful only if it is kept current. Several aspects of the maintenance of a data dictionary involve some special considerations.

Updates. Obviously, the data dictionary must be updated to reflect changes in the data base or files. However, authorization for such changes must come from one central source. Quite often this source will be the data base manager. One of his many duties is to resist change. He will often adopt an approach of conservative skepticism in regard to all proposed changes. **(Note: When I was a data base administrator, I made it a standard practice to reject the first submission of any proposed change.)** Since it is impossible to avoid updates, every step must be taken to reduce their impact. One of the major considerations, if not the major consideration, is that of timing.

Update Timing. When the data dictionary has been implemented, all applications will be driven by the same set of data elements. As a result, any change is likely to have a more pervasive impact. Therefore, everyone affected by a change must receive reasonable advance notice of that change. The data dictionary facilitates such notification. Programmers who are responsible for any program that uses a particular data element should receive at least two weeks advance notice of a proposed change.

To the maximum possible extent, data dictionary revisions should be implemented on a regular schedule. A once-a-month schedule will be adequate in most cases. The physical change to the data base should not be effected until the data dictionary has been revised and the update distributed. A reasonable sequence of events might be:

- On the first of the month distribute an update to the data dictionary, a summary of the changes, and a special notice to all those whose programs have been affected.
- On the 15th of the month implement the proposed changes to the actual data base.

Documentation. All relevant data dictionary documentation must be completely revised to reflect any modifications to the data base or other related files. This can be a very time-consuming and finicky kind of task, but it is essential. Out-of-date documentation is worse than no documentation at all. For the convenience of all concerned, updates should be distributed as either replacement or additional pages that modify the prior version of the data dictionary.

SECURITY AND CONTROL CONSIDERATIONS

There are several special security and control considerations associated with the use of a data dictionary. They should be of special interest to auditors involved in the review of a dictionary system.

Update Authority. As mentioned earlier, the basic authority for all data dictionary updates belongs to the data base administrator. However, a number of other groups should also be involved. Before an update is processed, and the data dictionary is revised, the following should formally approve the change:

- The user who owns the data element
- The quality control group
- The data base administrator.

It may be useful for the auditor also to sign off on modifications. This is not a formal approval, but serves to indicate that he was informed of the change. As a final step, the owner-user should review the data dictionary revision before it is issued. The auditor should review the implementation of the update controls for compliance with the installation's requirements and standards.

Distribution. Since the data dictionary provides the complete documentation in support of all information contained within the system, its distribution should be carefully controlled. It should should be made available only on a "need-to-know" basis. An individual programmer may have copies of the hierarchy diagrams and indexes, but should be given detailed documentation only for the segments he is authorized to use. There should be almost no distribution of the complete dictionary.

Physical Security. By this time the importance and relatively high cost of the data dictionary should be well established. It is necessary to provide this valuable asset with a high level of physical security.

Master copies of the complete data dictionary should be maintained under the exclusive control of the data base administrator or someone else on a similar level of management authority. Ideally, only one hardcopy master and a backup on microfilm should be provided. A duplicate microfilm set should be stored at a secure off-site location. The on-premises master copy should be stored in locked, fire-resistant cabinets. The data base administrator and his superior should be the only ones who have physical access to the hardcopy and the microfilm. In short, the data dictionary should receive maximum security protection.

AUDIT CONSIDERATIONS

Both internal and external auditors will find that the data dictionary is an extremely important aid to their efforts. It may well be that the auditors will be the only ones besides the data base administrator who have access to the complete documentation within the dictionary. Some of the audit uses that might be considered include:

Developing an Understanding of the System. The initial step in any audit procedure is to understand the system. It can be extremely difficult for an auditor to develop a good working knowledge of an EDP application. Normally, the starting point is a review of inputs and outputs. The auditors then trace transactions through the system to test their comprehension of the processing.

Because it is based upon the data elements within a system, the data dictionary seems tailor-made for the auditor's requirements. By providing a comprehensive, organized documentation format, the dictionary gives the auditor much of the basic information he will need to gain an understanding of the system so he can plan his audit approach. The presence of a data dictionary will greatly facilitate the auditor's work.

Review of Security Controls. An important aspect of any audit review is the study and evaluation of security controls. In most systems it is often quite difficult for the auditor to identify the elements of security associated with or applied to the data to be processed. The data dictionary will normally provide a full description of the security requirements that should be enforced in regard to each data element. The auditor can review the security descriptions in the data dictionary, determine the data elements that are particularly sensitive, and concentrate the review effort on those elements. Further, the data dictionary should provide the auditor with the information he will need to plan the detailed steps he will follow in evaluating the effectiveness of the data security for each element to be reviewed.

Review of Edits. An important element of control in a data processing system is the edit tests that are applied to the input. These form the first line of defense against errors and are of particular interest to auditors. In many systems it is extremely difficult for the auditor to determine the exact nature of the edits that have been implemented. To pinpoint the edits within a particular system, the auditor may in extreme cases be forced to review program code. (See EDPACS, August 1975, pp.1-7.)

The data dictionary should provide all the details that govern and describe the edit tests. It will allow the auditor to collect all the basic data he will need to evaluate the effectiveness of the edits within the system. As a result, it will save a considerable amount of audit time and effort.

Developing Compliance Tests. Once the auditor understands the controls, particularly the edits, and has evaluated their apparent effectiveness, he must test his understanding and evaluation. These tests are called compliance tests. In a typical system, developing these tests can be a complex task of considerable aid to the auditor. The data dictionary contains all the information needed to develop a comprehensive set of audit tests.

Utilization of Computer Audit Software. Anyone who has ever used computer audit software knows that the major problem he will encounter is determining the correct layout and content of the data files that are to be processed. This sounds like a trivial task. In theory it is, but in the real world it is extremely difficult. The data dictionary can provide the auditor with most of the information needed to implement a computer audit software application with a minimum amount of time and effort.

SUMMARY

While a data dictionary is usually implemented to support a data base management system, it provides an extremely useful plan of documentation that can be used to provide an understanding of any system, even a manual one. From the standpoint of EDPACS' readers, the dictionary approach offers some very significant advantages:

- Auditors will find the dictionary contains much of the information they need to plan and conduct an audit.
- Control will be facilitated by providing a standard definition for and edit of each data element.
- Security requirements will be formally established and documented.
- Systems design personnel will quickly realize that the preparation of a data dictionary is the first logical step toward the design and implementation of a data base management system.

Since it offers something for almost everyone, EDP installations should give serious consideration to implementing a data dictionary system. While it is not a project to be undertaken lightly, the results will be well worth the effort.

* * * * *

DATA DICTIONARY SOFTWARE PACKAGES

During the last few years a number of data dictionary software systems have been developed and placed on the market. While none of them will probably fill all the requirements of a particular installation, they should be reviewed carefully by any group contemplating the implementation of a directory. Information on each package is briefly given below:

Data Base Directory
Eastern Air Lines Incorporated
International Airport
Miami, Florida 33148
Hardware: IBM 360/370 (DOS or OS)
Specific Data Base Support: TOTAL
Highlights:

1. The data dictionary descriptions can be used to automatically generate data definition language (DDL) statements required to generate, update, and access a TOTAL data base.
2. A system-generated, unique identifying number is assigned to each data element. The data base administrator can use these numbers to detect and eliminate redundant data definitions.
3. Ten basic reports are provided. They include:
   - Overall narrative for each data base
   - Data elements within a data set
   - Data element definitions
   - Program narratives.
4. A number of useful utilities are furnished for use by programmers and the data base administrator; for example:
   - Correct or fix broken file linkages
   - Dump routines
   - Combine and condense log tapes
   - Check linkage paths and control files.
5. A videotape training course is available.

Data Catalogue
Synergetics Corporation
One Garfield Circle
Burlington, Massachusetts 01803
Hardware: IBM 360/370 (DOS, OS, and VS), Univac, Burroughs, Control Data, and Honeywell
Specific Data Base Support: TOTAL, IDMS, IDS, IMS, and System 2000.
Highlights:

1. Powerful cross-reference capabilities.
2. Provides detailed information on data structure and content.
3. Can be used to document forms and reports.
4. Flexible keyword features.
5. Enforces user-defined standards.
6. Generates COBOL File Descriptions.
7. Easy to install (less than one day).

Data Dictionary/Directory
IBM Corporation
(Contact your local office)
Hardware: IBM 360/370 (OS only)
Specific Data Base Support: None
Highlights:

1. Accommodates both narrative and technical data descriptions.
2. The system assigns a unique name to every data element.
3. A number of standard output reports are provided.
4. COBOL source programs can be used to provide input.
5. Over 80 keywords, operators, symbols, and conventions are provided for use in

developing standard descriptions.

6. Preprinted forms are used for input.

7. A wide range of logical relationships can be specified.

Datamanager

Management Systems and Programming Limited
71 Gloucester Place
London W1H 3PF, England

Hardware: IBM 360/370 (DOS or OS)
Specific Data Base Support: ADABAS
Highlights:

1. Full cross-referencing between all elements and segments.

2. Dictionary may be interrogated to produce almost any required form of report.

3. Built-in security checks control all accesses to the data dictionary.

4. COBOL or PL/1 source code can be used to provide input.

5. Dictionary may be used to generate data descriptions in COBOL, PL/1, or BAL formats

6. Can be used to generate test files.

LEXICON

Arthur Andersen & Co.

Hardware: IBM 360/370 (OS or VS)
Specific Data Base Support: IMS, ADABAS
Highlights:

1. Defines all data characteristics and relationships.

2. Generates code for record descriptions in COBOL, PL/1, and BAL.

3. Automatically generates input edit programs.

4. Can be used to produce virtually any kind of output report.

5. Edit processing can be changed without revising application programs.

6. "Data Extractor" and "Report Writer" features can be used to generate output reports to meet application system needs.

7. Generates control statements required to process IMS data bases.

Socrates

Cincom Systems, Inc.
2181 Victory Parkway
Cincinnati, Ohio 45206

Hardware: IBM 360/370 (OS and DOS)
Specific Data Base Support: TOTAL
(also supports non-TOTAL data sets)
Highlights:

1. Can prepare almost any type of documentation report required.

2. Provides a direct interface with TOTAL's internal "DATA-DICTIONARY".

3. Eleven different sets of specific cross-references may be maintained. For example, these may include:
- Programs to data elements
- Data elements to reports
- Programs to users.

4. A variety of other valuable information can be provided:
- Passwords assigned
- Core required by each program
- Program execution frequency.

(Note: Socrates is really a general-purpose report writer designed to work directly with TOTAL. However, the vendor has used it as the basis for creating a data dictionary system.)

Three D

Manager of Systems Development
Anderson, Clayton Foods
P. O. Box 6165
Dallas, Texas 75222

Hardware: IBM 360/370
Specific Data Base Support: IMS
Highlights:

1. Can be used to maintain all libraries required to support IMS.

2. Maintains operating statistics about IMS usage.

3. Provides a wide range of user and systems documentation.

(Note: Package is no longer marketed.)

UCC Ten

University Computing Company
7200 Stemmons Freeway
P. O. Box 47911
Dallas, Texas 75247

Hardware: IBM 360/370 (OS or VS)
Specific Data Base Support: IMS
Highlights:

1. Extensive cross-reference facilities.

2. Provides a central definition for all data elements.

3. A set of user-defined specifications can be used to enforce standards automatically.

4. Definitions supplied to IMS can be controlled since they must be passed through UCC Ten.

5. Test files or data bases may be separately defined.
6. Automatically generates IMS control parameters.

Other Products
Several other data dictionary program products have been announced, but no details about them could be located. They are:
- British Rail DD/D System
  Technical Design Officer
  Computing Division
  British Rail
  Furlong House
  Nottingham NG2 1AL, England
- PRIDE-Logik
  M. Bryce & Associates, Inc.
  P. O. Box 15459
  Cincinnati, Ohio 45215
- COMMANDD
  Software Unlimited
  63-84 Saunders Street
  Rego Park, New York 11374

(Note: Letters sent to the last firm are returned marked "Moved, not forwardable". If any of our readers has a better address, please let us know.)

Librarian Packages
As an alternative to a data dictionary software package, the facilities of a librarian package, coupled with the liberal use of comment statements in a COBOL program, can be used to perform many of the required functions. The user would have to create his own software to prepare cross-reference listings, however.

Donald L. Adams

ABSTRACTS & COMMENTARIES

PLANNING EFFECTIVE COMPUTER SYSTEMS, by Alexander J. Adams. CA Magazine (250 Bloor St. E., Toronto, Ontario, Canada, M4W 1G5). August 1975, pp.42-46.

In 1972 a performance survey of computer installations developed the following information:
- 31% of total computer costs relate to systems development and maintenance.
- 36% of computer staff and 22% of computer time are committed to these same tasks.
- 67-77% of computer time is available for production processing.
- 27-53% of computer time is used for routine accounting and production applications.

Since 1972 the findings of the survey have been subject to continuing confirmation. They seem to be holding up quite well.

These statistics give rise to some interesting questions:
- Should systems devel pment continue to absorb so much of our computer resources?
- Why is so much EDP t me spent on routine applications and why do they not provide more sat sfaction?
- Will things ever get better?

Unfortunately, the answer to the last question seems to be "no".

OPERATIONAL JUDGMENT
Many of the systems failures that have taken place can be traced to a lack of management judgment. Proper operational judgment is based upon taking an overall view of a problem and recognizing it for what it really is. Often, a new system may not be the answer. Perhaps, in the final analysis, enforcing the requirements of the existing system may be all that is needed. Management must do its best to ensure that the systems people solve the right problems.

PLANNING FOR EFFECTIVE SYSTEMS
Plans for developing new systems should be in accordance with overall long-range plans. Once this has been assured, further assessment of feasibility should be based upon three factors:

Technical Feasibility - Can the computer really solve the problem? Has a similar system ever been implemented and can it be obtained? Can the cost and risk of failure or delay be measured? Is the technology proven and will it still be valid when the system is complete?