NISTIR 4377

A FRAMEWORK FOR DEVELOPING A CALS DATA DICTIONARY

David K. Jefferson

U.S. DEPARTMENT OF COMMERCE
National Institute of Standards
and Technology
National Computer Systems Laboratory
Gaithersburg, MD 20899

U.S. DEPARTMENT OF COMMERCE
Robert A. Mosbacher, Secretary
NATIONAL INSTITUTE OF STANDARDS
AND TECHNOLOGY
John W. Lyons, Director

NIST

NISTIR 4377

A FRAMEWORK FOR DEVELOPING A CALS DATA DICTIONARY

David K. Jefferson

U.S. DEPARTMENT OF COMMERCE
National Institute of Standards
and Technology
National Computer Systems Laboratory
Gaithersburg, MD 20899

July 1990

U.S. DEPARTMENT OF COMMERCE
Robert A. Mosbacher, Secretary
NATIONAL INSTITUTE OF STANDARDS
AND TECHNOLOGY
John W. Lyons, Director

ABSTRACT

This paper provides guidance for the development of data dictionaries for the Computer-aided Acquisition and Logistic Support (CALS) Program. The objective is to present the costs and benefits of alternative architectures; five levels of service are analyzed. A brief tutorial on data dictionaries is included. Six steps are recommended for data dictionary development.

Key Words: CALS; Computer-aided Acquisition and Logistic Support; data dictionary; Information Resource Management; IRM; Three-Architecture Model; Three Schema Architecture.

TABLE OF CONTENTS

1. EXECUTIVE SUMMARY

A data dictionary system can help an organization, such as CALS, develop and maintain an organization-wide dictionary, to assist in managing the development, operation, and maintenance of the organization's information systems. A data dictionary system can be used to support data standardization and information management for virtually every aspect of the information system lifecycle. Data dictionary systems, however, have different types and degrees of functionality. Moreover, data dictionary systems can be used in a variety of ways. This paper addresses data dictionary architectures to determine the capabilities and usages needed to support the complex environment of CALS.

1.1 Purpose

The purpose of this paper is to present the major alternatives for data dictionary architectures and the relative advantages and disadvantages of those architectures. The paper is intended to be understandable to a wide audience concerned with the need for data dictionaries in CALS, so some of the material is of a tutorial nature.

The paper describes what can be done, not how to do it; technical details are omitted in the interests of brevity and the patience of the intended audience. The primary issue raised is that of the intended scope of the data dictionary -- an almost unlimited scope is possible, so it is necessary to balance benefits against cost.

1.2 Organization

The paper consists of Section 1, this Executive Summary, plus five major sections. Section 2 describes some important underlying concepts that form the basis of the analysis in this paper. The subsections are the following:

a. Information Resource Management (IRM) -- This subsection describes the scope of shared information resources in the CALS environment.

b. Three-Architecture Model -- This subsection describes the information sharing concepts that are fundamental to the management and control of shared resources in the CALS environment.

c. Quality of the Data Dictionary -- This subsection describes the need for a broad data dictionary scope and computer-based tools in order to ensure that the data dictionary correctly defines the actual and required information resources.

d. Data Dictionary Control of Information Resources -- This subsection describes three degrees of control that a data dictionary can exert over a database management system or other program.

Section 3 presents five levels of service which could be provided by data dictionaries. The series progressively enlarges the scope and functionality of the data dictionary, providing more benefits at a generally higher development cost. The subsections address the following levels of service:

a. Define isolated objects (e.g., define data elements)

b. Define simple structures (e.g., define a data hierarchy)

c. Define complex structures and rules (e.g., define a database schema)

d. Support multiple views and quality assurance (e.g., provide structural extensibility)

e. Support the IRM lifecycle (e.g., provide version control)

Section 4 presents the view that a data dictionary can be considered to be just another information resource, and as such should be managed according to IRM principles.

Section 5 expands upon the previous section to suggest the use of the data dictionary to manage its own lifecycle, as well as the information system lifecycle.

Section 6 then outlines a concept for development of integrated CALS data dictionaries. The suggested steps are:

a. Determine the long-term scope of the CALS data dictionary requirements

b. Select a data modeling technique

c. Acquire tools

d. Develop a baseline dictionary

e. Logically integrate other dictionaries

f. Physically distribute dictionaries for efficiency

2. CONCEPTS

A number of important underlying concepts are introduced in this section to provide a basis of understanding for the analysis presented in this paper. The concepts are briefly stated, but have very significant implications for the design, construction, operation, and maintenance of information systems. These concepts are fundamental to the notion that computer processable definitions are the key to developing complex information systems. If these concepts are accepted, then consequently, the appropriate use of data dictionaries is critical in establishing data sharing for CALS.

2.1 Information Resource Management

Information Resource Management (IRM) is a disciplined approach to managing all aspects of information and information processing as vital business resources. These resources, and techniques for managing them, are described below.

2.1.1 Data

Data may be classified in many different ways. For example, data may be classified according to usage or derivation (e.g., design, maintenance, etc.), external appearance (e.g., text, graphics, etc.), internal representation (e.g., character string, numeric, etc.), version (e.g., a number indicating a change level), and variation (i.e., specific modification of some baseline definition for a particular application or computer environment).

Data may also be classified according to its position in the Three-Architecture Model (Section 2.2 below), as data:

a. Conceptually representing reality in accordance with data integrity rules.

b. Presented to a user.

c. Physically stored in an information processing system.

IRM is based on explicit data definitions. These definitions are independent of the actual data values and the software that manipulates the data. The data definitions are part of the information resource, and can be managed by a data dictionary.

2.1.1.1 Levels of Data Abstraction

This subsection describes how data can be defined at the following increasing levels of abstraction:

a. **Application-level** data (i.e., data used in operational systems) describes the "real world." Logistics data is application-level data. For example, the value of a Commercial and Government Entity (CAGE) code identifies a design control activity or actual manufacturer.

b. **Dictionary-level** data (i.e., data describing data, or "metadata") defines the structure of application-level data. Data element definitions and process definitions are dictionary-level data. For example, a CAGE code has a specific length in characters, and might appear in a specific EDI_message.

c. **Dictionary-definition-level** data (i.e., data describing data definitions) defines the structure of dictionary-level data. Rules for defining data elements and processes are dictionary-definition-level data. For example, an EDI_message might be defined as an ENTITY that can contain data.

Note that the increasing levels of data abstraction answer a series of questions that ask "what is it?":

a. "What is this number?" "It's a CAGE code."

b. "What is a CAGE code?" "It's in an EDI_message."

c. "What is an EDI_message?" "It's an ENTITY that contains data." If ENTITY is a primitive concept, there is no need to continue with further levels of abstraction.

As a more general example, suppose that an application system is concerned with parts and suppliers of a weapon system. The application data could be stored as an **application-level** database and maintained by SQL (Structured Query Language) database manipulation commands. The definition of the columns, tables, etc., comprising that application-level database could be stored in a **dictionary-level** database and maintained by SQL data definition commands. The structure of the application database and the data integrity rules could also be defined with IDEF1X (Integrated Computer Aided Manufacturing Definition Method 1 - Extended) and stored in a **dictionary-level** database maintained by an IDEF1X Computer-Aided Software Engineering (CASE) tool. The concepts of column, table, etc., could be stored in a **dictionary-definition-level** database and maintained by the dictionary definition commands of an Information Resource Dictionary System (IRDS).

As another example, a document can consist of **application-level** data, such as text and graphics. A Standard Generalized Markup Language (SGML) Document Type Definition (DTD) can be viewed as a **dictionary-level** description of possible tags and document structures. The SGML standard itself is at the **dictionary-definition-level**.

2.1.1.2 Schema and Data Model

The term "schema" will be used to refer to dictionary-level data which describes the structure of a database; an example would be MIL-STD-1388-2B Appendix A, LSAR (Logistic Support Analysis Record) Relational Tables. The term "data model" will be used to refer to a schema plus related data, such as data integrity rules; "data model" will therefore refer to dictionary-level data. The term "data modeling technique" will be used to mean a technique, such as IDEF1X or NIAM (Nijssen Information Analysis Method), for constructing a schema or data model; an example would be the use of the IDEF1X data modeling technique to construct the LSAR IDEF1X data model[1].

2.1.1.3 Data Integrity Rules

Data integrity rules specify explicit representations of business rules. Examples include rules specifying data values (e.g., a CAGE code must consist of five characters), data relationships (e.g., people cannot be assigned to a task unless they have the appropriate skills), and security rules (e.g., only designers can modify drawings). The specific means of data integrity representation depends on the mechanism available; for example, SQL data definition provides the syntax for expressing various types of rules, while a data dictionary system may specify a mechanism for defining similar or other types of rules.

2.1.1.4 Business Rules

Business rules represent the basic principles which control the development and operation of manual and automated processes. Business rules should be explicitly defined and incorporated where possible into process and data definitions, in order to ensure that they are obeyed by the information processing systems. Business rules should be represented by data integrity rules wherever possible, so that all processes will conform to the same rules; representation in computer programs is less desirable, because the application programs may follow different rules.

2.1.2 Processes

An understanding of an organization's processes, whether performed by humans, external machines, or software programs, is considered to be an important part of IRM. Analyses of business processes generally drive the analysis and definition of data. If processes and data are explicitly defined in a computer-processable form, as in a CASE tool or a data dictionary, it is possible to perform a wide range of analyses and other functions.

[1] MRSA (Material Readiness Support Activity) MIL-STD-1388-2B LSAR Seminar, 3 August 1989, NIST.

Some functions that may be performed by CASE tools include: (1) analysis of completeness (e.g., data must be created, maintained, and used by some process or processes); (2) analysis of consistency (e.g., there should be only one organization responsible for a certain resource); (3) analysis of the effects of change (e.g., determine which processes and databases are affected by changing the definition of a data element); (4) definition of database schemas (i.e., derive normalized schemas for relational databases from data definitions); and (5) generation of application programs (i.e., derive code from module definitions).

The role of the data dictionary is to maintain an integrated, coordinated repository for all of the data and process definitions needed for IRM; the role of a CASE tool is to analyze those definitions. CASE tools and data dictionary systems can be used together to provide an integrated information system development environment. When used with CASE tools, the underlying data dictionary system is often referred to as the repository.

Note that CASE tools and data dictionary systems are themselves important information resources that must be managed.

2.1.3 Computer and Communications Hardware and Software

Computer and communication systems hardware and software, such as mainframes, personal computers, networks, application programs, system software, etc., may be explicitly represented and managed with assistance from one or more CASE tools and/or a data dictionary. This extends the degree to which analyses can be performed to determine completeness, consistency, the effects of change, etc. Usage statistics, for example, may assist in the optimization of a physical database structure.

2.2 Three-Architecture Model of Information Sharing

The ad hoc Study Group on Data Base Management Systems was established in 1972 by the Standards Planning and Requirements Committee (SPARC) of the American National Standards Committee on Computers and Information Processing (ANSI/X3). The Study Group produced what came to be known as the "ANSI/SPARC three schema architecture" for database management systems. This architecture was based on three types of schemas: a conceptual schema of current and planned information requirements, external schemas representing user views, and internal schemas of actual physical implementations.

The conceptual schema provides a single, complete, consistent data definition to which both external schemas and internal schemas can be mapped; a mapping from an internal schema to an external schema can therefore be defined as the concatenation of a mapping from the internal schema to the conceptual schema and a mapping from the conceptual schema to the external schema. The three schema architecture is a very important step toward data independence; modifications to the physical representation of data can be made without affecting the programs that use the data. The three schema architecture has been very successful in directing the development of a new generation of standards and products.

The Three-Architecture Model, an extension of the ANSI/SPARC three schema architecture, consists of the architectures described below.

2.2.1 Control Architecture

The Control Architecture is an extension of the conceptual schema to provide an IRM view. It contains not only the schema definition but also the conceptual views of other information resources such as the business rules and their representations as data integrity rules. Examples of Control Architectures are the LSAR data model developed using the IDEF1X data modeling technique, and the STEP (Standard for Exchange of Product Data) data model.

2.2.2 Information Architecture

The Information Architecture is an extension of the external schemas. It contains application-oriented views of data as it should appear, for example, on screens, in documents, and in files. The Information Architecture should include the rules specified by relevant standards such as the following:

a. MIL-STD-1388-2B Appendix B, Standard Reports for LSAR

b. MIL-M-28001 for document content

c. Future standards for the content of interactive electronic technical manuals

2.2.3 Computer Systems Architecture

The Computer Systems Architecture is an extension of the internal schemas. It is a technology view that provides for transparency of data distribution in an open systems environment, as well as interfaces to legacy data. The Computer Systems Architecture should include the rules specified by relevant standards such as the following:

a. MIL-STD-1388-2B Appendix A, LSAR Relational Tables

b. MIL-STD-1840A for the physical format of files of various types of data

c. MIL-D-28000, MIL-R-28002, and MIL-D-28003 for the physical format of graphics files

2.3 Quality of the Data Dictionary

The quality of the data dictionary is a measure of its fidelity to the actual or required information resources. It is possible to achieve high quality either by exhaustive human analysis or by the use of computer-based tools.

A data dictionary maintained entirely by people would probably have minimum scope and minimum cross-referencing in order to minimize maintenance effort and internal inconsistencies. Such a data dictionary could not be effectively analyzed by computer-based tools, because they would have no way of comparing it with the actual or required information resources.

A data dictionary maintained primarily by computer-based tools would be quite different; it would have maximum scope and cross-referencing in order to maximize the opportunities for comparing different points of view, and thereby detecting inconsistencies that should be examined by people. Specifically, one of the most important aspects of activity or event modeling is the possibility of discovering common data. For example, data object X may be produced by process A, and data object Y may be used by process B; activity modeling could reveal that X and Y are identical because A's output is B's input.

The framework suggested in this report emphasizes the use of computer-based tools and maximum scope. Note that this approach has the additional benefit that the data dictionary can be used in a Corporate Information Management study to improve processes as well as data.

2.4 Data Dictionary Control of Information Resources

Three degrees of information resource control that can be exerted by a data dictionary are:

a. A passive data dictionary is used to produce documentation and analyses suitable for use by humans, but has no direct link to the database management system or other software. A passive data dictionary is suitable for planning. It is generally not suitable for operational control of information resources, because it is frequently out-of-date and can easily be bypassed. Synchronization of the data dictionary with other software requires manual translation.

b. An active data dictionary can interchange data with other software. For example, it could provide data definitions to be compiled into a program. An active data dictionary is suitable for operational control if data dictionary changes trigger appropriate program recompilation. Synchronization of the data dictionary with other software requires initiation of a program (e.g., a compiler).

c. An integrated data dictionary (also called a dynamic data dictionary) shares data with other software. The term active data dictionary is sometimes used in this sense, but then it may not be clear whether data interchange or data sharing is intended. For example, an integrated data dictionary could provide data definitions to a database management system as the database management system was processing a user's query. An integrated data dictionary ensures that the data dictionary and other software are always synchronized, which may be critical for some applications, but it may result in unacceptable overhead for other applications.

3. LEVELS OF SERVICE FOR A DATA DICTIONARY

This section expands upon the preceding description of alternative degrees of control that can be provided by a data dictionary. Five levels of data dictionary service are presented for consideration. Level 1 is a passive data dictionary of minimal scope and value. Level 2 expands the scope but is still a limited, passive data dictionary.

Levels 3, 4, and 5 represent progressively more powerful data dictionaries that could each be passive, active, or integrated. Level 3 is adequate to support functional users. Levels 4 and 5 provide additional services to support functional developers.

Each level is described from three points of view: what facts it can represent, what it requires in terms of computer software, and what costs and benefits can be expected.

3.1 Level 1: Define Isolated Objects

The level 1 data dictionary provides documentation of isolated objects, such as data elements, to people. The following are typical of the data element definitions in such a data dictionary:

a. Name

b. Length

c. Text describing the meaning of the data element

d. Simple data integrity rules involving single data elements and such information as data type and valid data values

The minimal requirements for such a data dictionary are quite low:

a. Data element naming conventions are necessary to provide some level of consistency.

b. Paper is adequate to distribute the data definitions.

The benefits as well as costs may be very limited:

a. Low cost and development time are possible, because the scope of the data dictionary is so limited that people will be unlikely to realize how much redundancy exists among the data elements.

b. A minimal degree of understanding and consensus will be developed as a result.

c. Low quality is likely.

d. A high degree of maintenance effort may be required, because redundancy can be avoided only by comparing each new data element with every old data element.

e. The data dictionary may be unused, in part because it is likely to be unreliable and out-of-date.

The degree of understanding and consensus and the quality of the data dictionary may be increased at the cost of very substantial human effort. Even so, a high-quality, reliable, up-to-date data dictionary probably cannot be achieved, because this level of data dictionary cannot be adequately supported by computer analysis and maintenance. This level of data dictionary is insufficient for CALS, but may serve as the basis for a higher-level data dictionary.

3.2 Level 2: Define Simple Structures

A much more powerful type of data dictionary is possible with modest support from basic computer hardware and software. This level of data dictionary includes the definition of isolated objects, but adds capabilities for representing relationships among dictionary objects, and is much easier to maintain than a paper data dictionary. The types of objects and relationships that may be represented include the following:

a. Containment relationships, such as which files contain which records, and which records contain which data elements.

b. Keys (identifiers) for records.

c. Data integrity rules involving more than one data element from a single record.

The requirements are modest:

a. Data naming conventions are necessary for records and files as well as data elements.

b. A file management system that can represent hierarchical relationships is adequate (a word processing system may be adequate for a small number of objects).

The benefits are considerably greater than the previous level, while costs are quite low:

a. Low cost in computer resources.

b. Low development time is possible, because the scope is still quite limited.

c. Quality is only moderate, since extensive analysis to ensure quality is not possible given the limited scope of the data dictionary.

d. Only a moderate degree of maintenance effort may be required, since each new data element need be compared only with related old data elements.

e. Hard to enforce use of the data dictionary, since programs and queries can be developed without using it.

A high-quality, reliable, up-to-date data dictionary will be costly and the scope will be limited. This level of data dictionary can provide an index to pre-defined data collections or queries.

3.3 Level 3: Define Complex Structures and Rules

More powerful software and a richer set of data dictionary objects provide much more opportunity for ensuring the high quality of the data dictionary and the information system that it can document and control. This level of data dictionary service includes the preceding levels of service, but adds rules that can be enforced by appropriate database management systems. The types of objects and relationships that may be represented should include everything in the data definition language of the database management system, such as the following:

a. Relational data integrity rules (i.e., rules involving two or more records)

b. Event-driven triggering of processes

c. Complex rules, including rules for automatically propagating the effects of changes

d. Security rules

The human skills and the computer hardware and software required for this level are considerably greater than for the preceding levels:

a. A powerful database management system (e.g., relational, object oriented, or entity-relationship) is required to develop and maintain the complex structure of the data dictionary.

b. Data modeling methodology and graphic display are required to support human analysis and verification of the dictionary.

The benefits and costs are generally high, because of the increased scope and opportunities for greater quality control:

a. High cost in computer software and human training in that software.

b. High degree of organization-wide understanding and consensus, at the cost of a long development time.

c. Potentially high quality results.

d. Only a moderate degree of maintenance effort may be required, because the ripple effects of changes are offset by computer analyses.

e. Easily enforceable data dictionary use if the dictionary is active or integrated.

A high-quality, reliable, up-to-date data dictionary will be costly. This level of data dictionary will satisfy the operational requirements of CALS, including ad hoc data manipulation.

3.4 Level 4: Support Multiple Views and Quality Assurance

Additional utility and greater opportunities for analyzing the quality of a data dictionary are possible if the data dictionary is used to represent a larger part of the information resources. However, it is almost impossible for a data dictionary vendor to anticipate all the possible information resources of interest to a user. Therefore, the data dictionary must generally be structurally extensible by the user organization; that is, the user organization must be able to add new types of objects to those provided by the vendor. This means that the data dictionary must be capable of operating at two different levels of data abstraction: at the dictionary-definition-level, where object types are defined (e.g., screen, report, query); and dictionary-level, where object instances are defined (e.g., a specific screen definition).

A substantial benefit from this level is the capability of representing the components of and the mappings between the Architectures of the Three-Architecture Model. In particular, it is possible to integrate legacy databases and new databases through the data dictionary. This is easy only if the database schemas are identical, or if there is one schema that contains the others. The mapping among databases can then be described by data at the dictionary-definition-level.

This data dictionary level should include the following:

a. Definition of mapping among Architectures.

b. Documentation for human analysis, including the effects of change.

c. Definition of information resources such as communications.

d. Metadata interchange (i.e., dictionary-level interchange), in order to provide metadata to and accept metadata from CASE tools.

The benefits and costs are substantial:

a. Very high cost and development time, due in part to the requirement for a very high degree of understanding and consensus.

b. Potentially very high quality results, due to the number of analyses that can be performed.

c. High maintenance cost and time, due to substantial complexity.

d. Easily enforceable data dictionary use if the dictionary is active or integrated.

e. Potential support for a distributed database with legacy data.

A high-quality, reliable, up-to-date data dictionary will be costly, but the scope of the data dictionary contents will be quite extensive. This data dictionary architecture will satisfy the short-term planning requirements of CALS. An operational subset, consisting only of the dictionary-level data needed by the DBMS, would be efficient for the purposes of operational users.

3.5 Level 5: Support the IRM Lifecycle

The final data dictionary level provides the capabilities of the previous levels, and also provides the following functions to manage the information system lifecycle:

a. Configuration management (including the data dictionary itself) and protection of components with respect to considerations such as quality.

b. Definition of requirements.

c. Very detailed representation of designs and components in order to facilitate data sharing.

d. Facilitation of system integration by means of very accurate representations of data definitions.

e. Statistics collection and analysis to improve efficiency and minimize data transformation.

The requirements for such a data dictionary include those of the previous levels, plus the following:

a. Versioning capability to support traceability through modifications to data dictionary contents and definitions.

b. Variation capability to support nuances in meanings according to different user-views, lifecycle phases, and other uses.

The benefits and costs are again high:

a. High cost and time to develop.

b. High degree of organization-wide understanding and consensus.

c. Very high quality results.

d. Moderate maintenance cost and time.

e. Easily enforceable data dictionary use if the dictionary is active or integrated.

f. Potential support for a distributed database with legacy data.

A high-quality, reliable, up-to-date data dictionary will be moderately costly; the data dictionary will be sufficiently broad in scope to assist in its own maintenance. This data dictionary architecture will satisfy the short- and long-term planning requirements of CALS. An operational subset would be efficient for the purposes of operational users.

4. DATA DICTIONARY AS ANOTHER INFORMATION RESOURCE

These alternative data dictionary levels are sufficiently complex to require extensive time and money for design, acquisition, operation, and maintenance -- any alternative represents a substantial investment requiring a prominent place in IRM planning.

The more powerful data dictionaries can be viewed as supporting the Three-Architecture Model: representing the rules of the Control Architecture, presenting those rules via CASE tools in the Information Architecture, and representing implementations of those rules in the Computer Systems Architecture.

Data dictionaries can also be viewed as being supported by the Three-Architecture Model. Operational data dictionaries are implemented within the Computer Systems Architecture, and they can be represented within a planning data dictionary in order to understand their relationships with other resources, to plan for future roles and relationships, and to optimize operations.

A reasonable question is "How is a data dictionary different from an application database?" The following differences are significant:

a. The data dictionary is a repository of dictionary-level data, which describes application-level data, and which is generally less voluminous and less volatile than application-level data.

b. The data dictionary requires a broader scope in that it:

   (1) Encompasses all applications.

   (2) Controls and documents the lifecycle of the information system.

   (3) Includes descriptions of data, activities, and events.

   (4) Requires a greater degree of structural extensibility.

c. Finally, and perhaps most significantly, the data dictionary is owned by the information systems' organization, and not by any particular application.

5. THE LIFECYCLE OF A DATA DICTIONARY

The rather abstract alternatives and observations in the preceding sections lead to the conclusion that the data dictionary issue is exceedingly complex. This section is intended to bring together many different points by relating them to a data dictionary lifecycle. The next section will then present a development concept.

5.1 Development of Requirements

The first phase in the data dictionary lifecycle is the development of data dictionary requirements. This framework is intended to suggest what some of those requirements might be. The levels of service satisfy broad classes of requirements.

5.2 Development of Methodology and Techniques

The second phase in the data dictionary lifecycle is the development of methodology and techniques. The methodology and techniques provide the information collection and analysis procedures necessary to develop the information system. They should address the following questions:

a. What objects are to be defined in the data dictionary? The following are among the general classes of objects that could be defined in the data dictionary: data, data integrity rules, facts about the derivation of data, processes, events, computer systems, and the lifecycle relationships among these objects.

b. When are the objects to be defined? That is, is there a particular sequence in which each object or class of objects should be defined? Possible sequences include the following:

   (1) Top-down -- define general objects, such as data classes and major subsystems, then add details consistent with those objects.

   (2) Bottom-up -- define details, such as data elements and detailed processes, then generalize and integrate other details.

   (3) Critical first -- define critical parts of the system, such as the most important data classes and their data elements, then integrate other definitions.

   It is possible that a methodology may work adequately with any of these sequences.

c. How are objects to be defined? What methodologies and analysis techniques can be applied? Possible tools and techniques include the following:

   (1) IDEF0 for activity modeling

   (2) IDEF1X for data modeling (particularly powerful when used in conjunction with IDEF0)

   (3) NIAM for bottom-up data modeling

   (4) EXPRESS for data modeling

   (5) Naming conventions and tools to ensure that they are followed

d. How can separately developed subsystems be integrated? In particular, what are the areas of intersection among subsystems? The most critical issue is how to detect areas of intersection. Guidelines include the following:

(1) Synonyms indicate areas of intersection among the subsystems that are being integrated; they should be discovered and recorded.

(2) Homonyms may or may not indicate intersections, but they must be discovered and resolved for the system to be consistent.

(3) Naming conventions will assist in the recognition of synonyms and homonyms.

(4) Context, such as that provided by adding activity models to data models, will help to reduce ambiguities and help in discovering synonyms and homonyms.

(5) Definitions can be manually compared to help in discovering synonyms and homonyms.

(6) Rules may be compared to help find synonyms and homonyms; combined rules must be developed if there are inconsistencies among subsystems.

5.3 Analysis of Tools

The third phase in the data dictionary lifecycle is the analysis of tools for developing, operating, and maintaining the data dictionary. The primary tool is the data dictionary system itself, which may be passive, active, or integrated. Federal Information Processing Standard Publication (FIPS PUB) 156 establishes the ANSI Information Resource Dictionary System (IRDS) as the standard for passive data dictionary systems within the federal government; additional IRDS interfaces will provide the functionality needed for the IRDS to be active or integrated.

A data dictionary system can have a fixed set of object types, which are defined by the vendor, or it can be structurally extensible, as in the IRDS. A structurally extensible data dictionary system supports the user organization in manipulating dictionary-definition-level data. Structural extensibility is almost certainly a requirement for use in CALS, due to the scope of CALS.

A data dictionary system can provide various interfaces, such as the following:

a. Services interface responds to queries and updates from other software, such as database management systems -- this allows a data dictionary to be active and integrated.

b. Export/import interface exchanges bulk data with other software, such as other data dictionary systems or CASE tools -- this provides synchronization of different data dictionary systems and tools.

c. Human interface can have a specific syntax (e.g., the IRDS Command Language Interface) or can be functional (e.g., the IRDS Panel Interface, which could be implemented with graphics, menus, or forms).

Other useful tools include a database management system, which can serve as the host for a data dictionary system, and CASE tools, which assist in populating and maintaining the data dictionary, and can analyze and display the contents of the data dictionary.

5.4 Population, Use, and Maintenance of a Data Dictionary

Population of the data dictionary with metadata is a two-step process if the structure of the data dictionary is extensible: first the structure of the dictionary is defined in terms of dictionary-definition-level data, and then the dictionary-level data (metadata) is defined in terms of that structure.

Use and maintenance of a data dictionary can be both direct, through the capabilities of the data dictionary system itself, and indirect, through the use of CASE tools that can use the data dictionary as an integrated, coordinated repository of metadata.

The data dictionary may be either centralized or distributed; the latter is more likely in CALS Phase II, given the probable scope and the distributed nature of the CALS data itself. Considerations relevant to data dictionary distribution include:

a. Scope of component dictionaries.

b. Relationships among component dictionaries.

c. Configuration management of component dictionaries.

6. A CONCEPT FOR DATA DICTIONARY DEVELOPMENT

The suggested steps in data dictionary development for CALS are the following:

a. Determine the long-term scope of the CALS data dictionary requirements. The scope should not be limited by technology, as the technology is continually expanding.

b. Select a data modeling technique or methodology (IDEF1X, NIAM, EXPRESS, etc.) appropriate to the scope of the CALS data dictionary requirements and sufficient for integrating CALS data dictionaries.

c. Acquire tools appropriate to the methodology, considering the required levels of support for analysis, control, and documentation.

d. Develop a baseline dictionary. This step includes selection of a subsystem to be defined in the baseline dictionary, definition of the baseline dictionary structure in terms of dictionary-definition-level data, and population of the baseline dictionary with metadata.

e. Logically integrate each new dictionary with the baseline dictionary. This step includes selection of a subsystem to be defined, installation of the baseline structure, and population of the new dictionary with metadata. In rare cases the baseline structure will be extended, using dictionary-definition-level data, to form a new baseline structure.

f. Physically distribute dictionaries for efficiency.

Repeat steps e and f as many times as necessary for integration of all relevant subsystem descriptions.

7. ACKNOWLEDGEMENTS

The author is deeply indebted to Margaret H. Law for her technical contributions, for her encouragement, and for her editorial comments.

NIST-114A
(REV. 3-90)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY

BIBLIOGRAPHIC DATA SHEET

1. PUBLICATION OR REPORT NUMBER
NISTIR 4377

2. PERFORMING ORGANIZATION REPORT NUMBER

3. PUBLICATION DATE
July 1990

4. TITLE AND SUBTITLE

A Framework for Developing a CALS Data Dictionary

5. AUTHOR(S)

David K. Jefferson

6. PERFORMING ORGANIZATION (IF JOINT OR OTHER THAN NIST, SEE INSTRUCTIONS)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
GAITHERSBURG, MD 20899

7. CONTRACT/GRANT NUMBER

8. TYPE OF REPORT AND PERIOD COVERED

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (STREET, CITY, STATE, ZIP)

10. SUPPLEMENTARY NOTES

11. ABSTRACT (A 200-WORD OR LESS FACTUAL SUMMARY OF MOST SIGNIFICANT INFORMATION. IF DOCUMENT INCLUDES A SIGNIFICANT BIBLIOGRAPHY OR LITERATURE SURVEY, MENTION IT HERE.)

This paper provides guidance for the development of data dictionaries for the Computer-aided Acquisition and Logistic Support (CALS) Program. The objective is to present the costs and benefits of alternative architectures; five levels of service are analyzed. A brief tutorial on data dictionaries is included. Six steps are recommended for data dictionary development.

12. KEY WORDS (6 TO 12 ENTRIES; ALPHABETICAL ORDER; CAPITALIZE ONLY PROPER NAMES; AND SEPARATE KEY WORDS BY SEMICOLONS)

CALS; Computer-aided Acquisition and Logistic support; data dictionary; Information Resource Management; IRM; Three-Architecture Model; Three Schema Architecture.

13. AVAILABILITY

☒ UNLIMITED

□ FOR OFFICIAL DISTRIBUTION. DO NOT RELEASE TO NATIONAL TECHNICAL INFORMATION SERVICE (NTIS).

□ ORDER FROM SUPERINTENDENT OF DOCUMENTS, U.S. GOVERNMENT PRINTING OFFICE, WASHINGTON, DC 20402.

☒ ORDER FROM NATIONAL TECHNICAL INFORMATION SERVICE (NTIS), SPRINGFIELD, VA 22161.

14. NUMBER OF PRINTED PAGES
24

15. PRICE
A02

ELECTRONIC FORM