NISTIR 4547



# A Standard Generalized Markup Language Encoding of the Office Document Architecture Document Application Profile

**Ronald B. Wilson**

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

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

NIST

NISTIR 4547

# A Standard Generalized Markup Language Encoding of the Office Document Architecture Document Application Profile

**Ronald B. Wilson**

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

**April 1991**


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

# A STANDARD GENERALIZED MARKUP LANGUAGE ENCODING
*of the*
# OFFICE DOCUMENT ARCHITECTURE
# DOCUMENT APPLICATION PROFILE

by

Ronald B. Wilson

Systems and Software Technology Division
National Computer Systems Laboratory
National Institute of Standards and Technology

February 22, 1991

| | | |
|---|---|---|
| FY90 CALS TASK | 3.1.2. | "Assist in the SGML encoding development through the OSI Implementor's Workshop" |
| | 3.2.2. | "Develop Profile and supporting text" |
| Deliverables | | SGML/ODA Encoding Report |
| | | Common SGML/ASN.1 Profile |

## - ABSTRACT -

The Office Systems Engineering Group (OSE) at NIST was tasked by the CALS' Project Office to bring CALS' requirements to the Open Systems Interconnection (OSI) Implementor's Workshop sponsored by NIST. CALS tasked the OSE Group to assist in the SGML encoding of the Office Document Architecture (ODA) Document Application Profile (DAP). NIST offered to encode the ODA DAP in the Standard Generalized Markup Language (SGML) to illustrate the similarities between the two standards and to provide a common SGML/ASN.1 profile.

The report describes in various levels of detail the two international standards. It then discusses, by offering a simple example, the methodology involved in performing an SGML encoding. Subsequently, it examines the SGML encoding of the ODA DAP. The report provides two accompanying tables with the SGML encoding and finishes with a brief summary of the two standards.

# - Table of Contents -

LIST OF FIGURES

LIST OF TABLES

## 1. INTRODUCTION

This paper illustrates a Standard Generalized Markup Language (SGML) [1] encoding of the Office Document Architecture (ODA) Document Application Profile (DAP) [2]. This Document Application Profile is an ODA [3] application designed by groups of ODA users. One well known example in the United States of users designing specific applications in the electronic publishing area is the "Implementation Agreement on ODA" developed at the National Institute of Standards and Technology (NIST) by the Special Interest Group (SIG) for ODA of the Open Systems Interconnection (OSI) Implementors Workshop.

In 1983, at the request of industry, NIST organized the NIST Workshop for Implementors of OSI to bring together future users and potential suppliers of OSI protocols. The Workshop organizes its work through Special Interest Groups that prepare technical documentation. The Workshop meets four times a year at NIST where each SIG is required to convene its meeting. NIST organizes, administers, and makes technical contributions to the Workshop.

The Office Systems Engineering (OSE) Group at NIST was tasked by the Computer-aided Acquisition and Logistic Support (CALS) Project Office to bring CALS' requirements to the OSI Implementor's Workshop. Besides representing the MIL-M-28001 [4] requirements at the Workshops, CALS tasked the OSE Group to assist in the SGML encoding of the ODA DAP. This task was wholly undertaken by NIST in order to accelerate the standards process and demonstrate the similarities between the two standards: Standard Generalized Markup Language and Office Document Architecture.

## 2. PURPOSE

The purpose of this paper is to encode the ODA DAP in SGML in order to illustrate the similarity between SGML and ODA and provide a common SGML/ASN.1 profile. The method of interchange specified by the ODA DAP is the Abstract Syntax Notation One (ASN.1) [5]. The encoding demonstrates the nearly one-to-one mapping of the elements making up an SGML document type definition describing a document of book form to a logical representation of an ODA document of book form.

## 3. SCOPE

This paper illustrates an application of the Standard Generalized Markup Language (SGML) and is intended for persons with some knowledge of ODA and SGML. The SGML encoding defines a document application profile suitable for interchanging documents in processable form in accordance with ISO 8613. The ODA DAP is intended for interchange of compound documents between document processing systems. It is intended for documents containing character text, raster graphics, or geometric graphics. The documents addressed by this SGML encoded DAP range from simple memos to highly structured technical documents.

## 4. OVERVIEW

Two international standards were utilized to produce this paper: ISO 8879 [1] and ISO 8613 [3]. The following subsections highlight their purpose and provide brief overviews of the standards.

### 4.1 SGML Overview

SGML is a set of rules for defining generalized markup applications. SGML describes the structure of text in electronic form in order to accomplish the following objectives:

- To ensure that documents marked up using SGML will be processable by a wide range of devices and systems;
- To ensure that markup and references to external objects (e.g., special characters, photographs) will be independent of any application, system, or device;
- To provide markup that is usable by both humans and machines;
- To allow implementation on any text processor, word processor, Computer Aided Design (CAD) device, publishing system, or other document processing system;
- To require character-set (e.g., ASCII, EBCDIC) independence; and
- To establish a definitional framework for specific applications.

SGML provides the facilities for defining:

- the structure of the document;
- the characters transmitted in a document;
- text that is to be used more than once in the document;
- externally created information that is to be incorporated into the text;
- special techniques used in marking up the text (tag minimization, etc.); and
- the manner in which text is to be processed.

Each SGML-coded document is split into a number of clearly identifiable elements, each containing either text or further levels of embedded subelements. Where the element is made up of a number of levels of subelements, the lowest level in each structure always contains either text or some other form of printable information (i.e., a graphic). The position and appearance of each element on the printed page is handled by the formatting process and is not addressed by ISO 8879 since it is system dependent. When the document is formatted, a set of procedures maps each element identifier in the text to the appropriate formatting instruction.

### 4.2 ODA Overview

The purpose of ODA is to facilitate the interchange of documents in a manner such that:

- different types of content, including text, image, graphic and sound, can coexist within a document; and

- the intentions of a document originator with respect to editing, formatting and presentation can be communicated most effectively.

Office Document Architecture provides for the representation of documents in three forms:

- formatted form, that allows documents to be presented as intended by the originator;
- processable form, that allows documents to be edited and formatted; and
- formatted processable form, that allows documents to be presented as well as edited and reformatted.

The concept of ODA is based on:

- the existence of a layout view and a logical view of the document; the view from the physical viewpoint (i.e., a collection of pages) and the view of its abstract components (i.e., an assembly of sentences);
- the existence of a specific structure and generic structure; the specific document structure is the one that the user may read; the generic structure is the template that guides the creation of the document and that could be re-used for its amendment; and
- the existence of document classes; a document class is the set of generic features that are common to a category of documents (i.e., technical manuals).

## 5. DESCRIPTION OF THE STANDARDS

The following subsections describe each standard in more detail.

### 5.1 Description of SGML

SGML concerns itself only with the logical structure of a document. It does not concern itself with the layout structure of the document, i.e., the formatting of the document. SGML documents consist of a number of interrelated elements with each element containing characters which serve a specific purpose. Each SGML document starts with a document type definition (DTD) which defines the structure of the document in terms of the elements it contains. Within the DTD each type of element found in the document is given a name, a generic identifier, by which it can be recognized. When placed within special markup delimiter characters, these generic identifiers form the tags used to identify the start and end of each element. SGML documents can be built from a series of subdocuments, each subdocument being a document in its own right. These subdocuments can be processed separately or taken together as a whole and processed that way. They may be transferred to other machines individually or as part of a main document. In the following paragraphs we will examine in more detail the various features of an SGML document.

Each SGML document is split into a number of clearly identifiable elements, each containing either text or further levels of embedded subelements. The lowest level element either contains text or some other form of printable information such as a graphic. The position and appearance on the page is handled by the formatting system and is not addressed by the SGML encoding, since SGML only deals with the logical structure of the document.

Defining the structure of the document through DTDs is not, however, the sole purpose of SGML. Before documents can be transmitted between systems, it is essential that the numeric values transmitted between machines have clearly defined meanings. ISO 8879 defines a reference concrete syntax which provides a common code set. This code set, defined in ISO 646 [6], is an ASCII-compatible character set that defines English codes for alphanumeric characters, punctuation symbols, and four special control characters. This reference concrete syntax can be modified to define most other languages.

The structure of an SGML-encoded document and the applicable SGML features used during preparation are formally defined at the start of the document in a set of SGML declarations. Many times an author will use or modify a previously defined set of SGML declarations. These declarations define a set of markup tags that can be used to define the logical elements of the document. Separate sets of tags may be used for different applications, though in many cases, the same basic tags can be used in each set. For example, a paragraph of text forms a logical element of a letter, technical report, or book, and can use the same markup tag in each type of document. By identifying similar areas, document designers can reduce the number of tags that need to be remembered by the authors.

## 5.2 Description of ODA

The key concept in the document architecture is that of structure. Document structure is the division and repeated subdivision of the content of a document into increasingly smaller parts. The parts are called objects. The structure has the form of a tree (see fig. 1). Within ODA, the architecture permits two structures to be applied to a document: a logical structure and a layout structure. In the logical structure, the document is divided and subdivided on the basis of the meaning. Examples include chapters, figures, and paragraphs. For the purposes of this paper, the layout structure will only briefly be examined. In the layout structure, the document is divided and subdivided on the basis of the layout. Examples of layout objects are pages and blocks. Figures 2 and 3 illustrate the layout structure of a document. Table III, ODA Operators, provides definitions of the ODA operators illustrated in figures 2 and 3. The following types of layout objects are defined in the document architecture: block, frame, page, page set, document layout root.

For logical objects, no classification other than "basic logical object," "composite logical object" or "document logical root" is defined in the document architecture. Logical object categories such as "chapter," "section" and "paragraph" are application-dependent.

The basic elements of the content of a document are called content elements. For content consisting of character text, the content elements are characters. In the case of images or graphics, the content elements are picture elements (also called pels) or geometric graphics elements (line, arcs, polygons, etc.). When a document has both logical and layout structure, each content element belongs to exactly one basic logical object and one basic layout object. A content portion is a set of related content elements that belong to one basic logical object and one basic layout object. Therefore, it follows from this description that:

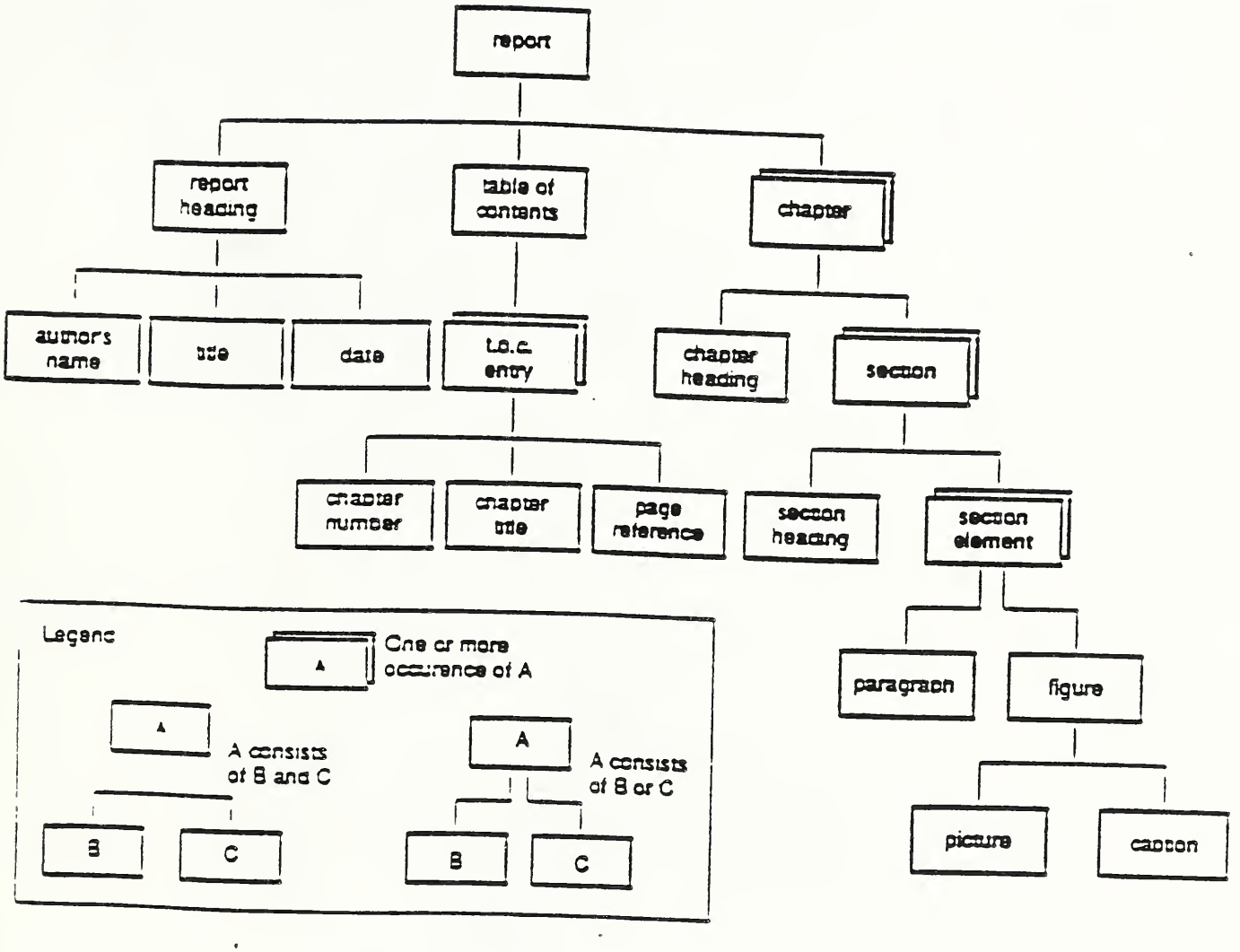



Figure 1. Example of a Logical View of a Document

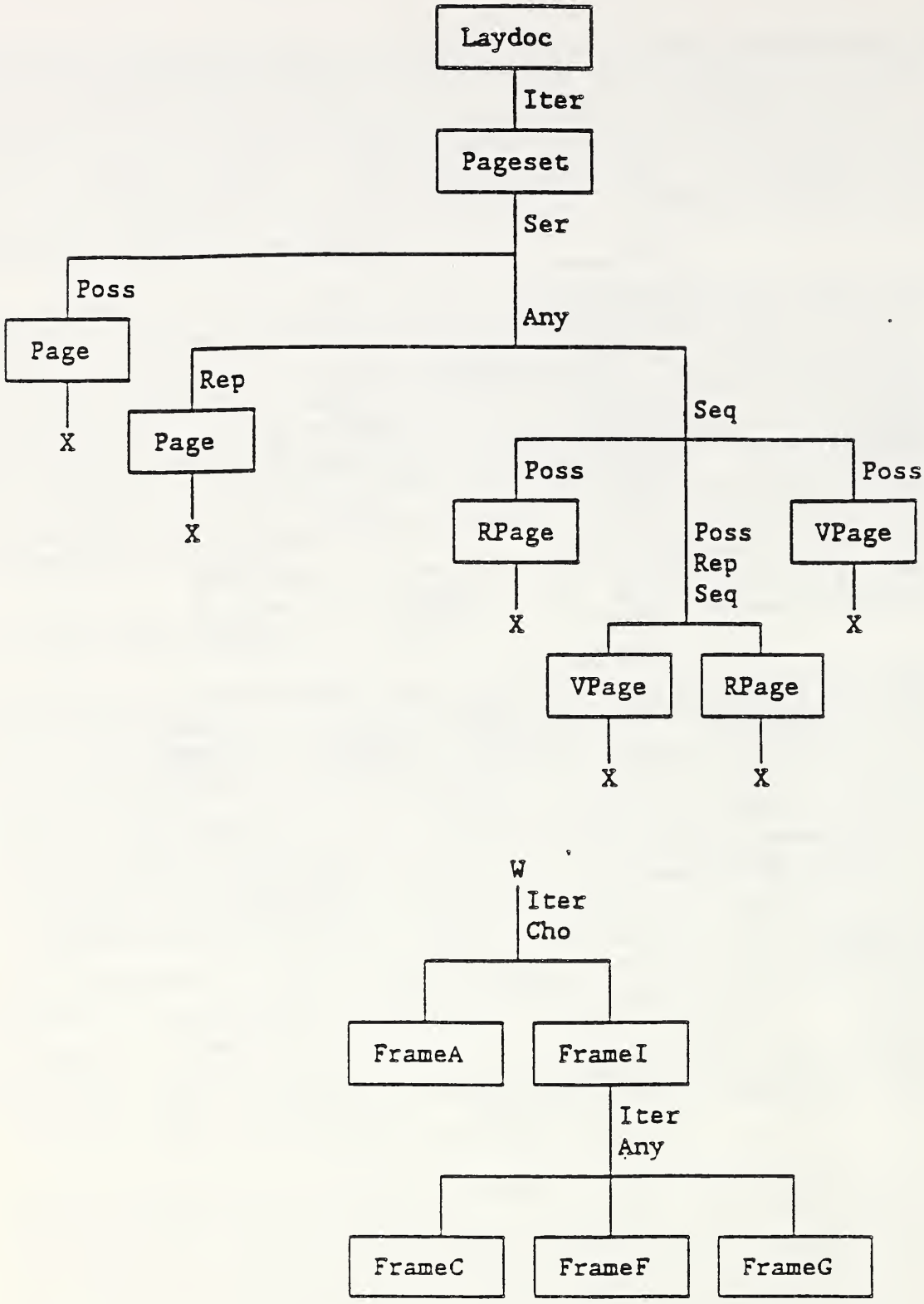
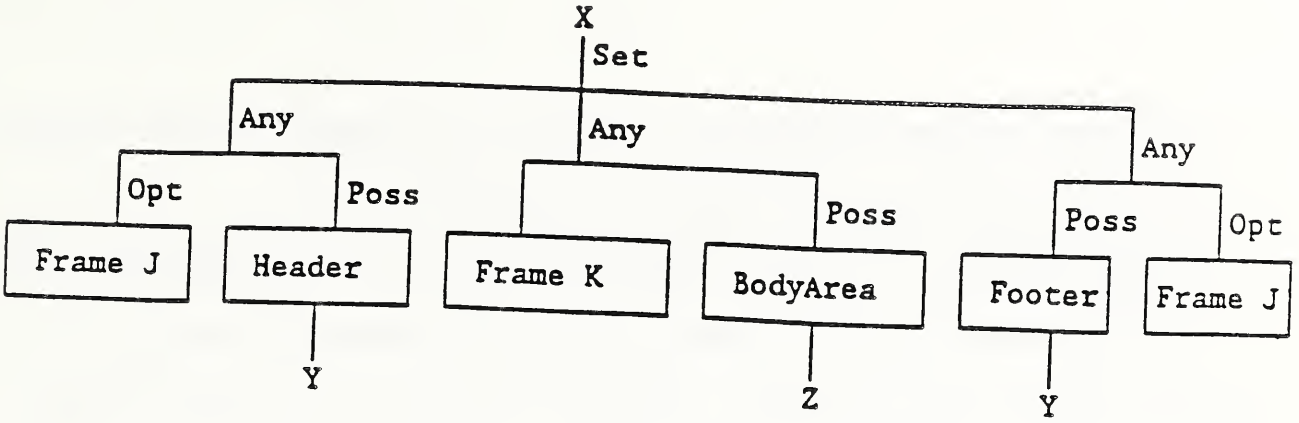
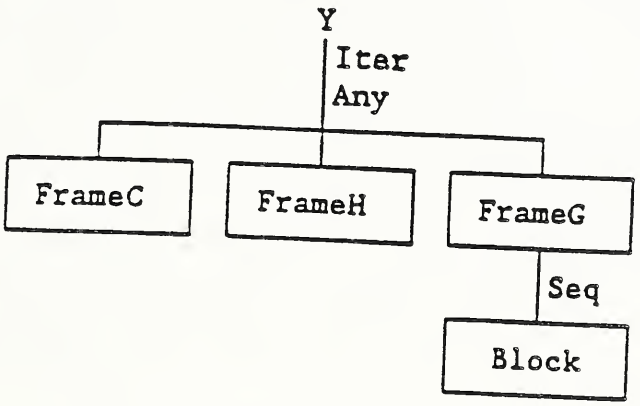



Figure 2. Diagram of a Layout Structure (1 of 2)



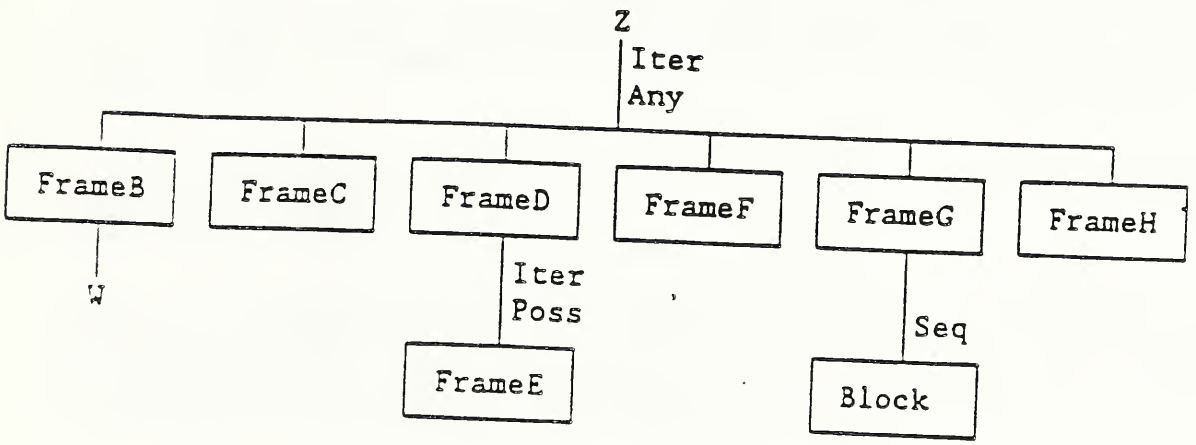




Figure 3. Diagram of a Layout Structure (2 of 2)

- a basic logical object has associated with it one or more content portions;
- a basic layout object has associated with it one or more content portions;
- any logical or layout object (basic or composite) has associated with it an integral number of content portions; and
- there is no general one-to-one correspondence between logical objects and layout objects.

Attributes are a property of a document, or a document constituent. They express a characteristic of the document or document component concerned, or a relationship with one or more documents or document components. A document profile is a set of attributes associated with a document as a whole and represents reference information about the document and may repeat information in the document content. Different sets of attributes are defined for basic logical objects, composite logical objects, document logical root, blocks, frames, pages, page sets and the document layout root. These are called document architecture attributes. Document architecture attributes are independent of the content type for the objects to which they apply. Examples of document architecture attributes are an "object identifier" for all objects, "subordinates" for composite objects, and layout directives such as the "offset" or "separation." In addition to document architecture attributes, there are presentation attributes that apply to basic logical and layout objects. A different set of presentation attributes is defined for each content architecture. Presentation attributes may be collected into presentation styles to which references may be made from both logical and layout objects.

The logical structure and layout structure are, in principle, independent of each other. The logical structure of a document is determined by the author and embedded in the document during the editing process. The layout structure is usually determined by a formatting process. The formatting process may be controlled by attributes called layout directives associated with the logical structure. Layout directives might include the requirement that a chapter begin on a new page, or the requirement that the title of a section and the first two lines of its first paragraph be presented on the same page.

The document profile consists of a set of attributes associated with a document as a whole. Along with reference information which facilitates storage and retrieval of the document, the document profile contains a summary of the document architecture features that are used. The attributes representing the capabilities for processing the document are called document characteristics and may include specifications concerning the form of the document or the content architectures used in the document. A document application profile is the specification of a combination of features that are defined in various parts of ISO 8613. It is identified by a unique ASN.1 object identifier obtained in accordance with the rules of ISO 8824. The document application profile must include:

- one or more document architecture levels;
- one or more content architecture levels;
- a document profile level; and
- an interchange format class.

## 6. APPLICATION OF SGML

The first step in applying SGML to a document is to select the document application and then perform a comprehensive document analysis. The document analysis is of critical importance for the success of the application. The analysis should be done toward a goal of producing a document type definition. The document type definition defines the structure of the document. It is very important to identify components of the document with a view toward processing the document in a different manner, such as multimedia publishing or producing secondary publications.

In analyzing the document, there are a number of tasks that must be performed. First, one should collect a number of document samples. These will be useful in assuring that the analysis adequately reflects the specific type of document and will help to point out any unusual structures or elements. Other tasks associated with the analysis include identifying the different data elements, naming the different data elements, assigning generic identifiers to these data elements, and assigning attributes if needed to these elements. In identifying the data elements, the hierarchical structure of the elements must be determined, starting from the complete document, then dividing it into major elements that in turn contain other elements and so on down to the character text level.

SGML specifies that the logical parts of a document can be expressed in a tree structure. Documents are composed of chapters; chapters of sections; sections of paragraphs; paragraphs of words. Words contain character data. A tree representation may be constructed for any document. Consider for example the office memorandum in figure 4 [7].

MEMORANDUM

To: Comrade Napoleon

From: Comrade Snowball

In Animal Farm, George Orwell says: ".... the pigs had to expend enormous labor every day upon mysterious things called files, reports, minutes and memoranda. These were large sheets of paper which had to be closely covered with writing, and as soon as they were covered, they were burnt in the furnace.....". Do you think SGML would have helped the pigs?

Comrade Snowball

**Figure 4. Office Memorandum**

The above example can be output from a word processor or typewriter and its tree structure can be illustrated as in figure 5.

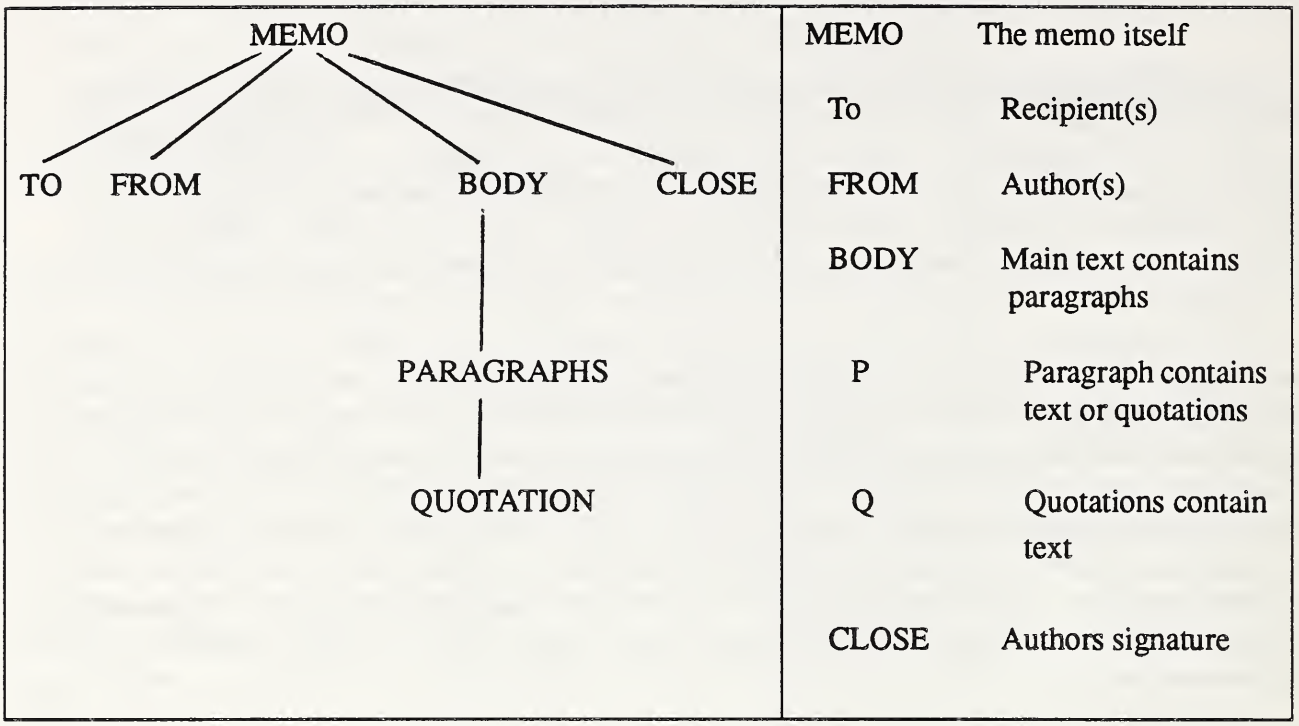



**Figure 5. Office Memorandum Tree Structure**

SGML can describe any data with a tree structure and is able to describe any tree representation, independent of hardware or software. The SGML document source of the memo is shown in figure 6.

```
<!DOCTYPE Memo SYSTEM "C:/MYDIR/MEMO3.DTD">
<Memo>
<To>Comrade Napoleon</To>
<From>Comrade Snowball</From>
<Body>
<Para>In Animal Farm, George Orwell says: <Q>.... the pigs had to expend enormous labor every
day upon mysterious things called files, reports, minutes and memoranda. These were large
sheets of paper which had to be closely covered with writing, and as soon as they were covered,
they were burnt in the furnace.....</Q>. Do you think SGML would have helped the pigs?
</Para>
</Body>
<Close>Comrade Snowball</Close>
</Memo>
```

**Figure 6. SGML Document Source**

In figure 6, the document content with the markup tags is displayed. The marked up form of the memorandum is called the document instance. The document instance contains the text of the document including a reference to the Document Type Declaration. The characters within the document instance are defined in the SGML declaration. The document is marked up according to the rules defined by the DTD.

The vocabulary of the parts of a tree representation is standardized through the use of a Document Type Definition (DTD). The DTD accompanies the marked up document wherever it goes so an SGML parser can analyze and check that the markup in the document satisfies the rules defined by the DTD. The Document Type Definition defines the structure and the rules for marking up the document instance and contains characters defined in the SGML declaration. It may also be stored externally to the document. After the document has been verified by the parser, it can be processed in many different ways.

The SGML parser is a program or a suite of programs that breaks down an SGML document into a series of logical elements and checks that these elements conform to the model defined in the associated Document Type Declaration. An SGML parser only checks the logical structure of the document's markup and reports any errors it finds. It does not correct errors, format the text or resolve the cross-references. The parser will expand the entity references within the document and remove the minimization by adding any omitted tags or replacing short tags by the full version. The SGML parser also:

1. identifies the markup tags and identifies any entities that need to be expanded or called from external sources;
2. checks the validity of the tags; and
3. checks to see if the tags are in the correct hierarchical order as defined by the DTD.

It should be remembered that SGML documents can be prepared without the aid of an SGML parser.

As mentioned in the section entitled, *"Description of SGML,"* before documents can be transmitted between systems, the numeric values must have clearly defined meanings. The rules defining the meanings of codes and reserved names used by a particular language are referred to as the syntax of the language. In SGML, there is the reference concrete syntax that is a set of formally defined rules within ISO 8879. Any system that conforms to ISO 8879 must be able to accept documents coded in this reference concrete syntax. Figure 7 [8] shows an SGML declaration suitable for a document prepared using the reference concrete syntax.

## 7. SGML APPLICATION OF THE ODA LOGICAL STRUCTURE

As the *Introduction* stated, the purpose of this task was to encode the ODA DAP in SGML, thereby illustrating the similarity of these two standards. In much of the previous sections, both standards were described in such a manner as to highlight the similarities between the two.

We first performed a document structure analysis on the ODA DAP. This step was far simpler than a normal rigorous document analysis since the logical structure of the ODA DAP was already

outlined in the DAP. With this simplication, we were able to proceed directly to the diagram of the logical tree structure displayed in figures 8 and 9.

At the top node the *logdoc* element followed by a series of layers containing different elements. This logical structure may be called a hierarchy of logical objects. This generic tree structure is remarkably similar to the very simple tree structure displayed previously in figure 5. The following is an example of a generic document logical structure derived from the ODA DAP:

Document
- Passage(s)
  - Paragraph
    - Text
    - Footnote
      - Footnote reference
      - Footnote body
    - Text
    - Figure
    - Text
  - Figure
  - Numbered Segment
    - Number
    - Title
    - Passage
      - Paragraph
      - Figure
  - Numbered Segment

To begin the SGML encoding of the ODA DAP, the top most node of the logical tree is identified. This element is called *logdoc*. For purposes of this encoding we will call this element *odadoc*. At the next level, there is a series of elements. On one branch, there is the element *title*, and on the other branch, there is a series of elements starting with *paragraph* and ending with *geometric*. This same series of elements is repeated almost in its entirety except for the substitution of the element *numbered segment* for the element *passage*. The top of figure 9 shows another series of elements beginning with *text* and ending with *phrase*. In studying the diagram, it becomes obvious that these series of elements are continually repeated. They are displayed as connecting links or branches "Y" and "U" hanging from a number of different elements. When a series of elements is continually repeated throughout the document, it may be replaced by a feature of SGML called an entity. To reduce the amount of repetitive keying within a document, SGML allows users to define *entities* that are required within the text. For example, a general entity called SGML could be used to enter the phrase "Standard General Markup Language." Entities are used for the following purposes:

```
<!SGML "ISO 8879-1986"
       -- Declaration for typical Basic SGML Document --
   CHARSET   BASESET   "ISO 646-1983//CHARSET International
                       Reference Version (IRV)//ESC 2/5 4/0"
             DESCSET   0   9  UNUSED
                       9   2  9
                       11  2  UNUSED
                       13  1  13
                       14  18 UNUSED
                       32  95 32
                       127 1  UNUSED
   CAPACITY  PUBLIC    "ISO 8879-1986//CAPACITY Reference//EN"
   SCOPE     DOCUMENT
   SYNTAX    SHUNCHAR  CONTROLS 0 1 2 3 4 5 6 7 8 9 10 11 12 13
                               14 15 16 17 18 19 20 21 22 23 24 25 26
                               27 28 29 30 31 127 255
             BASESET   "ISO 646-1983//CHARSET International
                       Reference Version (IRV)//ESC 2/5 4/0"
             DESCSET   0 128 0
             FUNCTION  RE           13
                       RS           10
                       SPACE        32
                       TAB  SEPCHAR  9
             NAMING    LCNMSTRT     ""
                       UCNMSTRT     ""
                       LCNMCHAR     "-."
                       UCNMCHAR     "-."
                       NAMECASE     GENERAL YES
                                    ENTITY  NO
                       GENERAL      SGMLREF
                       SHORTREF     SGMLREF
             NAMES     SGMLREF
             QUANTITY  SGMLREF
   FEATURES  MINIMIZE  DATATAG NO  OMITTAG  YES  RANK     NO  SHORTTAG YES
             LINK      SIMPLE  NO  IMPLICIT NO   EXPLICIT NO
             OTHER     CONCUR  NO  SUBDOC   NO   FORMAL   NO
   APPINFO   NONE
```

Figure 7. SGML Declaration

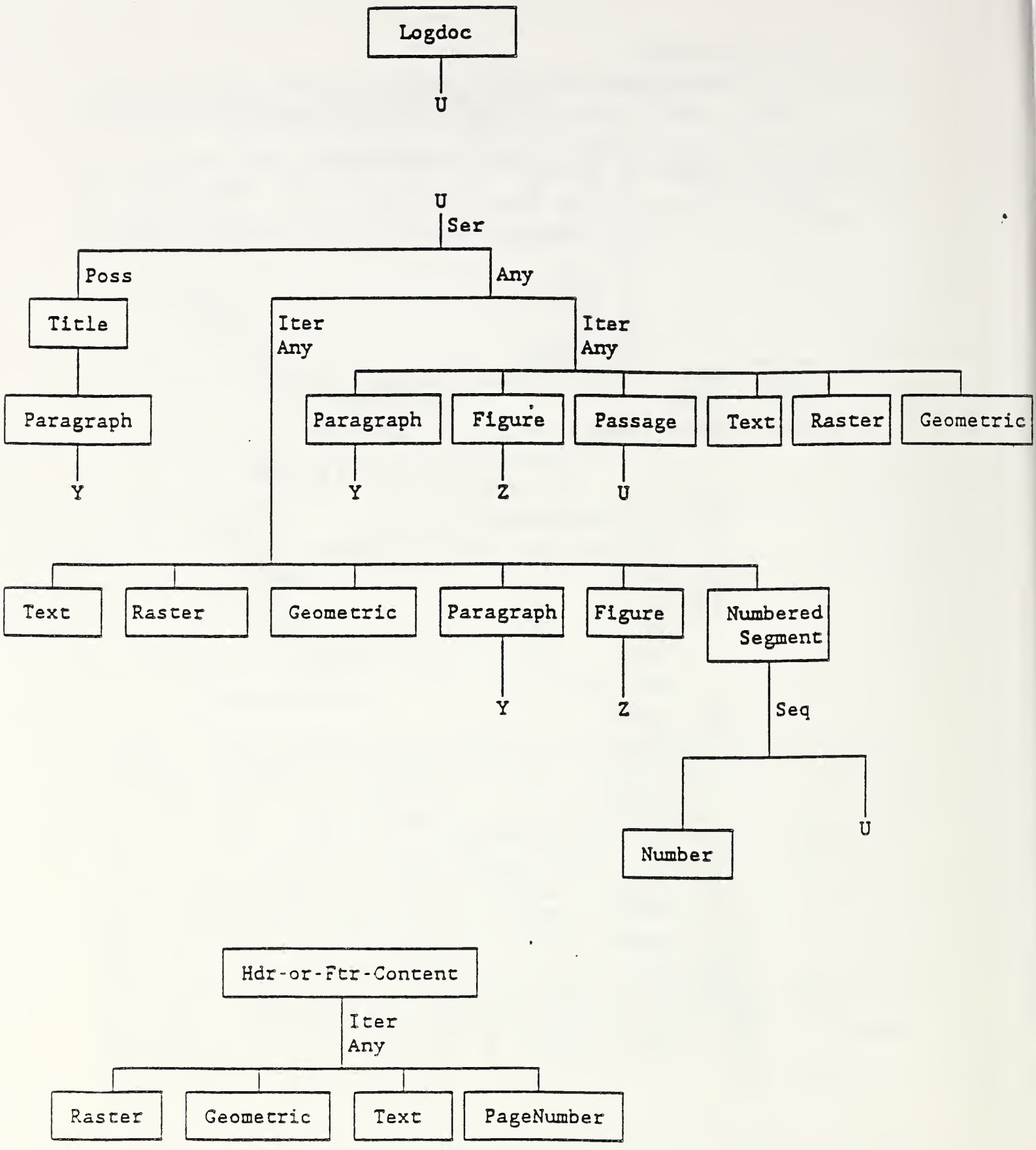



Figure 8. Diagram of a Logical Structure (1 of 2)

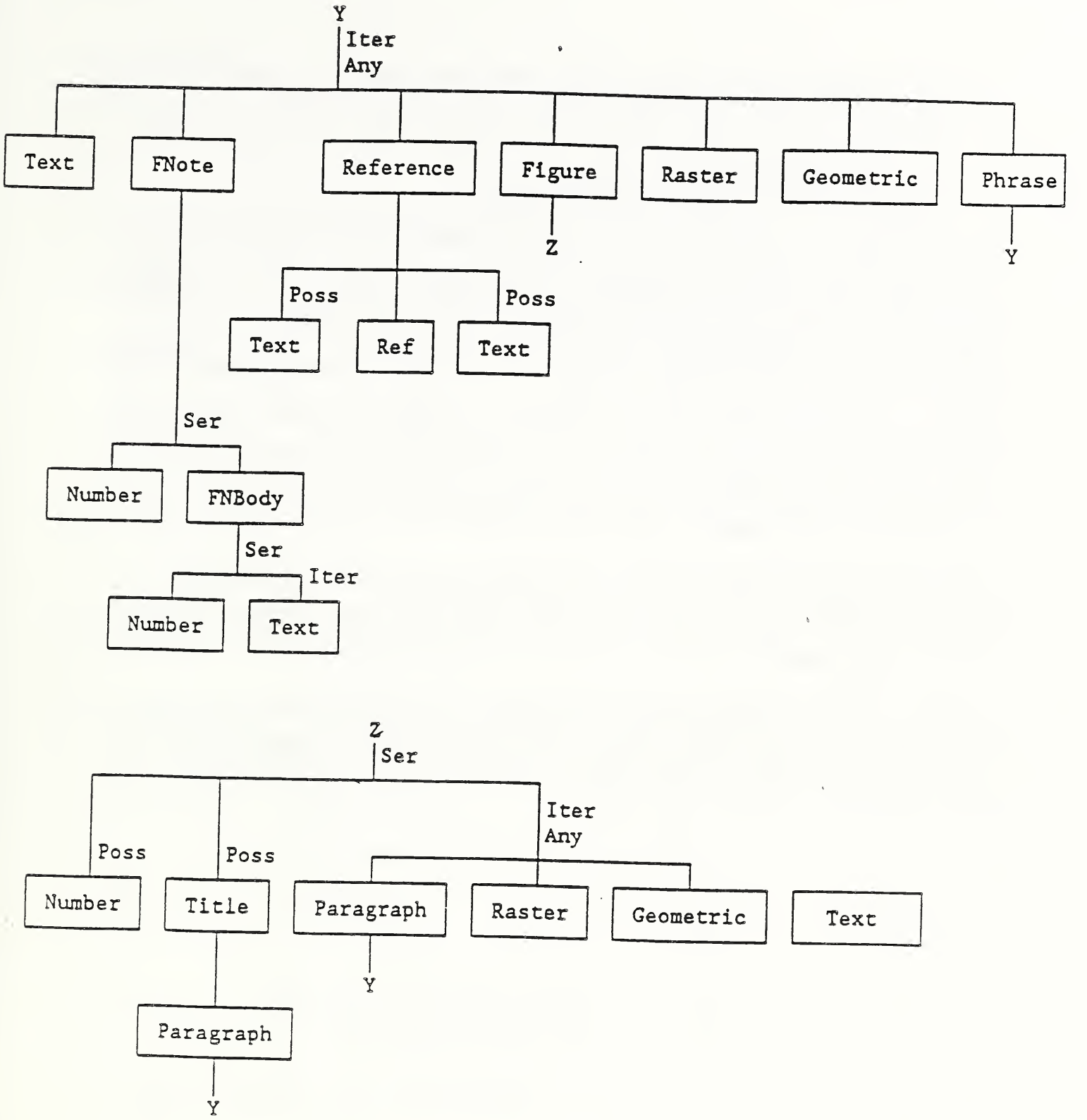






Figure 9. Diagram of a Logical Structure (2 of 2)

1. As a shorthand notation for long text strings or strings that are tedious to enter;
2. To provide a common way of using characters that do not exist as a key on the keyboard such as mathematical symbols; and
3. To include external documents in the main document.

This same feature may be used within the Document Type Definition. The DTD creator can define an entity that can be composed of a series of elements. In the future, whenever this series of elements is used, instead of rekeying the entire series, the entity name may be used. In our SGML encoding of ODA DAP displayed in figure 10, two entities are defined at the top of the page just under the SGML declaration. These two entities, *main* and *substrc*, are each defined as a particular list of elements. We have defined *main* as the main structure ("U") and *substrc* as the sub-structure ("Y"). Looking closely at both *main* and *substrc*, the identical elements are displayed within the diagram (fig. 8 and 9) defined in the branches "U" and "Y." Now when the DTD creator defines the top most element, *odadoc*, instead of listing all the elements of the "U" branches, only the element *title* and the entity *main* must be listed. The same convention is used for defining the elements *passage, numsegmt, para, and phrase*. The use of these two entities clearly shortens the DTD and makes it much easier to read.

It should also be noted that the DTD creator has shortened the names of some of the elements. To abide by the Reference Concrete Syntax, no element name can be longer than eight characters. The DTD creator could have changed the namelength characteristic within the SGML declaration to allow for longer names.

Another set of SGML and ODA conventions that appear to be quite similar are the set of SGML defined sequence connectors and occurrence indicators and the set of ODA defined operators. In SGML, there are three types sequence connectors and three types of occurrence indicators. The sequence connectors are listed in table 1 below:

| Default character | Delimiter name | Meaning |
|---|---|---|
| , | seq | All must occur, in the order specified |
| & | and | All must occur, in any order |
\| | or | One *and only one* must occur |

**TABLE 1. Sequence Connectors**

```
<!DOCTYPE odadoc [

<!--

<!DOCTYPE doc PUBLIC "-//USA-DOD//SGML ENCODED ODA LOGICAL DOCUMENT//EN" > -->

<!-- The following two entities refer to the main structure ("U") and a sub-structure ("Y") of the logical
document referred to on pages 1-36 & 1-37 of the ODA DAP  -->

<!ENTITY % main   "(((para | figure | numsegmt | text | raster | geometrc)* |
                        (para | figure | passage | text | raster | geometrc)*)*)" >

<!ENTITY % substrc   "((text | fnote | refrence | figure | raster | geometrc | phrase)*)"  >

<!ELEMENT odadoc - -   (title+, (%main;)*)* >

<!ELEMENT passage - -  (title+, (%main;)*)* >

<!ELEMENT numsegmt - - (number+, (%main;)*)* >

<!ELEMENT number - -   CDATA >

<!ELEMENT title  - -   (para) >

<!ELEMENT para   - -   (%substrc;)* >

<!ELEMENT phrase - -   (%substrc;)* >

<!ELEMENT fnote  - -   (number, fnbody)* >

<!ELEMENT fnbody - -   (number, text*)* >

<!ELEMENT figure - -   (number+, title+, (para | text | raster | geometrc))* >

<!ELEMENT text   - -   CDATA >

<!ELEMENT refrence - - (text+, ref, text+)* >

<!ELEMENT ref    - -   CDATA >

<!ELEMENT raster - -   EMPTY >

<!ATTLIST raster    boardno  ENTITY   #REQUIRED >

<!ELEMENT geometrc - - EMPTY >

<!ATTLIST geometrc   boardno  ENTITY   #REQUIRED >

<!ELEMENT hdftcntn - - (raster | geometrc | text | pgnumbr)* >

<!ELEMENT pgnumbr - -  CDATA >

]>
```

**Figure 10. SGML Encoding of the ODA DAP**

The use of an embedded subelement can be further qualified by the addition of an occurrence indicator. Three types of occurrence indicators are defined in SGML. They are illustrated in table 2.

| Indicator | Meaning |
|---|---|
| + | repeatable elements that must occur at least once at the current level |
| ? | optional elements that can occur ar most once at the current level |
| * | optional elements that may be repeated more than once at the current level |

**TABLE 2. Occurrence Indicators**

In ODA, there are a series of *operators* that perform tasks somewhat similar to the tasks performed by sequence connectors and occurrence indicators in SGML. These ODA operators are summarized in table 3 below:

| Indicator | Meaning |
|---|---|
| opt | optional construction factor specifying an object class identifier |
| rep | repetitive construction factor specifying an object class identifier |
| cho | choice construction factor specifying an object class identifier |
| seq | sequential construction factor specifying an object class identifier |
| agg | aggregate construction factor specifying an object class identifier |
| iter | may contain no construction expression or may contain sequence of objects belonging to various object classes. The object classes need be neither identical nor distinct. |
| poss | an empty sequence or an object belonging to a class. |
| ser | serial sequence of objects allowing only one instance of each. |
| any | may contain no construction expression or may be an object belonging to any class. |
| set | may contain no construction expression or may contain a sequence of objects. |

**TABLE 3. ODA Operators**

After studying these tables and analyzing the manner in which they are used in the various figures, it becomes obvious that there is a substantial similarity between the conventions of these two standards.

We have identified the top most element *odadoc* and have reduced the complexity of the DTD through the use of entities. We next look at some other elements listed in the DTD and examine their content in relation to the ODA diagram. We need not examine all the elements within the DTD here since many have similar constructions. We will, however, examine representative elements. Some elements contain various mixtures of one or two elements and an entity, while others are simpler. For instance, in looking at the top of figure 9 we see the element *FNote*. It is comprised of the elements *Number* and *FNBody*. In looking at the SGML encoding, we also see that the element called *fnote* has as its content two elements: *number* and *fnbody*. The element *fnbody* is composed of two additional elements, *number* and *text*, while the element *number* is a basic element. *Number* is the bottom most node on its branch and is composed of character text data. In SGML, an element composed of character text is represented as having a content of *CDATA*. If we look four lines above the element *fnote* in the SGML encoding, we see that the element *number* is in fact composed of *CDATA*.

Associated with the DTD of the ODA DAP are two tables. Table 4, **Baseline Tag Set Descriptions,** lists the element, its full name, and its description. The fifth table, **Alphabetic Listing of Elements,** lists the element name, its description, and other elements in which it is contained. The first table is used primarily as a summary listing of all elements within the DTD and their descriptions. The second table lists the elements and their descriptions, but the descriptions are not as complete as in table 4. This table is not primarily intentded to provide a description of the elements. Its primary function is to list each element and inform the reader as to which element or elements contain the element listed in the first column. Let's use the first element in the table as an example. In column one, the element *figure*, is displayed followed by its abbreviated description in column two. In column three, the elements *numsegmt, odadoc, para, passage,* and *phrase* are listed in alphabetical order. Examining the DTD, it becomes clear that the element *figure* is contained in each of the five elements in column three. At first glance this may not be obvious, since *figure* is contained in both entities *main* and *substrc*. One of these two entities is contained in all five of the elements in column three. To pick a simpler example, look at the element *ref*. We see from the description column that it is a general purpose reference mechanism. We also see that it is only contained in one element, *ref*. If we refer to the DTD, we see that ref only appears in a content model for the element, *refrence*.

## TABLE 4. Baseline Tag Set Descriptions

| Element/Attribute | Full Name | Description |
|---|---|---|
| FIGURE | Figure | An amount of geometric graphics or raster graphics content designed to occupy a rectangular area. |
| FNBODY | Footnote Body | Contiguous amount of text that can be read out of sequence from the paragraph containing a reference to it. |
| FNOTE | Footnote | Footnote reference and footnote body. |
| GEOMETRC | Geometric Graphic | Geometric Graphic(s) |
| HDFTCNTN | Header or Footer Content | Header or Footer Content |
| NUMBER | Number | Number |
| NUMSEGMT | Number Segment | Consists of either a number or a series of numbers separated by instances of an arbitrary specified character string separator. |
| ODADOC | ODA Document | Composed of a sequence of numbered segments or passages each of which is optionally titled and consists of a sequence of paragraphs and/or figures and/or further passages or numbered segments. |
| PARA | Paragraph | Contiguous amount of content in the intended reading order. |
| PASSAGE | Passage | Consists of any logical sequence of paragraphs, figures, and/or further passages or numbered segments that can be regarded as an entity for reading or for layout presentation. |
| PGNUMBR | Page Number | Page Number |
| PAGE NUMBER | Page Number | Page Number |
| PHRASE | Phrase | Phrase |
| RASTER | Raster Graphic | Raster Graphic(s) |
| REF | General Purpose Reference Mechanism | Consists of page numbers, chapter numbers, etc., provided within paragraphs. |
| REFRENCE | Reference | Reference |
| TEXT | Text | Text |
| TITLE | Title | Title |

## TABLE 5. Alphabetic Listing of Elements

| ELEMENT NAME | DESCRIPTION | CONTAINED IN ELEMENT(S) |
|---|---|---|
| figure | figure | numsegmt, odadoc, para, passage, phrase |
| fnbody | footnote body | fnote |
| fnote | footnote | para, phrase |
| geometrc | geometric graphic | figure, hdftcntn, numsegmt, odadoc, para, passage, phrase |
| number | number | figure, fnbody, fnote, numsegmt |
| numsegmt | number segment | numsegmt, odadoc, passage |
| para | paragraph | figure, numsegmt, odadoc, passage, title |
| passage | passage | numsegmt, odadoc, passage |
| pgnumbr | page number | hdftcntn |
| phrase | phrase | para, phrase |
| raster | raster graphic | figure, hdftcntn, numsegmt, odadoc, para, passage, phrase |
| ref | general purpose reference mechanism | refrence |
| refrence | reference | para, phrase |
| text | text | figure, fnbody, hdftcntn, numsegmt, odadoc, para, passage, phrase, refrence |
| title | title | figure, odadoc, passage |

## 8. CONCLUSION

Within various sections of this document, areas of similarity between ODA and SGML have been illustrated. Both standards define a document's logical structure as a hierarchy of objects. At the top of the hierarchy sits the ODA logical root or the SGML base document element. Below the top level there are any number of intermediate levels of ODA composite logical objects or SGML elements. At the bottom of the hierarchy are the ODA basic logical objects or SGML elements. Constraints between these objects are represented in ODA generic logical structures or SGML element declarations. The varied hierarchy of elements, whether it is composed of content or layout objects, has been illustrated in many figures throughout this paper. Structural relationships between document objects that cannot be captured by hierarchies may be denoted by ODA binding attributes or SGML ID references.

It has been discussed and demonstrated that both SGML and ODA may describe the data within a document in a tree structure. For SGML, this concept was explicitly illustrated within figure 5. For ODA, the concept appears again and again in many of this paper's figures (e.g., fig. 14 and 15).

ODA addresses various forms of content or mixed data types such as text, raster, and graphics. SGML, on the other hand, is primarily geared toward text; however, SGML offers a vehicle to include external files of text, raster or graphics. This vehicle is called an entity reference.

Tables one, two and three display the SGML sequence connectors and occurrence indicators and the ODA operators which perform the same relative tasks.

Layout is one area where the two standards differ the most. SGML is not concerned with the layout structure of the document. This function belongs to the application. ODA offers a set of objects to describe the layout and imaging of the contents.

The Output Specification, Appendix B of MIL-M-28001A and developed by the SGML community, addresses the presentation of a military technical manual. The Output Specification, developed by request of the CALS Project Office, describes a method for interchanging formatting requirements for military technical documents whose source files are tagged according to the Document Type Definitions developed in accordance with MIL-M-28001A. Within the Output Specification, there is a Formatting Output Specification Instance (FOSI) that interprets the style and formatting requirements of the military specification. The FOSI provides examples of how to use many concepts within the Output Specification. However, the FOSI is not complete in that it does not explicitly specify values for all elements within the DTD. It is mainly intended as a guideline for further FOSI development.

SGML will have, if successfully completed, a companion standard called Document Style Semantics and Specification Language (DSSSL) [9] that will address the layout structure and presentation of the document.

In the above paragraphs, we have continually seen a relative similarity exhibited between these two standards. Although they offer slightly different methods toward achieving the same goals, they view the logical document structure in similar fashion.

## 9. BIBLIOGRAPHY


[1] *ISO 8879: Information Processing - Text and Office Systems - Standard Generalized Markup Language (SGML),* International Organization for Standardization, 1986.

[2] *Document Application Profile (DAP) for the Office Document Architecture (ODA) and Interchange Format Standard,* National Institute of Standards and Technology, December 1988.

[3] *ISO 8613: Information Processing - Text and Office Systems - Office Document Architecture and Interchange Format (ODA).* International Organization for Standardization, 1989.

[4] *MIL-M-28001A - Technical Manuals: Markup Requirements and Generic Style Specification for Electronic Printed Output and Exchange of Text, July 1990.*

[5] *ISO 8824: Information Processing Systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1).* International Organization for Standardization, 1987.

[6] *ISO 646: Information Processing - 7-bit Coded Character Set for Information Interchange.* International Organization for Standardization, 1983.

[7] van Herwijnen, Eric, *Practical SGML,* Kluwer Academic Publishers, Norwell, MA., 1990.

[8] Bryan, Martin, *SGML, An Author's Guide to the Standard Generalized Markup Language,* Addison-Wesley Publishing Company, New York, 1988.

[9] "ISO work item number JTC1.18.15.06.01, Information Processing - Text Composition - Document Style Semantics and Specification Language (DSSSL)."

## 10. GLOSSARY

| | |
|---|---|
| abstract syntax | Rules that define how markup is added to the data of a document without regard to the specific characters used to represent the markup. |
| attribute (ODA) | An element of a constituent of a document that has a name and a value and that expresses a characteristic of this constituent or a relationship with one or more constituents. |
| attribute (SGML) | A characteristic quality of an element, other than type or content. |
| basic component | A basic logical or layout object, or an object class from which basic logical or layout objects may be derived. |
| basic layout object | An object in the specific layout structure that has no subordinate. |
| basic logical object | An object in the specific logical structure that has no subordinate. |
| block | A basic layout component that corresponds to a rectangular area within a frame or a page. |
| component | An object or an object class. |
| composite layout object | An object in the layout structure that has one or more subordinate objects. |
| composite logical object | An object in the logical structure that has one or more subordinate objects. |
| concrete syntax | A binding of the abstract syntax to particular delimiter characters, quantities, markup declaration names, etc. |
| content | The information conveyed by the document, other than the structural information, and is intended for human perception. |
| document application profile | The specification of a combination of features defined in ISO 8613, intended to form a subset to fulfill the requirements of an application. |
| document architecture | (1) Rules for defining the structure of documents, in terms of a set of components and content portions, and the representation of documents in terms of constituents and attributes.<br><br>(2) The structural information of a document consisting of the set of one or more of the following structures: specific logical structure, specific layout structure, generic logical structure, and generic layout structure. |

| | |
|---|---|
| document layout root | The composite object of the specific layout structure at the highest level of the hierarchy. |
| document type declaration | A markup declaration that contains the formal specification of a document type definition. |
| document type definition | Rules, determined by an application, that apply SGML to the markup of documents of a particular type. A document type definition includes a formal specification, expressed in a document type declaration, of the element types, element relationships and attributes, and references that can be represented by markup. It thereby defines the vocabulary of the markup for which SGML defines the syntax. |
| element | A component of the hierarchical structure defined by a document type definition; it is identified in a document instance by descriptive markup, usually a start-tag and an end-tag. |
| entity | A collection of characters that can be referenced as a unit. |
| formatted form | A form of representation of a document that allows the presentation of the document as intended by the originator and that does not support editing and (re)formatting. |
| formatted processable form | A form of representation of the document that allows presentation of the document as intended by the originator and also supports editing and (re)formatting. |
| frame | A type of composite layout component that corresponds to a rectangular area within a page or another frame. |
| generic layout structure | A set of layout object classes and associated generic content portions. |
| generic logical structure | A set of logical object classes and associated generic content portions. |
| logical object | An element of the specific logical structure of a document which may have a meaning that is significant to the application or user, for example, chapter, section, or paragraph. |
| logical structure | (1) The result of dividing and subdividing the content of a document into increasingly smaller parts, on the basis of the human-perceptible meaning of the content, for example, into chapters, sections, or paragraphs.<br><br>(2) All logical objects and associated content portions representing the logical hierarchy of a document. |

| | |
|---|---|
| object | An element of the specific layout structure or of the specific logical structure. |
| page | A layout component that corresponds to a rectangular area used for presenting the content of the document. |
| page set | A composite layout component that represents a collection of pages or further page sets. |
| reference concrete syntax | A concrete syntax, defined in the international SGML standard, that is used in all SGML declarations. |

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 4547

2. PERFORMING ORGANIZATION REPORT NUMBER

3. PUBLICATION DATE
APRIL 1991

4. TITLE AND SUBTITLE

A Standard Generalized Markup Language Encoding of the Office Document Architecture Document Application Profile for the Computer-aided Acquistion and Logistic Support.

5. AUTHOR(S)

Ronald B. Wilson

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.)

The Office Systems Engineering Group (OSE) at the National Institute of Standards and Technology (NIST) was tasked by the CALS' Project Office to bring CALS' requirements to the Open Systems Interconnection (OSI) Implementor's Workshop sponsored by NIST. CALS tasked the OSE Group to assist in the SGML encoding of the Office Document Architecture (ODS) Document Application Profile (DAP). NIST offered to encode the ODA DAP in the Standard Generalized Markup Language (SGML) to illustrate the similarities between the two standards and to provide a common SGML/ASN.1 profile.

The report describes in various levels of detail the two international standards. It then discusses, by offering a simple example, the methodology involved in performing an SGML encoding. Subsequently, it examines the SGML encoding of the ODA DAP. The report provides two accompanying tables with the SGML encoding and finishes with a brief summary of the two standards.

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

architecture; CALS; document application profile; document type declaration; generic logical structure; Implementor's Workshop; logical object; ODA; SGML

13. AVAILABILITY

[X] 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.

[X] ORDER FROM NATIONAL TECHNICAL INFORMATION SERVICE (NTIS), SPRINGFIELD, VA 22161.

14. NUMBER OF PRINTED PAGES
33

15. PRICE
A03

ELECTRONIC FORM