NOT MEASUREMENT SENSITIVE

MIL-D-87269
20 November 1992

MILITARY SPECIFICATION

DATA BASE, REVISABLE: INTERACTIVE ELECTRONIC TECHNICAL MANUALS, FOR THE SUPPORT OF

This specification is approved for use by all Departments and Agencies of the Department of Defense.

1. SCOPE.

1.1 Scope. This specification prescribes the requirements for an Interactive Electronic Technical Manual Data Base (IETMDB) to be constructed by a weapon-system contractor for the purpose of creating Interactive Electronic Technical Manuals (IETM). The requirements herein cover the specification for the IETMDB and are intended to apply to one or both of two modes as specified in a contract: (1) the interchange format for the data base to be delivered to the Government; or (2) the structure and the naming of the elements of the data base created and maintained by the contractor for purposes of creating IETMs which are in turn delivered to the Government.

1.2 Paragraphs with limited applicability. This specification contains paragraphs and specific requirements which are applicable to all Services. Such paragraphs or requirements are prefixed to indicate the Services to which they pertain: (A) for Army; (N) for Navy; (M) for Marines; and (F) for Air Force.

Beneficial comments (recommendations, additions, deletions) and any pertinent data which may be of use in improving this specification should be addressed to: HQ AFMC/ENCS, Wright-Patterson AFB, OH 45433-5001; by using the self addressed Standardization Document Improvement Proposal (DD Form 1426) appearing at the end of this document, or by letter.

AMSC F6847 **AREA TMSS**

Distribution Statement A. Approved for public release; distribution is unlimited.

1.3 Introduction. The following introductory material is included in this specification in order to fully acquaint offering contractors with the new requirements and characteristics of a data base developed specifically for IETMs.

1.3.1 Nature and purpose of a revisable source data base. For complex weapon systems and other types of military equipment, adequate logistic support in all its forms requires an enormous amount of current, readily accessible, accurate, and highly detailed data, consisting of Technical information (TI). This information has been traditionally prepared and distributed to the end user in paper form; but with new technology it can be better and more effectively displayed or presented electronically and interactively to an end user. The material presented is derived from material stored in textual, graphical, audio, or video form in a revisable data base which is composed of logically connected but randomly accessible IETM data elements. It is this starting point of the IETM electronic data chain that is specified in this document. The concept that the Services can either acquire and maintain large scale data bases of this type, or acquire access to such data bases, maintained continuously by a contractor, is an integral part of the IETM concept and in the larger arena of the Department of Defense (DoD) Computer-aided Acquisition and Logistic Support (CALS) program.

1.3.1.1 IETMDB capabilities. An IETMDB is a complete collection of data base elements relating to a weapon system or other equipment acquired by the Government and constructed in a standardized manner in order to provide the following capabilities:

a. The IETMDB can serve as the basis for construction and update of the entire suite of weapon system electronically displayed IETMs through the use of automated authoring systems.

b. Government activities or DoD contractors concerned with logistic support for the weapon system involved can access the data base directly to obtain needed logistic support information for specific purposes.

c. The IETMDB, or portions of it, can be interchanged by means of standardized formats and procedures throughout the DoD and its supporting contractors when needed for any purpose.

1.3.2 Technical information procurement options. Acquisition of IETMs may be carried out by one of several optional approaches. This specification provides requirements for a standardized IETMDB which will permit the Government to acquire TI by applying any of the following contractual options:

a. Acquisition of only the final form IETMs which are required. Although the author (equipment prime contractor) will need to establish an automated equipment or weapon-system (source) data base, this data base will not be acquired by the Government, but will be maintained, used, and controlled by the contractor, both for the preparation of IETMs and for other purposes. The Government under this specification requires that the data

base be structured and the individual data elements named and attributed in a standard manner. However, an explicitly tagged data file need not be prepared for delivery as no data base delivery is required.

b. Acquisition of the IETMDB. Acquisition of the IETMDB may involve either of the following options:

(1) Delivery to the Government in standardized form and subsequently maintained by the Government (with or without update information supplied on a continuing basis by the contractor).

(2) Title acquired to the IETMDB by the Government, but with the data base retained and maintained in standardized form in the contractor's plant. The Government could be provided with on line access to the data base.

c. Acquisition of fully constructed IETMs (fully prepared and validated by the contractor), as well as the IETMDB upon which they are based. Acquisition under this option may involve either option (1) or (2) as given in 1.3.2b above.

1.4 Format free technical information. The IETMDB will consist of an assemblage of data elements, including a listing of the specific attributes possessed by the data elements; and a list of explicit relationships providing logical links among the data elements. The relationships incorporated into the data base by the IETMDB author provide the basis of the technical structure of the IETMs and other logistic support TI which will be extracted from it. The IETMDB will not, however, contain format directions in the sense of arrangement of text and graphics on a display screen for presentation to the end user. The IETMDB itself will, of course, require a "format" (data base structure) but this specification does not impose structural requirements on the actual Data Base Management System (DBMS) methodology to be employed (e.g., the data base may be either relational or object oriented). The exterior view of the data base to be used for updating, adding cross references, producing tagged output files, etc. must conform to requirements of this specification.

1.4.1 Data portability. Elimination of formatting requirements for the IETMDB reduces the overall magnitude of data base and data interchange standardization effort, and permits use of a less complex DBMS by the contractor which is, in turn, less expensive and easier to modify. The "format-free" nature of the IETMDB is intended to provide the Government the capability to:

a. Acquire or access the data in a variety of ways (IETMs, other types of logistics reports; training TI, etc.);

b. Subsequently format and style the data in a variety of ways for electronic display options.

1.4.2 Integration support. Since one of the functions of the IETMDB is to provide direct on line data access to a variety of users and to a number of automated logistic support and management information systems throughout the Services, establishment of standard identifiers, data entity relationships, and multiple path access routes to individual data elements is an important part of IETMDB design and construction.

2. APPLICABLE DOCUMENTS.

2.1 Government documents.

2.1.1 Specifications, standards and handbooks. The following specifications, standards and handbooks form a part of this document to the extent specified herein. Unless otherwise specified, the issues of these documents are those listed in the issue of the Department of Defense Index of Specifications and Standards (DODISS) and supplement thereto, cited in the solicitation (see 6.2).

SPECIFICATIONS

Military

MIL-M-87268	-	Manual, Technical: General Content, Style, Format, and User Requirements for Interactive Electronic Technical Manuals
MIL-Q-87270	-	Quality Assurance Program: Interactive Electronic Technical Manuals and Associated Technical Information; Requirements for

STANDARDS

Military

MIL-STD-1388-1	-	Logistics Support Analysis
MIL-STD-1388-2	-	Logistics Support Analysis Record, DoD Requirements for a
MIL-STD-1840	-	Automated Interchange of Technical Information

(Unless otherwise indicated, copies of federal and military specifications, standards and handbooks are available from the Standardization Documents Order Desk, Building 4D, 700 Robbins Avenue, Philadelphia, PA 19111-5094.)

2.1.2 Other Government documents, drawings and publications. The following other Government documents, drawings and publications form a part of this specification to the extent specified herein.

Unless otherwise specified, the issues are those cited in the solicitation.

PUBLICATIONS

Department of Defense

DOD 5200.1-R	-	Information Security Program Regulations
DOD 5220.22-M	-	Industrial Security Manual for Safeguarding Classified Information

(Application for copies should be addressed to the Superintendent of Documents, US Government Printing Office, Washington, DC 20402.)

2.2 Non-Government publications. The following documents form a part of this document to the extent specified herein. The issues of the documents which are DoD adopted are those listed in the issue of the DODISS cited in the solicitation. Unless otherwise specified, the issues of documents not listed in the DODISS are the issues of the documents cited in the solicitation.

ISO 8879	-	Information Processing - Text and Office Systems - Standard Generalized Markup Language (SGML)
ISO/IEC IS10744	-	Information technology - Hypermedia/Time-based Structuring Language (HyTime)

(Application for copies should be addressed to the American National Standards Institute, 1430 Broadway, New York, NY 10018.)

2.3 Order of precedence. In the event of a conflict between the text of this document and the references cited herein, the text of this document takes precedence. Nothing in this document, however, supersedes applicable laws and regulations unless a specific exemption has been obtained.

3. REQUIREMENTS.

3.1 General requirements. An IETMDB developed in accordance with this specification shall conform to the Content Data Model (CDM) specified herein. The CDM employs a two layered approach to define technical information. The top layer, called the "Generic Layer", defines the semantic rules for the data characteristics. The generic layer is defined in Appendices A and C of this specification. The bottom layer, called the "Content Specific Layer", employs the generic layer when defining elements for weapon system specific TI. Appendices B and D contain a content specific layer model developed for organizational level maintenance. Many content specific layers can be developed in accordance with the generic layer. The CDM generic layer defined in Appendices A and C of this specification will be the DoD standard for any data base of IETM technical information procured using this specification. In addition, unless otherwise specified by the procuring activity, the content specific

layer defined in Appendices B and D of this specification shall also be part of this specification (see 6.2). If Appendices B and D of this specification are not specified by the procuring activity, some other content specific layer Document Type Definition (DTD) must be specified and approved by the government. The IETMDB can be invoked by a procuring activity in either one of two modes as follows, depending on whether a data base is (1) specified for interchange and delivery to the Government, or (2) being developed and maintained for the subsequent preparation of IETMs, but not actually delivered to the Government.

3.1.1 Data Base interchange requirements. When specified, IETMDBs which are to be delivered to the Government under this specification shall be structured and tagged in accordance with the DTDs and the tag set descriptions included as Appendices A through D of this specification (see 6.2).

3.1.2 Data base structuring and data element naming requirements. Unless otherwise specified, a deliverable instance created under this specification shall be structured in accordance with the hierarchical relationships defined in the CDM DTDs contained in Appendices A and B, and created and named in accordance with the tag set descriptions contained in Appendices C and D (see 6.2). When a tagged instance is not specified for delivery, the contractor shall maintain the ability to map the internal element names to the specified content specific DTD names.

3.1.3 Data maintainability. The IETMDB shall be constructed with provisions to allow the incorporation of any change to automatically update all aspects of the data base affected by that change. This data-maintainability requirement shall involve changes to the IETMDB of the following two kinds:

a. Additions to, eliminations of, or changes to individual data elements and attributes;

b. Changes to relationships including establishment of new relationships or elimination of old relationships.

3.1.4 Additional content specific DTDs. When specified, additional content specific DTDs shall be used in addition to or instead of the content specific DTD defined in Appendices B and D of this specification (see 6.2). These DTDs shall be incorporated into the overall CDM in accordance with the requirements of 3.2.

3.2 Generic layer. The generic layer of the CDM is defined in the DTD listed in Appendix A. This DTD provides templates, which shall be used to define content specific elements. The generic layer includes a definition for each template and the attribute lists associated with the template. The DTD provides a definition of three other data types: primitive data elements that shall remain standard across all content specific applications; user interaction elements, called dialogs; and the context filtering elements, which shall be used to provide the most appropriate information to a user. The following paragraphs provide a description of the components of the generic layer.

3.2.1 Templates. Templates shall be used as described in Appendix A to define elements declared in content specific DTDs. The generic layer contains five templates: Node, Node Alternatives, Node Sequence, If Node, and Loop Node. Each template has two components: (1) a set of semantic rules that govern the template's activities, and (2) a list of attributes.

3.2.1.1 Node template. All elements conforming to the node template provide the capability for creating composite structures within the content specific layer. Composite structures contain primitives, links, and preconditions. When a composite structure contains other composite structures within its content model, this implies hierarchy. Elements employing the node template shall have a set of required attributes as follows.

3.2.1.1.1 Name. The "name" attribute of the element shall consist of the standard nomenclature for an instance of the element.

3.2.1.1.2 Item-Id. The "Item-Id" attribute shall specify the reference designator(s) and other identifiable designator(s) of the system(s), subassemblies, or part(s) referred to by the element.

3.2.1.1.3 Type. The "type" attribute shall specify the type of information contained in the element.

3.2.1.1.4 CDM. The "cdm" attribute shall identify the type of template being employed by the content specific element.

3.2.1.1.5 Ref. The "ref" attribute shall facilitate the reduction of data redundancy by allowing data elements to be referenced.

3.2.1.2 Node Alternatives (Alts) template. All elements conforming to the node alts template shall contain a list of mutually exclusive nodes, only one of which will be used at the time of presentation.

3.2.1.3 Node Sequence (Seq) template. All elements conforming to the node seq template group elements together as well as providing an order or presentation sequence to the elements. The elements conforming to the Node Seq shall also allow an author to define branching logic within the TI.

3.2.1.4 If node template. Elements conforming to the if node template provide a method for conditional branching. These elements shall use the same logic as the IF-THEN-ELSE statement in a programming language. The "IF" part is the expression in the content model. The "THEN" part is the first node seq, and is selected when the expression evaluates to true. The "ELSE" part is the second node seq, which is optional in the CDM, and is selected when the expression evaluates to not true.

3.2.1.5 Loop node template. The loop node template provides the equivalent of a loop in a programming language. This element shall provide the capability to create either a "FOR" loop or a "WHILE" loop within the data. The expressions and assertions developed in accordance with this template provide the testing criteria for the loop. The node sequence shall contain the actual elements to be repeated within the loop.

3.2.2 Relational links. Elements shall have relationships to other elements in TI, when applicable. These relationships shall be represented through two or more link ends. The link element shall provide the capability to show the relationship between several elements. The contractor shall include the specific cross references to elements within the IETMDB as well as information sources outside the IETMDB.

3.2.2.1 Links to reduce redundancy. Links shall be used to reduce the number of redundant elements by referencing common elements. The templates defined within the generic layer CDM DTD define attributes to reduce redundant elements. These elements shall utilize the Standard Generalized Markup Language (SGML) #CONREF reference capability in accordance with International Standards Organization (ISO) 8879. A #CONREF attribute, used only in instances where an elements content model is EMPTY, shall contain the unique identifier of an element using a template or a location element.

3.2.2.2 Location elements. The location elements are defined by International Standards Organization/International Electrotechnical Commission (ISO/IEC) IS10744. Elements shall be referenced by other elements in accordance with ISO/IEC IS10744.

3.2.2.3 Logistics support and task-analysis link. The contractor shall establish linkages (information-access capabilities) with the IETMDB when external logistics support and task analysis systems developed in accordance with MIL-STD-1388-1 and MIL-STD-1388-2 exist.

3.2.3 Primitive elements. An IETMDB shall be composed of the primitive elements defined in the generic layer DTD as follows. Content and style for these elements shall be in conformance with the requirements of MIL-M-87268.

3.2.3.1 Textual information. Textual information shall consist of alphanumeric (i.e., character) data. Textual information shall also contain embedded references to some higher level elements such as those describing parts or consumables.

3.2.3.2 Tables. Tables shall be represented as a series of separate entries, each entry being associated with a specific row and column intersection (cell) of a table. Each entry in the table may be associated with other primitive types of information presentation and attributes. Each entry may refer (through a relationship) to any other template element or primitive element in the IETMDB.

3.2.3.3 Graphics. Graphics (drawings, illustrations) information shall be structured in a hierarchical manner and consist of logically related groups. Graphics shall be composed of a series of illustrations which can be overlaid on each other to build a complete graphic. These graphic "building blocks" are called graphic primitives. Graphic primitives may be combined to produce composite information which can be referenced and selected. Graphics shall be composed of information represented in accordance with the graphic standards included in MIL-STD-1840.

3.2.3.4 Audio, video, and process. The elements audio, video, and process provide the capability for the author to define an audio sequence, a video sequence, or a call to a software process.

3.2.3.5 Dialogs. Dialog elements are the basic element which provides the capability for user interaction with the TI. During a presentation these elements shall be used to prompt the user to input a response ("fillin"), select a choice from a set of alternatives ("menu"), or to select items from within a text, table or graphic ("selection").

3.2.4 Context dependent filtering. Context dependent filtering shall be accomplished through author-defined preconditions. A precondition shall contain an expression which will contain all the information necessary to identify what conditions must be present to display the TI.

3.2.4.1 Preconditions. A precondition shall contain an expression which identifies the conditions which must be present to display the TI. Precondition elements may be referenced by node elements. This implies that the element's information is relevant only if the precondition is true in the presentation situation.

3.2.4.2 Postconditions. Postconditions assert the value of an expression to a property. Once these property values are asserted, they shall be accessible to the presentation software for later testing and processing to determine the user's situation.

3.2.4.3 Expressions. Any expressions developed for an IETMDB shall conform to one of four types of expressions defined in the CDM. The first is a binary operation between two expressions; the second is a unary operation which is applied to an expression; the third and fourth are operations that identify an unique property (variable) or a value to be used in an expression.

3.3 Content specific layer. All TI shall be structured in accordance with a content specific DTD. One content specific DTD shall apply for an entire set of information regardless of the desired access to the information. The CDM will define the content and structure of the TI but will not describe format information.

3.3.1 Control of content specific DTDs. The contractor shall not exchange TI with the DoD unless it has been developed in accordance with the generic layer DTD and one or more of the latest versions of DoD approved content specific DTDs. If a content specific DTD does not exist which meets the contract's requirements, the contractor shall submit a content specific DTD to the Government for approval.

3.3.2 Development of content specific DTDs. If a new content specific DTD is developed, the contractor shall assure the content specific DTD meets the requirements of ISO 8879, and the requirements imposed by the generic layer DTD.

3.3.2.1 Use of generic DTD primitive elements. The generic layer of the CDM shall define a set of primitive elements. Those elements shall be available to any content specific layer DTD that includes the generic layer in an entity declaration. Any element defined within a content specific DTD which requires the use of any of the primitive elements need only include text, table, graphic, or dialog within its content model. The contractor shall not redefine the primitive elements within the content specific DTD. The elements using the primitive elements shall be restricted to the structure of the primitive elements as defined within the generic layer.

3.3.2.2 Use of generic DTD template elements. Elements within a content specific DTD shall conform to one of the templates defined within the generic layer. Elements shall include the attributes listed under the generic layer's definition of the templates. The two common attributes among the five templates are identification (id) and content data model (cdm). Each element employing a template

includes an identification attribute for referencing. The cdm attribute identifies which template an element is employing.

3.3.3 Content specific DTD for Organizational Level (O-Level) maintenance. The following describes requirements for the content specific DTD included in this specification.

3.3.3.1 Item/System hierarchy. The vehicle, weapon system, or other equipment that is being maintained and operated is composed of several layers of subsystems, components, and parts. This hierarchical representation of the equipment being maintained and operated shall be accomplished by use of a system element that is used recursively, and which decomposes the equipment into only those components that are being maintained or operated. Each component of this hierarchy has associated with it one or more of the following four categories of information:

a. Descriptive information

b. Procedural information

c. Troubleshooting information

d. Parts information

3.3.3.2 Descriptive information. Descriptive information may provide information on system (subsystem, component, part) physical arrangement, functional behavior, theory of operation, and other aspects. Descriptive information shall contain a hierarchy of narrative paragraphs. Paragraphs, in turn, may refer to primitive elements.

3.3.3.3 Procedural information. Procedural information shall be composed primarily of task statements. Each task element shall be associated with attributes which provide related information such as: estimated completion time; maintenance level(s) where the task is to be performed; required conditions which must be met before performing the task; and the number of people required to perform the task. A procedural element may be linked to other elements which define the support equipment and consumables that task requires, through the establishment of appropriate relationships.

3.3.3.4 Troubleshooting information. Troubleshooting information shall contain data necessary to isolate faults found in a system. Troubleshooting information shall contain fault elements, fault state elements, test elements, outcome elements, and rectification elements.

3.3.3.4.1 Fault elements. Fault elements shall identify potential faults which might occur in the system.

3.3.3.4.2 Fault state elements. Fault state elements shall present a list of faults implicated as the result of a test that has been performed. Each suspected fault in the list shall be weighted, based on the probability that it is the cause of the observed malfunction. The fault state element may also present a list of possible faults that have been eliminated from consideration as the result of tests performed.

3.3.3.4.3 Test elements. Test elements shall contain a link to the procedural instructions a technician must follow to carry out a required task at a particular juncture in the troubleshooting procedure. Test elements shall also provide all possible test outcomes.

3.3.3.4.4 Outcome elements. Outcome elements shall contain definitions of new fault states associated with the results of a particular test. Outcome elements shall also contain a description of the state of the item being maintained. An outcome is based on one or more expressions (i.e., system states which must be established for the specific outcome to apply). The final outcome element of a fault isolation procedure shall have a relationship which associates it with an identified fault. The identified fault has, in turn, associated with it the initial element of the appropriate corrective maintenance action.

3.3.3.4.5 Rectification elements. Rectification (i.e., corrective maintenance actions) elements shall contain references to procedural rectification tasks, checkout tests used to report the success of completed rectification tasks, and a list of all faults that the rectification shall repair.

3.3.3.5 Parts information. Two types of parts information shall be included: (1) maintainer/operator information, and (2) supply information. Elements containing either type shall refer explicitly to corresponding elements of the other type.

3.3.3.5.1 Parts information for the maintainer or operator. Parts information provided for a system maintainer and/or operator shall include such items as units per assembly, usable-on code, Mean Time Between Failures (MTBF), and reference designator, if applicable.

3.3.3.5.2 Parts information provided for parts supply. Parts information provided for the parts supply process shall constitute unambiguous identification of a part so that it can be reordered, and may consist of such items as: the part number; Commercial And Government Entity (CAGE) code; Source, Maintenance, and Recoverability (SMR) code; Hardness Critical Item (HCI) identification; and National Stock Number (NSN), if applicable.

4. QUALITY ASSURANCE PROVISIONS.

4.1 Responsibility for inspection. The contractor is responsible for the performance of all inspection requirements (examinations and tests) as specified herein. Quality Assurance (QA) for the IETMDB preparation shall be in accordance with the requirements of MIL-Q-87270. Except as otherwise specified in the contract or purchase order, the contractor may use his own or any other facilities suitable for the performance of the inspection requirements specified herein, unless disapproved by the Government. The Government reserves the right to perform any of the inspections set forth in this specification where such inspections are deemed necessary to ensure supplies and services conform to prescribed requirements.

4.1.1 Responsibility for compliance. All items shall meet all requirements of sections 3 and 5 of this specification. The inspection set forth in this specification shall become a part of the contractor's overall inspection system or quality program. The absence of any inspection requirements in this specification shall not relieve the contractor of the responsibility of ensuring that all products or supplies submitted to the Government for acceptance comply with all requirements of the contract.

Use of sampling for inspections shall be in accordance with MIL-Q-87270; however, Government approval for use of sampling in QA procedures does not authorize submission of known defective material, either indicated or actual, nor does it commit the Government to accept defective material.

5. PACKAGING.

5.1 Preparation for delivery. Items shall be packaged in the most economical manner that will provide adequate protection during shipment in accordance with accepted industrial packaging procedures.

5.1.1 Digital product packaging. Packaging of encoded computer products, in preparation for delivery, shall be in accordance with the requirements of MIL-STD-1840.

5.1.2 Classified material. Classified material shall be packaged and identified in accordance with DOD 5200.1-R and DOD 5220.22-M.

6. NOTES.

(This section contains information of a general or explanatory nature that may be helpful, but is not mandatory.)

6.1 Intended use. An IETMDB is the source data for the preparation of IETMs. IETMs prepared in accordance with this specification are intended for use in the installation, operation, maintenance, repair, and logistics support of equipment/systems or for the accomplishment of the assigned mission of users.

6.2 Acquisition requirements. Acquisition documents must specify the following:

a. Title, number and date of the specification.

b. Issue of the DODISS to be cited in the solicitation, and if required, the specific issue of individual documents referenced (see 2.1.1, 2.1.2).

c. If IETM program elements shall be other than as specified herein (3.1).

d. If the specification applies to the delivery and tagging of an IETMDB (3.1.1).

e. If the specification applies to the structuring of the IETMDB and naming of the IETMDB elements which are created and maintained by the contractor (3.1.2).

f. Content specific DTDs other than the one included herein and whether these are to be used in addition to or instead of the content specific DTD included herein, if additional DTDs apply (3.1.4).

6.3 Data requirements. Any applicable Data Item Descriptions (DIDs) must be listed on the Contract Data Requirements List (CDRL) when this specification is applied on a contract, in order to obtain the data, except where DOD FAR Supplement 27.475-1 exempts the requirement for a DD Form 1423. DID citations shall be as follows:

Reference Paragraph	DID Number	DID Title	Suggested Tailoring
(None Required)			

6.4 Definitions of acronyms and terms. Acronyms and IETM terms not listed in MIL-STD-12 are included in the definitions contained in 6.4.1 through 6.4.3.

6.4.1 Acronyms.

ALHRD	Armstrong Laboratory Human Resource Division
AFMC	Air Force Materiel Command
AMSC	Acquisition Management System Control (number)
ANSI	American National Standards Institute
ATE	Automatic Test Equipment
BL	Buttock Line
CAGE	Commercial And Government Entity
CALS	Computer-aided Acquisition and Logistics Support
CDM	Content Data Model
CDRL	Contract Data Requirements List
DBMS	Data Base Management System
DD	Department of Defense (document-number prefix)
DID	Data Item Description
DLA	Defense Logistics Agency
DLSC	Defence Logistics Service Center
DoD	Department of Defense
DODISS	Department of Defense Index of Specifications and Standards
DTD	Document Type Definition
DTRC	David Taylor Research Center
EDS	Electronic Display System
FS	Fuselage Station
FSC	Federal Stovk Classification
HCI	Hardness Critical Item
ICC	Item Category Code
IEC	International Electrotechnical Commission
IETM	Interactive Electronic Technical Manual
IETMDB	IETM Data Base
IETMQA	IETM Quality Assurance (program)
ISO	International Standards Organization
LRU	Line Replaceable Unit
MTBF	Mean Time Between Failures

NATO	North Atlantic Treaty Organization
NSN	National Stock Number
QA	Quality Assurance
QAP	Quality Assurance Program
SGML	Standard Generalized Markup Language
SMR	Source, Maintenance, and Recoverability (Code)
STD	Standard
TI	Technical Information
TMSS	Technical Manual Specifications and Standards (standardization program)
WL	Water Line
WS	Wing Station

6.4.2 Interactive Electronic Technical Manual (IETM). A technical manual, prepared (authored) by a contractor and delivered to the Government, or prepared by a Government activity, in digital form on a suitable medium, by means of an automated authoring system; designed for electronic screen display to an end user, and possessing the following three characteristics:

a. The format and style of the presented information are optimized for screen presentation to assure maximum comprehension; that is, the presentation format is "information oriented", not "page oriented".

b. The elements of technical data constituting the IETM are so interrelated that a user's access to the information required is facilitated to the greatest extent possible, and is achievable by a variety of paths.

c. The computer controlled IETM display device can function interactively (as a result of user requests and information input) in providing procedural guidance, navigational directions, and supplemental information; and also in providing assistance in carrying out logistic support functions supplemental to maintenance.

6.4.3 Quality assurance. A planned and systematic series of procedures carried out by the contractor as monitored by the Government, necessary to ensure a high degree of confidence that deliverable technical data conforms to contract requirements. The culminating action in a Quality Assurance Program (QAP) is contractor support of the Government verification procedure.

6.5 Subject terms (key word) list.

Database
Interactive Electronic Technical Manual (IETM)
Content Data Model (CDM)
Technical Manuals

Custodian:
Air Force - 16
Army - TM
Navy - AS

Preparing Activity:
Air Force - 16

Review Activities:
Air Force - 11, 13, 14, 18, 19, 30, 70, 71, 80, 82, 84, 99
Army - AL, AR, AT, AV, CR, EA, MI, PT, SC, TR
Navy - AS, EC, MC, SA, SN, TD, YD

(Project TMSS-0296)

User Activities:
Air Force - 11, 13, 14, 18, 19, 30, 70, 71, 80, 82, 84, 99
Army - AL, AR, AT, AV, CR, EA, MI, PT, SC, TR
Navy - AS, EC, MC, SA, SN, TD, YD

GENERIC LAYER
DOCUMENT TYPE DEFINITION (DTD)

10. SCOPE.

10.1 Scope. The DTD within this appendix provides the structure and content of documents prepared in accordance with this specification. Unless otherwise specified by the procuring activity, this Appendix is a mandatory part of this specification. The information contained herein is intended for compliance.

20. APPLICABLE DOCUMENTS.

20.1 Government documents.

20.2 Non-Government publications. The following documents form a part of this document to the extent specified herein. Unless otherwise specified, the issues of the documents which are DoD adopted are those listed in the issue of the DODISS cited in the solicitation. Unless otherwise specified, the issues of documents not listed in the DODISS are the issues of the documents cited in the solicitation.

ISO 8879	Information Processing - Text and Office Systems - Standard Generalized Markup Language (SGML)
ISO/IEC IS10744	Information Technology - Hypermedia/Time-based Document Structuring Language (HyTime)

(Application for copies should be addressed to the American National Standards Institute, 1430 Broadway, New York, NY 10018.)

30. GENERIC LAYER DOCUMENT TYPE DEFINITION.

30.1 Use of SGML. The markup tags described herein are based on rules outlined in ISO 8879. All data to be delivered digitally in accordance with this specification shall be tagged using the DTD in this section and associated content specific DTD(s).

30.2 Template document type. The DTD for this specification is as follows:

```
<!-- ************************************************************

                IETM CONTENT DATA MODEL     Version 6.1
                Generic Layer            1 October 1992
```

****

The IETM CDM provides a representation of technical information elements and their relationships. The CDM is composed of two separate layers. The first is the "Generic Layer". It defines general characteristics which are common across all applications. The second layer is the "Content Specific Layer," which contains content specific DTDs.

The generic layer defines the templates, linking elements, primitive elements, and context filtering elements which are used to create content specific DTDs. Templates define rules which must be followed in the creation of content specific DTD's and document instances. The templates provide the structure for creating composite nodes, context dependent filtering, user interaction and branching. The templates provide basic sets of rules to which elements must adhere. Those rules are explained after the declaration of each template in this document.

The CDM linking elements are taken from the HyTime Model. These elements provide the capability to link between CDM elements, other SGML files, and non SGML documents. These capabilities are explained in the HyTime Linking Mechanism section of this document.

The primitive elements ("text", "table", "graphic", "audio", "video","process", and "dialog") are defined in the generic layer. These elements may be used to construct a variety of composite elements in the content specific layer. The primitive elements' structures shall remain constant.

Context dependent filtering provides the capability to present the user with only the information that applies to a specific situation. The precondition and postcondition elements provide the mechanism for context dependent filtering. The precondition element enables the selection of the appropriate information for presentation. The postcondition element enables the recording of presentation events for later filtering.

****

PUBLIC ENTITY DECLARATIONS

****

<!ENTITY % dietmdb-a PUBLIC "-//USA-DOD//DTD Content Data Model Generic Layer//EN"> -->

FIGURE A-1. Sample Output - Kip-To-Kip Mode

```
<!ENTITY % hytime   PUBLIC
      "-//ANSI X3V1.8M//DTD Hypermedia/Time-based Document//EN"
      "hytime.dtd">
      %hytime;

<!--  This entity provides the capability of using the HyTime
linkage mechanism.  It defines the PUBLIC document to be included
into this DTD when parsed, then it initializes it with the
"%hytime;" declaration immediately after the entity declaration.
-->

<!-- Inclusion of MIL-M-28001 math package -->

<!ENTITY % mathpac PUBLIC "-//USA-DOD//DTD SUP MIL-M-28001
MATHPACK 900102//EN" "math.dtd">
%mathpac;
<!ENTITY % mathtxt "dfref | f" >
<!ENTITY % mathcon "df | dfg" >

<!--  The following entity declarations provide a mechanism for
referencing primitive elements in the generic layer, and within
any content specific DTD.  In SGML, an entity must be declared
prior to an element referencing that entity.  Therefore, all
primitive entities have been moved to this section.  A detailed
description of each primitive will appear when the element is
declared later in this document.  -->

<!ENTITY % text     "text | text-alts" >
<!ENTITY % table    "table | table-alts" >
<!ENTITY % graphic "graphic | graphic-alts | grphprim |
                    grphprim-alts" >
<!ENTITY % audio    "audio | audio-alts" >
<!ENTITY % video    "video | video-alts" >
<!ENTITY % process "process | process-alts" >
<!ENTITY % dialog   "dialog | dialog-alts" >

<!--  The following entity provides a simple method for
referencing the primitive elements defined in the generic layer.
-->

<!ENTITY % primitive  " %text; | %table; | %graphic; | %audio;
         | %video; | %process; | %dialog; | expression |
         assertion " >
```

```
<!-- ********************************************************

                    NOTATION DECLARATIONS

     ********************************************************

     The following notations define external references to
"public" graphics standards used in the CDM.  The specified
abbreviations (cgmbin, cgmclear, cgmchar, fax, iges, dxf, gks)
are used by the element "graphprm" to specify the type of graphic
representation used to encode a particular graphic primitive. -->


<!NOTATION   cgmbin      PUBLIC "ISO 8632/2//NOTATION Binary
encoding//EN">

<!NOTATION   cgmchar     PUBLIC "ISO 8632/2//NOTATION Character
encoding//EN">

<!NOTATION   cgmclear    PUBLIC "ISO 8632/2//NOTATION Clear text
encoding//EN">

<!NOTATION   fax         PUBLIC "-//USA-DOD//NOTATION CCITT Group
4 Facsimile//EN">

<!NOTATION   iges        PUBLIC "-//USA-DOD//NOTATION Initial
Graphics Exchange Specification//EN">

<!NOTATION   dxf         PUBLIC "-//USA-DOD//NOTATION DXF
 Encoding//EN">

<!NOTATION   gks         PUBLIC "-//USA-DOD//NOTATION Graphics
 Kernel System//EN" >


<!-- ********************************************************

                    TEMPLATES

     ********************************************************

     The following section defines the generic layer templates.
These templates define semantic rules for creating content
specific elements.  These semantic rules make up the minimum set
of constraints on content specific elements.
```

There are two general rules to follow when creating a content specific element. First, the element's content model must comply with the template's content model. Second, the template's attribute entity must be included in the element's attribute list. The attribute entities for all templates include the attributes "id", "cdm", "ref". The "cdm" attribute indicates which template the element is employing. The "id" and "ref" attributes are used for non-redundant referencing and linking.

The "ref" attribute utilizes the SGML #CONREF capability. A #CONREF attribute is only filled in when the element's content model is empty. In this case, the #CONREF attribute contains a reference which is a unique identifier to either an element of the appropriate type or a location element that resolves to an element of the appropriate type (see section on Hytime linking mechanism). When an element uses the #CONREF capability, the referencer's attribute list will take precedence over the referenced element's attributes.

This section includes an element declaration for each template (NODE, NODE ALTS, NODE SEQ, IF NODE, LOOP NODE). The declarations are enclosed within comments, and are not formally a part of the DTD. These element declarations use template names, in all caps, to describe content model constraints for each template. When creating content specific elements, these template names must be replaced by element names of the appropriate type. -->

<!-- ***** NODE TEMPLATE ***** -->

<!-- The NODE contains the content of the technical information. The NODE element creates hierarchy within the CDM. NODE also contains context filtering preconditions and postconditions. The link element within the NODE provides the capability to cross reference other technical information. The use of link, from the Hytime model, provides additional functionality by allowing a link to be made to a document outside the CDM specification boundary.

The NODE template provides the capability to create composite structures within the content specific layer. Composite structures may contain subcomponents that employ the NODE, NODE ALTS, or NODE SEQ templates. The NODE subcomponents may be composite structures themselves or they may be primitive NODES (text, tables, graphics, audio, video, process, and dialog). Composite structures create hierarchy within the CDM. When composite nodes contain other composite nodes there is an implied

hierarchy. The composite node in the content model is at a lower level in the hierarchy (e.g. a Task Node contains Step Nodes in its content model).

The following defines the NODE template:

```
<!ELEMENT "NODE"  - -  ( precond*, link*, ( NODE | NODE-ALTS |
                         NODE-SEQ | %primitive; )*, postcond* )>

-->

<!ENTITY % a.node
                      "id         ID          #IMPLIED
                       name       CDATA       #IMPLIED
                       type       CDATA       #IMPLIED
                       itemid     CDATA       #IMPLIED
                       cdm        NAME        #FIXED      'node'
                       ref        IDREF       #CONREF" >
```

<!-- The following semantic rules apply to any content specific element employing the NODE template: (1) The element may contain a list of preconditions that identify the element's applicability. The list of preconditions will be evaluated at presentation time, and if all preconditions evaluate to true, that node will be presented. (2) The element may contain relational links to other data items. (3) The element may contain subcomponents that employ the NODE, NODE ALTS, or NODE SEQ templates. (4) The element may contain a list of postconditions which record presentation events. The postconditions will be evaluated after the NODE and all its subcomponents have been presented. The postcondition values will then be assigned to their specified properties. -->

<!-- ***** NODE ALTS TEMPLATE ***** -->

<!-- NODE ALTS (node alternatives) will contain a list of mutually exclusive nodes. Their grouping is due to the fact that they apply in different contextual situations. In this manner, the NODE-ALTS element is a logical reference that contains a set of NODES which might apply to different situations. An important fact in the NODE-ALTS structure is that no hierarchy is implied between the generic identifier and the content model NODES (e.g. a Task-alts element will contain Task-nodes in its content model).

The following defines the NODE ALTS template:

```
<!ELEMENT "NODE-ALTS"        - -         ( NODE )+ >

-->

<!ENTITY % a.node-alts
          "id        ID         #IMPLIED
           cdm       NAME       #FIXED     'node-alts'
           ref       IDREF      #CONREF"                    >
```

<!-- The following semantic rules apply to any content specific element employing the NODE ALTS template. (1) The element must contain components that employ the NODE template. (2) The components must be of the same element type and at the same level in the hierarchy. (3) At presentation time, the precondition for each NODE alternative will be evaluated. The NODE whose precondition evaluates to "true" will be selected for presentation. (4) These components must contain mutually exclusive preconditions. In any specific situation, at most one node would have a precondition which evaluates to true. (5) There need not be an applicable component for every possible situation.

-->

<!-- ***** NODE SEQ TEMPLATE ***** -->

<!-- The NODE SEQ template is the mechanism for creating interactive sequences with the user.

The following defines the NODE SEQ template:

```
<!ELEMENT "NODE-SEQ"    - -     ( NODE | NODE-ALTS | IF-NODE |
                                  LOOP-NODE )+ >

-->

<!ENTITY % a.node-seq
          "id        ID         #IMPLIED
           cdm       NAME       #FIXED     'node-seq'
           ref       IDREF      #CONREF"                    >
```

<!-- The following semantic rules apply to the NODE SEQ template. (1) Any content specific element employing NODE-SEQ must contain components that employ the NODE, NODE ALTS, IF NODE, or LOOP NODE templates. (2) The components of a NODE SEQ are always traversed in the order they appear. This traversal

```
includes the branching and iteration implicit in any IF NODES or
LOOP NODES in the sequence logic.  -->

                <!-- *****  IF NODE TEMPLATE  ***** -->

<!--  The IF NODE template uses the same logic as the
IF-THEN-ELSE statement in a programming language.  The "IF" part
is the expression in the content model.  The "THEN" part is the
first NODE SEQ; the "ELSE" part is the second NODE SEQ, which is
optional.

     The following defines the IF NODE template:

<!ELEMENT "IF-NODE"    - -      ( expression, NODE-SEQ, NODE-SEQ? )
>

-->

<!ENTITY % a.if-node
           "id        ID         #IMPLIED
            cdm       NAME       #FIXED    'if-node'
            ref       IDREF      #CONREF"                    >

<!--  The following semantic rules apply to the IF NODE template.
(1)  The expression will be evaluated at presentation time; (2)
If the expression evaluates to "true" the first NODE SEQ will be
traversed; (3) If the expression evaluates to anything but
"true", and the second NODE SEQ is present, the second NODE SEQ
is traversed. (4) If the expression evaluates to anything but
"true", and the second NODE SEQ is not present, the next element
in the sequence will be presented. -->

                <!-- *****  LOOP NODE TEMPLATE  ***** -->

<!--  The LOOP NODE template provides the capability similar to
that found in a programming language for creating loops.  The
template provides the syntax for creating WHILE or FOR NEXT
loops, whichever applies to the situation.  For example, when
creating a FOR NEXT loop, the first assertion initializes the
control variable for the loop.  The expression is the test
criterium for exiting the loop.  The second assertion alters the
control variable at the end of each loop iteration.  The node
sequence provides the actual element(s) to be repeated within the
loop.
```

The following defines the LOOP NODE template:

```
<!ELEMENT "LOOP-NODE"     - -     ( assertion?, expression,
                                    assertion?, NODE-SEQ ) >

-->

<!ENTITY % a.loop-node
          "id        ID         #IMPLIED
           cdm       NAME       #FIXED    'loop-node'
           ref       IDREF      #CONREF"                    >
```

<!-- The following semantic rules apply to the LOOP NODE template, when employing it as in a FOR NEXT loop. (1) At the beginning of the loop the first assertion is evaluated and the value is assigned to the specified property. (2) The expression is evaluated and if the expression evaluates to anything but "true" the loop is terminated. (3) If the expression evaluates to true, the NODE SEQ is traversed. (4) At the end of each iteration, the second assertion is evaluated and the value is assigned to the specified property. (5) Steps 2-4 are continued until the loop terminates.

The semantic rules which apply to the LOOP NODE template, when employing it as in a WHILE loop, are as follows. (1) The expression is evaluated and if the expression evaluates to anything but "true" the loop is terminated. (2) If the expression evaluates to true, the NODE SEQ is traversed. (3) Steps 1-2 are continued until the loop terminates.

****

HYTIME LINKING MECHANISM

****

This section defines the linking mechanism which is based on the HyTime standard. -->

```
<!ELEMENT link      - -       ( #PCDATA )  >
<!ATTLIST link
          id              ID              #IMPLIED
          Hytime          NAME            #FIXED    'ilink'
          endtypes        CDATA           #FIXED    'desc parts text
                                                     table graphic
                                                     audio video
                                                     process'
```

```
            linkends        IDREFS          #REQUIRED
            desc            IDREFS          #IMPLIED
            parts           IDREFS          #IMPLIED
            text            IDREFS          #IMPLIED
            table           IDREFS          #IMPLIED
            graphic         IDREFS          #IMPLIED
            audio           IDREFS          #IMPLIED
            video           IDREFS          #IMPLIED
            process         IDREFS          #IMPLIED >

<!--  The "link" element provides the capability for creating
relational links within the CDM.  "Link" is included within the
content model of the NODE template, therefore, any content
specific element employing the NODE template may include
relational links.  The 'HyTime' attribute indicates that "link"
employs the "ilink" architectural form (template) from the HyTime
model.

  The endtypes define the allowable "anchors" to which a node
may point.  The initial endtypes defined are links to descriptive
information, parts information or any primitive.  There are no
limits to the type of linkends, and others may be added in the
future.

    ***********************************************************

                    PRIMITIVE ELEMENT DECLARATIONS

     ***********************************************************

     The following element declarations define the primitive data
elements used throughout the technical information.  Each element
is defined in terms which can be employed in any content specific
DTD.    -->

                    <!-- *****  TEXT  ***** -->


<!ELEMENT text        - -      ( precond*, link*, ( #PCDATA  |
                                 text-alts | text | %mathtxt;
                                 | %mathcon; )+ ) >
<!ATTLIST text
          %a.node; >

<!--  A "text" unit is basically a text string of "parsable
character data" or PCDATA.  Within a text string, there may be
embedded "text" elements which allow the referencing of other
```

```
elements or parts of elements through the link/location mechanism
explained in the HyTime section of this document.  Those embedded
"text" references are inserted in the text string that contained
them.  For example, the string may contain a reference to a
standard system name, a standard part nomenclature or a standard
task name.  By using this mechanism, standard terminology can be
referenced consistently throughout the data base, and any changes
to the standard terminology can be made in one location and
automatically updated throughout the data base.  -->

<!ELEMENT text-alts      - -      ( text )+ >
<!ATTLIST text-alts
          %a.node-alts;                      >

<!-- This element provides the ability to have context sensitive
filtering of text.  -->

                  <!-- *****  TABLE  ***** -->


<!ELEMENT table          - -      ( precond*, link*, rowhddef*,
                                    ( colhddef?, entry+ )+ ) >
<!ATTLIST table
          %a.node;                             >

<!-- This element defines how a table is constructed.  A "table"
will contain a list of the row headers.  Each column header will
be followed by one or more entries.  The combination of column
header and entries may be repeated for as many columns as the
"table" requires.  -->

<!ELEMENT table-alts      - -        ( table )+ >
<!ATTLIST table-alts
          %a.node-alts;                        >

<!-- This element provides the ability to have context sensitive
filtering of "tables".  -->

<!ELEMENT rowhddef    - -        ( %text; ) >
<!ATTLIST rowhddef
            id                ID          #IMPLIED
            ref               IDREF       #CONREF
            row               NUTOKEN     #REQUIRED >

<!-- This element defines a row header to be a piece of text and
a row number.  However, if a row header has been previously
```

```
defined, the 'ref' attribute allows the referencing of that
element from another table.  The 'row' for the element that
references a previously defined header takes precedence over the
original 'row' in the referenced header.  -->

<!ELEMENT colhddef     - -          ( %text; ) >
<!ATTLIST colhddef
            id                  ID          #IMPLIED
            ref                 IDREF       #CONREF
            colnum              NUTOKEN     #REQUIRED >

<!--  This element defines a column header to be a piece of text
and a column number.  However, if a column header has been
previously defined, the 'ref' attribute allows the referencing of
that element from another table.  The 'colnum' for the element
that references a previously defined header takes precedence over
the original 'colnum' in the referenced header.  -->

<!ELEMENT entry        - -          ( %text; | %graphic; ) >
<!ATTLIST entry
            id                  ID          #IMPLIED
            ref                 IDREF       #CONREF
            colnum              NUTOKEN     #REQUIRED
            row                 NUTOKEN     #REQUIRED >

<!--  This element defines an entry for a cell in a table to be a
piece of text and the appropriate row and column.  The row and
column define the cell to place the entry.  However, if an entry
has been previously defined, the 'ref' attribute allows the
referencing of that element from another table.  The 'colnum' and
'row' for the element that references a previously defined entry
take precedence over the original 'colnum' and 'row'. -->

                    <!-- *****  GRAPHICS  ***** -->

<!--  The CDM allows the referencing of "graphics" in external
representations or embedded within the CDM.  Graphics are an
integral part of technical information.  Therefore, all possible
standard representations have been included within the following
primitive elements.  -->


<!ELEMENT graphic      - -      ( precond*, link*, ( %graphic;)+ ) >
<!ATTLIST graphic
     %a.node;
     minsize          NUTOKENS              #IMPLIED
```

```
     penshape        CDATA                   #IMPLIED
     penpatt         CDATA                   #IMPLIED
     transfrm        NUTOKENS                #IMPLIED
     window          NUTOKENS                #IMPLIED >

<!-- This element describes graphics in terms of primitives or
references to other graphics, thus providing the ability to
create composite graphics.  -->

<!ELEMENT graphic-alts    - -       ( graphic )+ >
<!ATTLIST graphic-alts
          %a.node-alts;                          >

<!-- This element provides the ability to have context sensitive
filtering of graphics.  -->

<!ELEMENT grphprim      - -    ( precond*, link*, (%text;) ) >
<!ATTLIST grphprim
     %a.node;
     coding   (cgmchar | cgmbin | cgmclear |
               fax     | iges   | dxf      |
               gks)                          'cgmbin'
     minsize          NUTOKENS               #IMPLIED
     penpatt          CDATA                  #IMPLIED
     penshape         CDATA                  #IMPLIED
     transfrm         NUTOKENS               #IMPLIED
     x-location       NUTOKEN                #IMPLIED
     y-location       NUTOKEN                #IMPLIED
     window           NUTOKENS               #IMPLIED >

<!-- This element defines a primitive graphic which may be
contained in the content model or referenced by the 'ref'
attribute.  The graphic is represented in one of the valid
formats (cgmbin, cdmchar, cgmclear, fax, iges, dxf, gks), and the
format is indicated by the coding attribute.  The 'type'
attribute may identify a graphic as a "hotspot", thus making it
selectable during presentation.  The minsize attribute specifies
the minimum height requirements for display of the graphic.  Any
transformations or manipulations of the graphic, other than those
described by the notations, can be defined using the penpatt,
penshape, transfrm, or window attributes on the graphic primitive
element.  -->

<!ELEMENT grphprim-alts         - -      ( grphprim )+ >
<!ATTLIST grphprim-alts
          %a.node-alts;                            >
```

```
<!--  This element provides the ability to have context sensitive
filtering of graphic primitives.  -->

          <!-- *****  AUDIO, VIDEO & PROCESS  ***** -->

<!-- The elements "audio", "video", "process" provide the
capability for the author to define an audio sequence, video
sequence, or a call to a software process.  These element
definitions require further inspection and updating, which will
be done upon completion of a closer look at the HyTime
multi-media event definitions. -->


<!ELEMENT audio      - -      ( precond*, link* ) >
<!ATTLIST audio
          %a.node;
          external-ptr   IDREF   #REQUIRED  >

<!--  This element will be used to include an audio sequence into
technical information.  The model is incomplete pending the
Hytime completion.  -->

<!ELEMENT audio-alts  - -  ( audio+ ) >
<!ATTLIST audio-alts
          %a.node-alts;                >

<!--  This element provides the ability to have context sensitive
filtering of audio sequences.  -->


<!ELEMENT video      - -      ( precond*, link* ) >
<!ATTLIST video
          %a.node;
          external-ptr   IDREF   #REQUIRED  >

<!--  This element will be used to include an video sequence into
technical information.  The model is incomplete pending the
Hytime completion.  -->

<!ELEMENT video-alts  - -  ( video+ ) >
<!ATTLIST video-alts
          %a.node-alts;                >

<!--  This element provides the ability to have context sensitive
filtering of video sequences.  -->
```

```
<!ELEMENT process    - -      ( precond*, link*, parameter* ) >
<!ATTLIST process
          %a.node;
          external-ptr   IDREF   #REQUIRED    >

<!-- This element is used to reference an external software
process. The external pointer attribute will point to a location
element defined within Hytime. The parameter element will
provide some expression for passing parameters to the software
process. -->

<!ELEMENT process-alts  - -  ( process+ ) >
<!ATTLIST process-alts
          %a.node-alts;                    >

<!-- This element provides the ability to have context sensitive
filtering of processes and reduces data redundancy through the
referencing capability of #CONREF. -->

<!ELEMENT parameter     - -    ( expression ) >
<!ATTLIST parameter
          mode      ( in | out | in-out )    'in' >

<!-- This element includes an expression which will be used to
create the parameters required by an external software process.
For example: the 1553 bus on the aircraft might require
parameters concerning a given channel to look up. The parameter
element will contain the channel required by the process. -->


                <!-- *****  DIALOGS  ***** -->

<!-- "Dialogs" provide interactivity between the user and the
electronic technical information. It is sometimes necessary to
receive data from the user in order to present the proper
information at the proper time. "Dialogs" provide the capability
of prompting the user to input a response ("fillin"), select a
choice from a set of alternatives ("menu"), or to select items
from within a text, table or graphic ("selection"). -->

<!ELEMENT dialog      - -   ( precond*, link*, ( %text;)?,
                              ( %dialog; | fillin | menu |
                              selection )+ ) >
<!ATTLIST dialog
          %a.node;
          agent      CDATA       'human' >
```

```
<!-- This element defines a "dialog" which provides the
capability for user interaction. A "dialog" could contain a
subdialog, a "fillin", a "menu", a "selection", or any
combination of the four. It may also contain an optional text
string which would be the title of the composite dialog. The
'agent' attribute defines to whom the question is asked ( i.e. a
technician or a 1553 Bus ). -->

<!ELEMENT dialog-alts     - -      ( dialog )+ >
<!ATTLIST dialog-alts
          %a.node-alts;                          >

<!-- This element provides the ability to have context sensitive
filtering of dialogs and reduces data redundancy through the
referencing capability of #CONREF. -->


<!ELEMENT fillin  - -   ( link*, prompt, property, ( %text )?,
                        generic-range? ) >
<!ATTLIST fillin
          id              ID              #IMPLIED
          ref             IDREF           #CONREF  >

<!-- This element defines how a fill-in shall be constructed.
It would contain a "prompt", a "property", and an optional
default value. The "prompt" contains the question to be
presented to the user. The property element identifies the
variable which will receive a value from the user's response.
The property element also identifies the value type of the user's
response. The fill-in will be presented to the user according to
the value type. The optional text element provides the
capability for defining a default value for the fill-in. The
generic range element may be used to provide a range for the
value(s) of the fill-in. -->

<!ELEMENT generic-range  - -  ( set | sequence | num-range ) >

<!ELEMENT num-range  - -  ( low-bound, high-bound ) >

<!ELEMENT low-bound  - -  ( integer | real ) >

<!ELEMENT high-bound  - -  ( integer | real ) >

<!-- These elements define two types of range constraints. If
the generic range contains a set or sequence, then the contents
of that set or sequence become the constraints for the fillin.
```

```
If the generic range contains a number range, then the low and
high bounds define the constraints for the fillin. -->

<!ELEMENT menu       - -    ( link*, prompt, choice+ ) >
<!ATTLIST menu
          id              ID              #IMPLIED
          ref             IDREF           #CONREF
          select          ( single | multiple )  'single' >

<!-- This element defines how a "menu" is built for technical
information.  It consists of a "prompt" followed by one or more
"choices".  The "select" attribute allows the author to designate
the number of choices that may be selected by the user.   -->

<!ELEMENT prompt     - -    ( %text; | %graphic; ) >
<!ATTLIST prompt
          id              ID              #IMPLIED
          ref             IDREF           #CONREF >

<!-- This element defines the "prompt" to be displayed to the
user for the presentation of a "fillin" or a "menu".  It allows
the prompt to be either a text string (probably in the form of a
question) or a graphic (a picture which requires an answer).-->

<!ELEMENT choice     - -    ( ( %text; | %graphic; ),
                             ( assertion+ | %dialog;) ) >
<!ATTLIST choice
          id                 ID           #IMPLIED
          ref                IDREF        #CONREF
          default         ( Yes | No )    'No'  >

<!-- This element defines the choices for a menu.  A "choice"
contains a "text" or "graphic" element followed by an assertion
or "dialog" element.  The "text" or "graphic" element will be
displayed to the user as a part of the menu.  The assertion or
dialog identifies the action to be taken if the user selects that
choice.  The default attribute provides a method of indicating
whether a choice is designated as a default for the menu.   -->

<!ELEMENT selection  - -  ( ( link, ( assertion+ | %dialog; ) )+,
                             ( text | table | graphic ) ) >
<!ATTLIST selection
          id              ID              #IMPLIED
          ref             IDREF           #CONREF >

<!-- This element provides the capability of creating a special
"dialog" that allows selection within a given picture, text
```

```
string or table.  The semantics require a 'link' for each
selectable item within the text, table, or graphic selection.
Each 'link' must have at least one linkend specifying the
selectable element in the text, table, or graphic.  Each link
will be paired with an assertion or dialog specifying the action
to be taken if that item is selected.   -->

<!--*******************************************************

                CONTEXT FILTERING ELEMENT DECLARATIONS

     *******************************************************
```

Context dependent filtering provides the capability to present the user with only the information that applies to his specific situation. The precondition and postcondition elements provide the mechanism for context dependent filtering. The precondition element enables the selection of the appropriate information for presentation. The postcondition element enables the recording of presentation events for later filtering.

This mechanism assumes that a state table is maintained at presentation time to represent the current situation. The current situation or state is defined by a set of property value pairs. A property value pair associates a value to a property name. It provides the capability to obtain a value by looking up a property name in the state table. -->

```
<!ELEMENT precond     - -       ( expression ) >
<!ATTLIST precond
           id                 ID                      #IMPLIED
           ref                IDREF                   #CONREF  >

<!--  A precondition contains an expression to be evaluated at
presentation time.  The precondition is satisfied if the
evaluation results in "true".  -->

<!ELEMENT postcond     - -       ( assertion ) >
<!ATTLIST postcond
            id                 ID          #IMPLIED
            ref                IDREF       #CONREF >

<!--  The postcondition contains an assertion which is evaluated
whenever the node containing the postcondition is traversed.
After a NODE has been presented, the assertion will be evaluated
and the appropriate property value pairs will be asserted.  The
most recent assignment will overwrite any previous value.   -->
```

```
<!ENTITY % binop  " eq | ne | lt | gt | le | ge | and | or
                  | xor | concat | substring | append
                  | plus | minus | times | divide
                  | idivide | exponent | mod | remove
                  | union | intersect | set-diff
                  | member | subset | disjoint | add
                  | subsequence " >

<!-- The binary operation entity enumerates all of the possible
binary operators which may be used within an expression.  -->

<!ENTITY % unop  " not | empty | size | head | tail | neg
                 | remove | trunc | float | index | undef
                 | max | min" >

<!-- The unary operation entity enumerates all of the unary
operators which may be used within an expression.  -->

<!ENTITY  % value  " boolean | string | sequence | set
                   | real | integer | nil " >

<!-- This entity enumerates the legal value types which
properties may contain.    -->

<!ELEMENT expression   - -  ( ( expression, (%binop;),expression)
                            | (( %unop;), expression )
                            | property | %value; ) >
<!ATTLIST expression
          id        ID        #IMPLIED
          ref       IDREF      #CONREF  >

<!-- The expression element contains one of four types of
subexpressions: a binary operation between two expressions, a
unary operation upon an expression, a property, or a value. If
the expression contains a binary or unary operation, the value is
defined by the semantic rules specified later in this section.
If the expression contains a property, the value of the
expression is obtained by looking up the current value of the
property in the state table. If the expression contains a value,
that value is returned as the result.  -->

<!ELEMENT assertion  - -  ( property, expression ) >
<!ATTLIST assertion
             id            ID            #IMPLIED
             ref           IDREF         #CONREF  >
```

```
<!-- The assertion element provides the mechanism for pairing a
value with a property. The semantics of assertions vary from
postconditions. When an assertion is present in a node, the
presentation rules for that content specific element will
determine whether the assertion is evaluated. -->

<!ELEMENT ( eq, ne, lt, gt, le, ge, and,
            or, xor, concat, substring,
            plus, minus, times, divide,
            idivide, exponent, mod, union,
            intersect, set-diff, member, subset,
            disjoint, append, subsequence,
            not, empty, size, head, tail,
            neg, trunc, float, undef, max, min )   - o   EMPTY >

<!ELEMENT add  - o              ( index-value )? >

<!ELEMENT remove   - o          ( index-value, index-value? )? >

<!ELEMENT index     - o       ( index-value, index-value? ) >

<!ELEMENT index-value    - o          ( #PCDATA )  >

<!-- The above elements are used to identify the operators which
may be applied in an expression.    -->

<!ELEMENT property       - -            ( #PCDATA ) >
<!ATTLIST property
          id                    ID        #IMPLIED
          ref                   IDREF     #CONREF
          type                  CDATA     #REQUIRED
          value-type            CDATA     'general'
          dialog-ref            IDREF     #IMPLIED>

<!-- The property element contains parsable character data which
represents the property (variable) name. The value of a property
may be obtained by finding the current value associated with the
property name in the state table.

The 'type' attribute contains a character string which may be
used by the author to identify different property classes. The
'value-type' attribute is used to denote the allowable data types
which may be assigned to the property. The current legal values
for 'value-type' are any combination of the following: "boolean",
"integer", "real", "set", "sequence", "string", and "general".
The 'dialog-ref' attribute will hold the IDREF of a "dialog" or
"process" element which will acquire a value for the property, if
```

```
"property" is undefined (i.e., equal to "nil") at presentation
time.  -->

<!ELEMENT ( boolean, string, real, integer ) - -   ( #PCDATA ) >

<!--  These elements define the values boolean, string, real, and
integer to be character data.  -->

<!ELEMENT ( set, sequence )      - -    ( %value; )* >

<!--  These element are used to define a set or sequence as being
zero, one or more values.  -->

<!ELEMENT nil                - -      EMPTY >

<!--This element signifies that the value of the associated
property is undefined. Property's of any type can take on the
"nil" value.  -->


<!-- ************************************************************

                    Semantics of expression operations

     ************************************************************

Listed below are the value-types allowed in the generic layer and
the valid operators under each value-type, and the semantics of
each operation including the return value-type.


OPERATIONS WHICH APPLY TO MULTIPLE DATATYPES:

Operation:      <eq|ne>
Form:           <value><eq|ne><value>
Return Value:   <boolean>
Meaning:         If both operands are the same value-type (or both
                 are numbers ) then the return value is dependent
                 upon what eq|ne means for that value-type. If the
                 operands are of different types, the return value
                 is 'False'.

Operation:      <size>
Form:           <size><string>
                <size><sequence>
                <size><set>
```

Return Value: <boolean>
Meaning: An integer value which is the length of the <string> or the number of values in the <set|sequence>. For <set|sequence> this number represents the members of the <set|sequence>. It does not count the elements which are members of an included <set|sequence>.

Operation: <empty>
From: <empty><string>
<empty><set>
<empty><sequence>
Return Value: <boolean>
Meaning: True if the string, set, or sequence is empty. False otherwise. Logically equivalent to size(<..>) = 0.

Operation: <index>
Form: <index><string>
<index><sequence>
Return Value: <string> | <sequence>
Meaning: The index operator can have one or two index-values in its SGML content. An index-value is a signed integer value. Its meaning is dependent upon its sign. A positive value means an index position counted from the beginning of the <string|sequence>. A negative number means an index position counted back from the end of the <string|sequence>. A zero means the end of the string.

Operation: <add>
Form: <set><add><value>
<sequence><add><value>
Return Value: <set> | <sequence>
Meaning: For a set, add simply means make a new set which has all the members of the old set plus <value>. For a sequence the add operator shall have an index-value as described above for the index operation. The <value> will be inserted before the position pointed to by the index position. If no index-value is given the <value> is added at the end of the sequence.

Operation: <remove>

Form: <set><remove><value>
<sequence><remove><value>
<remove><sequence>
<remove><string>
Return Value: <set> | <sequence> | <string>
Meaning: For a <set> remove returns a <set> with <value> removed. For a <sequence> using the binary operand form it returns a <sequence> which has the first instance of <value> removed. For a <sequence> or <string> as a unary operator remove must contain an index-value which refers to the position from which the character in the <string> is to be removed or the value in the <sequence> is to be removed. The new string or sequence will be the old one up to but not including the index position concatenating with the old one after the index position.

Operation: <member>
Form: <value><member><set>
<value><member><sequence>
Return Value: <boolean>
Meaning: True if <set|sequence> contains an member who is equal to <value>. False otherwise. This is not a recursive search on any <set|sequence> that might be part of the <set|sequence>.

BOOLEAN OPERATIONS:

Operation: <or>
Form: <boolean><or><boolean>
Return Value: <boolean>
Meaning: The boolean or function.

Operation: <and>
Form: <boolean><and><boolean>
Return Value: <boolean>
Meaning: The boolean and function.

Operation: <xor>
Form: <boolean><xor><boolean>
Return Value: <boolean>
Meaning: The boolean xor function.

Operation: <not>
Form: <not><boolean>
Return Value: <boolean>

Meaning: The boolean not function.

STRING OPERATIONS:

Operation: <concat>
Form: <string><concat><string>
Return Value: <string>
Meaning: The return value is a new string which is equal to the first string with the second string concatenated to the end of it.

Operation: <empty>
Form: <empty><string>
Return Value: <boolean>
Meaning: True if the string is empty (size = 0). False otherwise. This is equivalent to size (<string> = 0).

Operation: <substring>
Form: <string><substring><string>
Return Value: <boolean>
Meaning: True if the first string is a substring of the second string. False otherwise.

SEQUENCE OPERATIONS

Operation: <append>
Form: <sequence><append><sequence>
Return Value: <sequence>
Meaning: A new sequence equal to the first sequence with the second sequence appended to the end.

Operation: <subsequence>
Form: <sequence><subsequence><sequence>
Return Value: <boolean>
Meaning: True if the first sequence is a subsequence of the second. False otherwise.

Operation: <head>
Form: <head><sequence>
Return Value: <value>
Meaning: Returns the first element in sequence.

Operation: <tail>

Form: <tail><sequence>
Return Value: <sequence>
Meaning: Returns a sequence with the first element removed.

SET OPERATIONS

Operation: <union>
Form: <set><union><set>
Return Value: <set>
Meaning: A new set which is the union of the two sets.

Operation: <intersect>
Form: <set><intersect><set>
Return Value: <set>
Meaning: A new set which is the intersection of the two sets.

Operation: <set-diff>
Form: <set><set-diff><set>
Return Value: <set>
Meaning: A new set which is the difference of the two sets.

Operation: <disjoint>
Form: <set><disjoint><set>
Return Value: <boolean>
Meaning: True if the intersection of the two sets is empty. False otherwise. This is equivalent to empty(<set1> intersect <set2>).

Operation: <subset>
Form: <set><subset><set>
Return Value: <boolean>
Meaning: True if the first set is a subset of the second. False otherwise.

NUMBER OPERATIONS

Operation: <gt>
Form: <integer><gt><integer>
<integer><gt><real>
<real><gt><integer>
<real><gt><real>
Return Value: <boolean>

Meaning: True if the first number is greater than the second. False otherwise.

Operation: <ge>
Form: <integer><ge><integer>
<integer><ge><real>
<real><ge><integer>
<real><ge><real>
Return Value: <boolean>
Meaning: True if the first number is greater than or equal to the second. False otherwise.

Operation: <lt>
Form: <integer><lt><integer>
<integer><lt><real>
<real><lt><integer>
<real><lt><real>
Return Value: <boolean>
Meaning: True if the first number is less than the second. False otherwise.

Operation: <le>
Form: <integer><le><integer>
<integer><le><real>
<real><le><integer>
<real><le><real>
Return Value: <boolean>
Meaning: True if the first number is less than or equal to the second. False otherwise.

Operation: <plus>
Form: <integer><plus><integer>
<integer><plus><real>
<real><plus><integer>
<real><plus><real>
Return Value: <integer> | <real>
Meaning: Return the value of the first number plus the second number. The return value is a real unless both numbers are integers.

Operation: <minus>
Form: <integer><minus><integer>
<integer><minus><real>
<real><minus><integer>
<real><minus><real>
Return Value: <integer> | <real>

Meaning: Return the value of the first number minus the second number. The return value is a real unless both numbers are integers.

Operation: <times>
Form: <integer><times><integer>
<integer><times><real>
<real><times><integer>
<real><times><real>
Return Value: <integer> | <real>
Meaning: Return the value of the first number times the second number. The return value is a real unless both numbers are integers.

Operation: <divide>
Form: <integer><divide><integer>
<integer><divide><real>
<real><divide><integer>
<real><divide><real>
Return Value: <real>
Meaning: Return the value of the first number divided by the second number. The return value is a real.

Operation: <idivide>
Form: <integer><idivide><integer>
<integer><idivide><real>
<real><idivide><integer>
<real><idivide><real>
Return Value: <integer>
Meaning: Return the value of the first number divided by the second number. The return value is truncated to an integer.

Operation: <exponent>
Form: <integer><exponent><integer>
<integer><exponent><real>
<real><exponent><integer>
<real><exponent><real>
Return Value: <integer> | <real>
Meaning: Return the value of the first number raised to the power of the second number. The value is a real unless the first number is an integer and the second number is a positive integer.

Operation: <mod>
Form: <integer><mod><integer>
Return Value: <integer>

```
Meaning:        The return value is equal to the integer
                remainder after the first number is
                integer-divided by the second. ( This is the
                standard definition of the mod operator ).

Operation:       <neg>
Form:            <neg><integer>
                 <neg><real>
Return Value:    <integer> | <real>
Meaning:        The return value is the negative of the number.
                It is an integer if the number is an integer, and
                real if the number is a real.

Operation:       <trunc>
Form:            <trunc><integer>
                 <trunc><real>
Return Value:    <integer>
Meaning:               The return value is the number truncated to
                       be an integer.

Operation:       <float>
Form:            <float><integer>
                 <float><real>
Return Value:    <real>
Meaning:              The return value is the number converted to
                      a real number value.

-->
```

A CONTENT SPECIFIC DOCUMENT TYPE DEFINITION (DTD)

10. SCOPE.

10.1 Scope. The DTD within this appendix provides the structure and content of documents prepared in accordance with this specification. Unless otherwise specified by the procuring activity, this Appendix is a mandatory part of this specification. The information contained herein is intended for compliance.

20. APPLICABLE DOCUMENTS.

20.1 Government documents.

20.2 Non-Government publications. The following documents form a part of this document to the extent specified herein. Unless otherwise specified, the issues of the documents which are DoD adopted are those listed in the issue of the DODISS cited in the solicitation. Unless otherwise specified, the issues of documents not listed in the DODISS are the issues of the documents cited in the solicitation.

ISO 8879	Information Processing - Text and Office Systems -Standard Generalized Markup Language (SGML)
ISO/IEC IS10744	Information Technology - Hypermedia/Time-based Document Structuring Language (Hytime)

(Application for copies should be addressed to the American National Standards Institute, 1430 Broadway, New York, NY 10018.)

30. A CONTENT SPECIFIC DOCUMENT TYPE DEFINITION.

30.1 Use of SGML. The markup tags described herein are based on the rules outlined in ISO 8879. All data to be delivered digitally in accordance with this specification shall be tagged using the DTD in this section and the generic DTD found in Appendix A.

30.2 Template document type. The DTD for this specification is as follows:

```
<!DOCTYPE  techinfo  [

<!--  ***************************************************

          IETM CONTENT DATA MODEL          Version 6.1
          Content Specific DTD          1 October 1992

      ***************************************************  -->

<!-- This document contains a content specific DTD for O-level
maintenance.  The creation of a content specific DTD represents
the second layer of the CDM.  It identifies all the content
specific elements and their relationships for a given
application.  In this instance, the application happens to be the
display of organizational level data to the technician.

     The DTD employs the characteristics defined by the
templates of the "Generic Layer."  The use of the generic layer
primitives means that we do not have to redefine the text, table,
graphic, audio, video, or process elements within this document.

     This document breaks down O-level data into a hierarchy
based upon the system/subsystem structure of the weapon system.
It identifies four different types of information which may be
referenced within the document.  They are; procedural,
descriptive, parts, and fault information.  Each type of
information is referenced by the system where it is most
appropriate.

     ***********************************************************

                    PUBLIC ENTITY DECLARATIONS

     ***********************************************************

<!ENTITY % dietmdb-b PUBLIC "-//USA-DOD//DTD Content Data Model
Content Specific Layer//EN">  -->

<!ENTITY  % dietmdb-a PUBLIC "-//USA-DOD//DTD Content Data Model
Generic Layer//EN" "cdm61.dtd" >
%ietmdb-a;

<!-- This entity includes the public identifier for the generic
layer of the CDM.  It provides access to the template, primitive,
user-interaction, and filtering elements within the generic
layer.  -->
```

```
<!-- The following entities are used to refer to the elements
used in this content specific DTD. They use the node and node alt
templates from the CDM generic layer. These entities are here
because of the top down methodology of SGML.  By defining the
entities at the beginning of the DTD, any element below this
point can use the entity declarations.-->

<!ENTITY % sub-prims  " %text; | %table; | %graphic; | %audio; |
%video; | %process; " >
<!ENTITY  % system      "system | system-alts" >
<!ENTITY  % descinfo    "descinfo | descinfo-alts" >
<!ENTITY  % task        "task | task-alts" >
<!ENTITY  % reqcond     "reqcond | reqcond-alts"  >
<!ENTITY  % input       "input | input-alts"  >
<!ENTITY  % person      "person | person-alts"  >
<!ENTITY  % equip       "equip | equip-alts"  >
<!ENTITY  % refmat      "refmat | refmat-alts"  >
<!ENTITY  % expend      "expend | expend-alts"  >
<!ENTITY  % consum      "consum | consum-alts"  >
<!ENTITY  % alert       "alert | alert-alts" >
<!ENTITY  % step        "step | step-alts" >
<!ENTITY  % follow-on   "follow-on | follow-on-alts"  >
<!ENTITY  % partinfo    "partinfo | partinfo-alts"  >
<!ENTITY  % partbase    "partbase | partbase-alts"  >
<!ENTITY  % connection  "connection | connection-alts" >
<!ENTITY  % attach-part "attach-part | attach-part-alts" >
<!ENTITY  % location    "location | location-alts" >
<!ENTITY  % faultinf    "faultinf | faultinf-alts " >
<!ENTITY  % test        "test | test-alts" >
<!ENTITY  % outcome     "outcome | outcome-alts" >
<!ENTITY  % fltstate    "fltstate | fltstate-alts" >
<!ENTITY  % fault       "fault | fault-alts" >
<!ENTITY  % rect        "rect | rect-alts" >

<!--  ************************************************

                    Techinfo Declaration

      ************************************************  -->


<!ELEMENT techinfo   - -   ( version+, (%system;)+ )  >
<!ATTLIST techinfo
          %a.node;   >

<!-- This element declaration represents the top layer of the
information contained in the DTD.  The content model contains the
top level system, such as "F-15", "M-1" or "F/A-18".  The
```

```
hierarchy begins at this level.  -->

<!--  ****************************************************

                        System Declaration

      **************************************************** -->

<!-- The system element defines the vehicle/system/subsystem/
subassembly hierarchy for the weapon system.  A system element
must be created for any component (ie., vehicle, system,
subsystem, subassembly) which has associated technical
information (ie., descriptive, procedural, fault, or part
information). -->

<!ELEMENT system  - -  ( precond*, link*, (%system;)*,
                       (%descinfo;)*, (%task;)*, (%partinfo;)*,
                       (%faultinf;)*  )  >
<!ATTLIST system
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a )'a'  >

<!-- The system element employs the 'NODE' template from the
generic layer.  A "system" contains a list of preconditions which
define the elements applicability, relational links to other
elements, sub-system elements and descriptive, task, part, and
fault information about the system.  -->

<!ELEMENT system-alts  - -  ( system )+  >
<!ATTLIST system-alts
          %a.node-alts; >

<!-- This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering at the system
level.  -->

<!--  ****************************************************

                        Version Declaration

      **************************************************** -->

<!-- The following declaration is an attempt at controlling
different versions of technical information in the database.  A
more robust description of how the version element will work is
to follow.  -->
```

```
<!ELEMENT  version     - -       ( %text; )?  >
<!ATTLIST  version
                  %a.node;
                  revisionNMTOKEN#REQUIRED
                  revdate              NUMBER        #REQUIRED
                  changenoNUMBER  #REQUIRED
                  chgdate  NUMBERS       #REQUIRED
                  deleted  NMTOKENS      #IMPLIED    >

<!--  ************************************************

              Descriptive Information  Declaration

      ************************************************  -->

<!-- The element "descinfo" is used to define general purpose,
non-procedural, narrative information such as theory of
operation, schematics, etc which are associated with a system
component.  "Descinfo" is  very flexible.  It can be used to
describe any arbitrary, hierarchical hypertext-like node
containing sub-paragraphs ("para-seq"), ("text", "table",
"graphic", "audio", "video", "process"), user interaction
instructions ("dialog"), and postcondition properties
("postcond") which are asserted whenever the "descinfo" is
read.   -->


<!ELEMENT descinfo      - -  ( precond*, link*, para-seq,
                               postcond*  ) >
<!ATTLIST descinfo
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a )'a' >

<!-- The descinfo element employs the 'NODE' template from the
generic layer.  "Descinfo" contains a list of preconditions
which define the element's applicability, relational links to
other elements, paragraph sequences, and a list of postconditions
which may change the state of the system.   -->

<!ELEMENT descinfo-alts  - -  ( descinfo )+  >
<!ATTLIST descinfo-alts
          %a.node-alts; >

<!-- This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of descriptive
information.  -->
```

```
<!--  ************************************************

                        Para Declaration

      ************************************************  -->

<!ELEMENT  para    - -  ( precond*, (%sub-prims;)+, para-seq?,
                          postcond* ) >
<!ATTLIST  para
           %a.node;
           version IDREF#REQUIRED
           status  ( u | a )'a' >

<!-- The "para" element employs the NODE template.  It defines
the information which may be contained within the descriptive
information as any primitive element defined in the generic
layer.  -->

<!ELEMENT  para-alts - -  ( para )+  >
<!ATTLIST  para-alts
           %a.node-alts;    >

<!-- This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of paragraph
information.  -->

<!ELEMENT  para-seq    - -  ( %descinfo; | para | para-alts |
                              if-para | loop-para )+ >
<!ATTLIST  para-seq
           %a.node-seq;   >

<!-- This element employs the 'NODE SEQ' template from the
generic layer.  It provides the capability to create sequences of
paras.
  -->

<!ELEMENT if-para      - -  ( expression, para-seq, para-seq? ) >
<!ATTLIST if-para
           %a.if-node;   >

<!-- This element employs the 'IF NODE' template from the
generic layer.  It allows conditional selection of paras
depending on a precondition.  -->

<!ELEMENT loop-para    - -   ( assertion?, expression, assertion?,

                               para-seq ) >
```

```
<!ATTLIST loop-para
          %a.loop-node;  >

<!--  This element employs the 'LOOP NODE' template from the
generic layer.  It provides the capability of looping through a
sequence of paras.  -->

<!--**********************************************

                    Task Declaration

      **********************************************  -->

<!--  The element "task" defines a maintenance procedure, such as
a removal, repair, replacement, test, adjustment, etc. associated
with a "system" component.  -->


<!ELEMENT task      - -  ( precond*, link*, (%input;),
                           step-seq, (%follow-on;)*, postcond* ) >
<!ATTLIST task
          %a.node;
          version         IDREF                        #REQUIRED
          status          ( u | a )                      'a'
          esttime         NUTOKEN                      #IMPLIED
          operability     CDATA                        #IMPLIED
          servicedes      CDATA                        #IMPLIED  >

<!--  The "task" element employs the 'NODE' template from the
generic layer.  A "task" element contains a list of preconditions
which define the task's applicability, relational links to other
information elements and input conditions for beginning the task,
precautionary messages ( i.e., warnings, cautions and notes), a
sequence of procedural steps, a list of follow on conditions
which must be accomplished sometime following the completion of
the task, and a list of postconditions which define any state
changes to be made after the task is accomplished.
-->

<!ELEMENT task-alts  - -  ( task )+  >
<!ATTLIST task-alts
          %a.node-alts; >

<!--  This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of tasks.  -->
```

```
<!--  ***************************************************

                        Input Declaration

      ***************************************************  -->

<!-- The input element identifies all the set-up conditions
which must be met prior to beginning a task.  -->


<!ELEMENT input    - -  ( precond*, link*, (%alert;)*,
                        (%reqcond)*, (%person;)+, (%refmat;)*,
                        (%equip;)*, (%expend;)*, (%consum;)* ) >
<!ATTLIST input
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a )'a' >

<!-- The "input" element employs the 'NODE' template from the
generic layer.  An "input" contains applicability preconditons,
relational links to other elements, and the personnel,
consumables,
equipment and required conditions for accomplishing the task.
-->

<!ELEMENT input-alts       - -     ( input )+ >
<!ATTLIST input-alts
          %a.node-alts; >


<!-- This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of input
conditions.  -->

<!--  ***************************************************

                    Required Condition Declaration

      ***************************************************  -->

<!-- A required condition (RECOND) identifies a maintenance
condition (eg., aircraft safe for maintenance ), which must be
satisfied before beginning a task.  It also identifies the
task(s) or step(s) which accomplish the required condition if it
is not
satisfied. -->
```

```
<!ELEMENT reqcond      - -  ( precond*, link*, (%text;)?,
                            ( expression, ( %task; | %step; ),
                              assertion* ), postcond* ) >
<!ATTLIST reqcond
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a)'a' >

<!-- The "reqcond" element employs the 'NODE' template from the
generic layer. A "reqcond" contains a set of preconditions which
define the required maintenance condition's applicability,
relational links, an optional text element which describes the
maintenance condition, a list of task(s) or step(s) which provide
instructions for accomplishing the maintenance condition, and a
set of postconditions which define the state changes to be made
once the maintenance condition is accomplished.   -->

<!ELEMENT reqcond-alts - -      ( reqcond )+ >
<!ATTLIST reqcond-alts
          %a.node-alts;    >

<!-- This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of required
conditions.  -->

<!--  ****************************************************

                    Reformat and Expend Declarations

      ****************************************************  -->

<!-- The following elements identify reference material and
expendables for a task.     -->

<!ELEMENT  refmat  - -  ( precond*, link*, (%text)? ) >
<!ATTLIST  refmat
        %a.node;
        version    IDREF#REQUIRED
        status     ( u | a )'a'
        desig      CDATA#REQUIRED >

<!ELEMENT  refmat-alts    - -      ( refmat )+ >
<!ATTLIST  refmat-alts
                  %a.node-alts;    >

<!ELEMENT  expend  - -  ( precond*, link*, (%partbase)?,
                        (%consum;)* ) >
<!ATTLIST  expend
```

```
                    %a.node;
                    versionIDREF#REQUIRED
                    status( u | a )'a'
                    quantityNUTOKEN#REQUIRED >

<!ELEMENT  expend-alts     - -     ( expend )+  >
<!ATTLIST  expend-alts
                    %a.node-alts;     >

<!--  ***************************************************

                        Person Declaration

      *************************************************** -->

<!-- This element is used to identify the personnel requirements
for a given task.  The 'type' attribute will be used to identify
the kind of technician required.  The 'quantity' attribute
identifies the number of that type of technician required for the
task.  -->


<!ELEMENT person      - -     ( precond*, link*, (%text;)? )  >
<!ATTLIST person
          %a.node;
          version IDREF#REQUIRED
          status   ( u | a )'a'
          quantity  NUTOKEN   #IMPLIED  >

<!-- The person element employs the 'NODE' template from the
generic layer.   -->

<!ELEMENT person-alts       - -     ( person )+ >
<!ATTLIST person-alts
          %a.node-alts; >

<!-- This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of person
elements.  -->

<!--  ***************************************************

                        Equipment Declaration

      *************************************************** -->

<!-- This element identifies all the support equipment required
for the completion of the task.  -->
```

```
<!ELEMENT equip    - -  ( precond*, link*, (%equip;)*,
                        (%text)?, (%partbase;) ) >
<!ATTLIST equip
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a )'a'
          quantity   NUTOKEN  #IMPLIED  >

<!-- The equip element employs the 'NODE' template from the
generic layer.  An "equip" contains applicability preconditons,
relational links to other elements, and any alternate equipment.
The quantity attribute identifies the number of equipment items
required to complete the task.  -->

<!ELEMENT equip-alts        - -      ( equip )+ >
<!ATTLIST equip-alts
          %a.node-alts;   >

<!-- This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of equipment
elements.  -->

<!--  ****************************************************

                        Consumable Declaration

      ****************************************************  -->

<!-- This element identifies all the consumables required for
the completion of the task.  -->


<!ELEMENT consum    - -  ( precond*, link*, (%partbase;)?,
                         (%consum;)* )  >
<!ATTLIST consum
          %a.node;
          version          IDREF     #REQUIRED
          status           ( u | a ) 'a'
          govstd           CDATA     #IMPLIED
          mfgcode          CDATA     #IMPLIED
          milspec          CDATA     #IMPLIED
          quantity         NUTOKEN   #REQUIRED
          unit-of-measure  NMTOKEN   #IMPLIED  >

<!-- The consum element employs the 'NODE' template from the
generic layer.  A "consum" contains applicability preconditons
and relational links to other elements.  The "consum" element
contains many attributes which identify what the consumable is
```

```
(govstd, mfgcode, milspec), and the amount required (quantity,
unit-of-measure) for accomplishing the task.  -->

<!ELEMENT consum-alts       - -     ( consum )+ >
<!ATTLIST consum-alts
          %a.node-alts; >

<!--  This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of equipment
elements.  -->

<!--  ****************************************************

                        Alert Declaration

      ****************************************************  -->

<!--  This element identifies an alert that may accompany a task
or step.  The 'type' attribute will identify the kind of alert,
either Warning, Caution, Note.  -->


<!ELEMENT alert   - -  ( precond*, link*, (%text;)+,
                       (%graphic;)* ) >
<!ATTLIST alert
          %a.node;
    version       IDREF#REQUIRED
    status        ( u | a )'a' >

<!--  The alert element employs the 'NODE' template from the
generic layer.  An "alert" contains applicability preconditions,
relational links, text elements which make up the content of the
alert message, and optional "graphic" icons to be displayed.  -->

<!ELEMENT alert-alts       - -     ( alert )+ >
<!ATTLIST alert-alts
          %a.node-alts;   >

<!--  This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of alerts.  -->


<!--  ****************************************************

                        Step Declaration

      ****************************************************  -->
```


shall be discontinued until the busy state is cleared. If BUSY

```
<!-- Steps are the primary component of a maintenance procedure.

They describe the actions to be performed in order to
successfully complete the task.   -->

<!ELEMENT step     - -  ( precond*, link*, (%alert;)*,
                        (%sub-prims;)*, step-seq?, postcond* ) >
<!ATTLIST step
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a )'a'
          esttime NUTOKEN#IMPLIED  >

<!-- The step element employs the 'NODE' template from the
generic layer.  A "step" contains a list of preconditions which
delimit the step's applicability, relational links, precautionary
alerts, an optional sequence of substeps, and a list of
postconditions which define the state changes to be made after
the step is accomplished. -->

<!ELEMENT step-alts       - -     ( step )+ >
<!ATTLIST step-alts
          %a.node-alts;     >

<!-- This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of steps.
  -->

<!ELEMENT step-seq  - -  ( step | step-alts | if-step |
                           loop-step | task | task-alts )+ >
<!ATTLIST step-seq
          %a.node-seq;     >

<!-- This element employs the 'NODE SEQ' template from the
generic
layer.  It provides the capability to create sequences of steps.

    -->

<!ELEMENT if-step  - -  ( expression, step-seq, step-seq? ) >
<!ATTLIST if-step
           %a.if-node;   >

<!-- This element employs the 'IF NODE' template from the
generic layer.  It allows conditional selection of steps
depending on a precondition.  -->
```

```
<!ELEMENT loop-step  - -  ( assertion?, expression, assertion?,
                            step-seq ) >
<!ATTLIST loop-step
          %a.loop-node;     >

<!-- This element employs the 'LOOP NODE' template from the
generic layer.  It provides the capability of looping through a
sequence of steps.  -->


<!--  ************************************************

                    Follow on Declaration

      ************************************************  -->

<!--    A follow on condition is a maintenance
condition which must be accomplished sometime following the
completion of a task to clean up or undo actions performed during
the task.  For example, in order to fix a component a task might
require that an access panel be removed.  The panel would then
need to be replaced as a follow on action.  This task might be
performed sometime after the repair task is completed, but not
immediately after the repair task.  Other maintenance tasks might
be performed in the same area before the follow on task is
accomplished. -->


<!ELEMENT follow-on    - -  ( precond*, link*, (%text;)?,
                            ( expression, ( %task; | %step; ),
                              assertion* ), postcond* ) >
<!ATTLIST follow-on
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a )'a' >

<!-- The "follow-on" element employs the 'NODE' template from
the generic layer.  A "follow-on" element contains a set of
preconditions which define the follow on maintenance condition
which must be satisfied, relational links, an optional text
element which describes the follow on condition, a list of
task(s)/step(s) which provide instructions for accomplishing the
follow on condition, and a set of postconditions which define the
state changes to be made once the follow on condition is
accomplished.     -->

<!ELEMENT follow-on-alts       - -     ( follow-on )+ >
<!ATTLIST follow-on-alts
```

```
          %a.node-alts;  >

<!--  This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of follow-on
elements.  -->

<!--  ************************************************

                      Parts Information

      ************************************************  -->

<!--  "Partinfo" describes the maintainer's view of the part
information. Each "partinfo" element is related to a "partbase."
However, several "partinfo" items could be related to the same
"partbase."  -->


<!ELEMENT partinfo    - -  ( precond*, link*, (%partinfo;)*,
                           (%partbase;)+, (%connection;)*,
                           (%attach-part;)*, (%text;)?,
                           (%graphic;)*, (%location;)* ) >
<!ATTLIST partinfo
          %a.node;
          version          IDREF      #REQUIRED
          status           ( u | a )  'a'
          indexnum         NUTOKENS   #IMPLIED
          lru              NUTOKEN    #IMPLIED
          mtbf             CDATA      #IMPLIED
          refdes           NMTOKEN    #IMPLIED
          replvl           CDATA      #IMPLIED
          unitsper         NUTOKEN    #IMPLIED
          usablon          NUTOKENS    #IMPLIED  >

<!--  The "partinfo" element employs the 'NODE' template.  A
"partinfo" element contains a list of preconditions, relational
links, and alternate parts information ( the "partinfo" in the
content model).  "Partinfo" also identifies the components of the
part (partbase), any connecting parts (connection), attaching
parts (attach-part), a formal name for the part (text), a picture
of the part (graphic), and the location of the part in reference
to the weapon system (location).   -->

<!ELEMENT partinfo-alts       - -     ( partinfo )+ >
<!ATTLIST partinfo-alts
          %a.node-alts;   >

<!--  This element employs the 'NODE ALTS' template from the
```

```
generic layer to facilitate the context filtering of part
information elements.  -->

<!--  ***************************************************

                        Partbase Declaration

      ***************************************************  -->

<!--  "Partbase" describes the supply system's view of the part
information.  It describes the item in terms of its part number
('partnum').  -->


<!ELEMENT partbase      - -  ( precond*, link*, (%partbase;)*,
                             (%text;)?, (%location;)* ) >
<!ATTLIST partbase
          %a.node;
          version          IDREF#REQUIRED
          status           ( u | a )'a'
          cage             NUTOKENS #REQUIRED
          fsc              CDATA    #REQUIRED
          partnum          CDATA    #REQUIRED
          smr              CDATA    #REQUIRED
          nsn              CDATA    #IMPLIED
    pmic             CDATA              #IMPLIED
    cac              NUTOKEN            #IMPLIED
    qpei             NUTOKEN          #IMPLIED
    hci             (Y1|N1)      "N1"
    lox             (Y2|N2)      "N2"
    esds            (Y3|N3)      "N3"
    qec             (Y4|N4)      "N4"
    magnetic        (Y5|N5)      "N5"  >


<!--  The "partbase" element employs the 'NODE' template from the
generic layer.  It allows for the declaration of preconditions
for partbase information and relational linking to other
information from the partbase element.  -->

<!ELEMENT partbase-alts       - -     ( partbase )+ >
<!ATTLIST partbase-alts
          %a.node-alts;   >

<!--  This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of part base
elements.  -->
```

```
<!--  ********************************************************

          Connecting and Attaching Parts Declaration

      ********************************************************  -->

<!ELEMENT connection    - -  ( precond*, link*, (%partinfo;)+ ) >
<!ATTLIST connection
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a)'a' >

<!-- The connection element employs the 'NODE' template.  It
defines a connection between two "partinfo" elements.  -->

<!ELEMENT connection-alts       - -     ( connection )+ >
<!ATTLIST connection-alts
          %a.node-alts;   >

<!-- This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of connection
elements.  -->

<!ELEMENT attach-part  - -    ( precond*, link*, (%partinfo;)+ )
>
<!ATTLIST attach-part
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a )'a' >

<!-- The attaching part element employs the 'NODE' template.  It
defines the attaching parts for a "partinfo" element.  -->

<!ELEMENT attach-part-alts       - -     ( attach-part )+ >
<!ATTLIST attach-part-alts
          %a.node-alts;   >

<!-- This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of attach-part
elements.  -->

<!--  ********************************************************

                      Location Declaration

      ********************************************************  -->

<!-- The location element provides information for physical
```

assessment. It will contain x, y, z location(s) for a system with respect to the x, Fuselage Station (FS), y, Buttock Line (BL), and z, Water Line (WL) reference system. Where appropriate BL may be replaced by Wing Station (WS). -->

```
<!ELEMENT location        - -  ( precond*, link* ) >

<!ATTLIST location
          %a.node;
          version         IDREF#REQUIRED
          status          ( u | a )'a'
          location-x      NUTOKENS#IMPLIED
          location-y      NUTOKENS#IMPLIED
          location-z      NUTOKENS#IMPLIED >
```

<!-- The location element employs the 'NODE' template from the generic layer. It allows for the declaration of preconditions for a physical location and relational linking to other information from the location element. -->

```
<!ELEMENT location-alts     - -      ( location )+ >
<!ATTLIST location-alts
          %a.node-alts;        >
```

<!-- This element employs the 'NODE ALTS' template from the generic layer to facilitate the context filtering of location elements. -->

<!-- **********

Fault Information Declaration

********** -->

<!-- The "faultinf" element identifies all the fault isolation information associated with a system. "Faultinf" can be used to support dynamic troubleshooting models or static troubleshooting trees. -->

```
<!ELEMENT faultinf  - -  ( precond*, link*, (%test;)+,
                         (%fault;)* ) >
<!ATTLIST faultinf
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a )'a' >
```

```
<!-- The faultinf element employs the 'NODE' template.  It
contains a list of preconditions, relational links to other
elements, and lists of tests and faults associated with a
system.  -->


<!ELEMENT faultinf-alts     - -     ( faultinf )+  >
<!ATTLIST faultinf-alts
          %a.node-alts; >

<!-- This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of fault
information elements.  -->

<!--  ******************************************************

                        Test Declaration

      ******************************************************  -->

<!-- The "test" element identifies a prescribed task to perform
and is the usual way of entering the troubleshooting process.
The result of a test is an outcome; a test will have two or more
outcomes.  -->


<!ELEMENT test          - -  ( precond*, link*, (%task;),
                             (%outcome;)+ ) >
<!ATTLIST test
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a )'a'
          agent   CDATA"human"
          range   CDATA#IMPLIED  >

<!-- This element identifies the task needed to complete the
test and all the possible outcomes as a result of the test.  -->

<!-- The test element employs the 'NODE' template.  It contains
a list of preconditions and relational links to other
information.  A "test" element identifies the task which will
accomplish the test.  All the possible outcomes are contained
within the test.
-->

<!ELEMENT test-alts     - -     ( test )+  >
<!ATTLIST test-alts
          %a.node-alts;  >
```

```
<!--  This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of tests.  -->

<!--  ****************************************************

                        Outcome Declaration

      ****************************************************  -->

<!--  This element identifies a result of a test.  The
precondition list is evaluated against the result of the test,
and the appropriate outcome is selected.  In a dynamic
troubleshooting model, the outcome will contain a faultstate that
identifies an implicated or exculpated set of faults.  In a
static troubleshooting model, the outcome will contain another
test or a fault.  The outcome will contain the information
necessary for both models, but it will be up to the diagnostic
software to choose the correct path to follow for its logic.  -->

<!ELEMENT outcome - -  ( precond*, link*, expression,
                         ( (%fltstate;) | ( ( %test; | %fault; ),
                         (%fltstate;)? ) ) ) >
<!ATTLIST outcome
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a )'a' >

<!--  The outcome element employs the 'NODE' template.  It
contains a list of preconditions, and relational links to other
information.  The faultstate element will identify the implicated
or exculpated faults for the outcome.  The test and rectification
elements identify the next step in a static fault tree.  -->

<!ELEMENT outcome-alts    - -    ( outcome )+  >
<!ATTLIST outcome-alts
          %a.node-alts;   >

<!--  This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of outcomes.
-->

<!--  ****************************************************

                      Faultstate Declaration

      ****************************************************  -->

<!--  The "fltstate" element identifies a set of implicated or
```

```
exculpated faults.  Implicated faults are faults suspected of
being bad; exculpated faults are faults known to be good.  Each
implicated fault will have an associated weight based on its
likelihood of causing the discrepancy.  -->

<!ELEMENT fltstate  - -  ( precond*, link*, (%fault;)+ ) >
<!ATTLIST fltstate
          %a.node;
          version         IDREF#REQUIRED
          status          ( u | a )'a'
          weight          NUTOKENS        #IMPLIED >

<!--  The fltstate element employs the 'NODE' template.  It
contains a list of preconditions, and relational links to other
appropriate information.  The 'type' attribute will designate
whether the list of faults are "implicated" or "exculpated." -->

<!ELEMENT fltstate-alts     - -     ( fltstate )+  >
<!ATTLIST fltstate-alts
          %a.node-alts; >

<!--  This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of fault
states.  -->

<!--  ******************************************************

                        Fault Declaration

      ******************************************************  -->

<!--  The "fault" element identifies the cause of a discrepancy
on the weapon system.  The fault will identify the appropriate
rectification to correct the discrepancy.  When transitioning
between maintenance levels the fltstate element is used.  -->

<!ELEMENT fault   - -  ( precond*, link*, ( %rect; | %fltstate;
)+,                          (%system;)+ ) >
<!ATTLIST fault
          %a.node;
          version IDREF#REQUIRED
          status  ( u | a )'a'
          mtbf    CDATA#IMPLIED  >

<!--  The fault element employs the 'NODE' template.  It contains
a list of preconditions, and relational links to other
appropriate information.  The rectifications contain tasks which
will correct the discrepancy.  The system and part information
```

```
elements will create a back link to the part that has failed.
The "fltstate" represents the system at the next level of
maintenance.  -->

<!ELEMENT fault-alts    - -    ( fault )+  >
<!ATTLIST fault-alts
          %a.node-alts;  >

<!--  This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of faults.  -->

<!--  ****************************************************

                    Rectification Declaration

      ****************************************************  -->

<!--  The "rect" element identifies the prescribed task that will
repair the fault causing the discrepancy and all other faults
that could be fixed by the rectification.  Upon completion of the
task, a test is performed to verify the effect of the
rectification.
-->

<!ELEMENT rect    - -  ( precond*, link*, (%task;)+, (%fault;)+,
                         (%system;),  (%test;)* ) >
<!ATTLIST rect
          %a.node;
          version IDREF                 #REQUIRED
          status  ( u | a )             'a'
          action  ( swap | maint )      "swap"
          agent   CDATA                 "human"  >

<!--  The rect element employs the 'NODE' template.  It contains
a list of preconditions, and relational links to other
appropriate information.  The "system" element provides a
reference to the system which will be repaired by the
rectification.  The test element identifies all check-out tests
required before completing the maintenance session.  -->

<!ELEMENT rect-alts    - -    ( rect )+  >
<!ATTLIST rect-alts
          %a.node-alts;    >

<!--  This element employs the 'NODE ALTS' template from the
generic layer to facilitate the context filtering of
rectifications.  -->
```

GENERIC LAYER
TAG SET DESCRIPTIONS

10. SCOPE.

10.1 Scope. This appendix provides the detailed description of the elements and attributes to be included in an IETMDB. It is formulated as a description of possible tags or names for components in an IETMDB whose structure is defined by the generic layer DTD specified within Appendix A of this specification. Unless otherwise specified by the procuring activity, this Appendix is a mandatory part of this specification. The information contained herein is intended for compliance.

20. APPLICABLE DOCUMENTS.

20.1 Government documents.

20.2 Non-Government publications. The following documents form a part of this document to the extent specified herein. Unless otherwise specified, the issues of the documents which are DoD adopted are those listed in the issue of the DODISS cited in the solicitation. Unless otherwise specified, the issues of documents not listed in the DODISS are the issues of the documents cited in the solicitation.

ISO 8879 Information Processing - Text and Office Systems - Standard Generalized Markup Language (SGML)

ISO/IEC IS10744:1992
Information Technology - Hypermedia/Time-based Document Structuring Language (HyTime)

(Application for copies should be addressed to the American National Standards Institute, 1430 Broadway, New York, NY 10018.)

30. GENERIC LAYER TAG SET DESCRIPTIONS.

30.1 Use of SGML. The markup tags described herein conform to rules defined in ISO 8879.

30.2 Tag Set Descriptions. Data elements shall be defined in accordance with the tag set descriptions included below (see 3.1.2).

AGENT Dialog Agent

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in the **dialog** element, this attribute defines to whom the question is asked. The value of this attribute contains character data which identifies the person or computer to whom the dialog should be presented. The default value is **'human'**.

ASSERTION **Assertion**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element is used to make an assertion from within the content model of an application specific element. Whenever an assertion appears in an element's content model, there shall be set of semantic rules describing when the assertion is to be evaluated. For example, under required conditions the assertion is only evaluated when the user decides to skip a task reference.

AUDIO **Audio Sequence**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: This element is used to hold an audio sequence.

BINOP Binary Operation

Descriptor: EntityFormat: N/A
Template Used: N/A

Description: This entity enumerates all of the possible binary operators which may be used within an expression. This element must contain one of the following elements: **eq, ne, lt, gt, le, ge, and, or, xor, concat, substring, append, plus, minus, times, divide, idivide, exponent, mod, remove, union, intersect, set-diff, member, subset, disjoint, add, subsequence.**

CDM **CDM Template Type**

Descriptor: AttributeFormat: NAME
Template Used: N/A

Description: Used in all element declarations, to identify the generic template which the element follows. The attribute's value is a fixed default value (ie. cannot be changed by entry of another value). It is set to '**node**' if the element follows the '**node**' template. It is set to '**node-alts**' if the element follows the '**node alts**' template. It is set to '**node-seq**' if the element follows the '**node seq**' template. It is set to '**if-node**' if the element follows the '**if node**' template. It is set to '**loop-node**' if the element follows the '**loop node**' template.

CHOICE **Choice**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element defines a choice in a menu. Choices consist of a piece of text or a graphic to be displayed. Once the user selects a choice from a menu, the presentation system will either assert some postcondition or will branch to another **dialog** (which could contain another **menu**, **fillin** or **selection**).

CODING **Coding**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used by the **grphprim** element, this attribute identifies the particular storage type of the current graphic file (e.g. IGES, CGM). The default value is '**cgmbin**'.

COLHDDEF **Column Header Definition**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element defines a column header for a specific column of tabular information.

COLNUM **Column Number**

Descriptor: AttributeFormat: NUTOKEN
Template Used: N/A

Description: Used by the **colhddef** and **entry** elements, the value of this attribute consists of the column number of a table.

DEFAULT **Default Indicator**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used by the **choice** element, this attribute contains an enumerated list with values of either a '**Yes**' or '**No**'. The **default** attribute provides a method of indicating whether a **choice** is designated as a default for the **menu**. The default value for this attribute is '**No**'.

DIALOG **User Interactive Dialogs**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: This element provides the capability for user interaction. A **dialog** could contain a sub**dialog**, **fillin**, **menu**, **selection**, or any combination of the four. It may also contain an optional text string which would be the title of the composite dialog.

DIALOG-REF **Dialog Reference**

Descriptor: AttributeFormat: IDREF
Template Used: N/A

Description: Used in the **property** element, this attribute contains the ID of either a **dialog** element or a **process** element which will acquire a value for the **property**, if **property** is undefined (ie., equal to '**nil**') at presentation time.

ENTRY **Column Entry Definition**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element defines an entry for a cell in a table. An **entry** is a piece of text and a column number.

EXPRESSION **Expression**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: The **expression** element provides the capability to create mathematical expressions to be used for **preconditions** and **postconditions.** There can be one of four types of expressions: a binary operation between two expressions, a unary operation with an expression, a property, or a value.

EXTERNAL-PTR **External Process Pointer**

Descriptor: Attribute Format: IDREF
Template Used: N/A

Description: Used in the **audio, video** and **process** elements, this attribute is a pointer which points to an external file. The external file shall contain the appropriate audio, video or software process that will present to the user a multimedia event.

FILLIN **Fill In The Blank Question**
Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element defines a fill in the blank question. It will contain a **prompt**, a **property**, and an optional default value. The **prompt** contains the question to be presented to the user. The **property** element identifies the variable which will receive a value from the user's response. The **property** element also identifies the legal value type of the user's response. The **fillin** will be presented to the user according to the value type.

GENERIC-RANGE Generic Range

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element provides a mechanism for defining valid range checking for fillin elements. The element may identify a maximum and minimum for numeric entries or a set of valid values that may be entered for an alpha numeric entry.

GRAPHIC Graphic

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: This element represents a composite graphic which is made up of graphic primitives (**grphprim**) or other graphic components (**graphic**).

GRPHPRIM Graphic Primitives

Descriptor: ElementFormat: N/A
Template Used: Node, Node-alts

Description: This element defines a graphic primitive to be a single graphic component which, when combined with other primitives, can become a composite graphic. A graphic primitive references a file that contains the detailed graphic information in some standard (e.g., CGM, IGES, FAX, or DXF graphic codes).

HIGH-BOUND High Bound

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element identifies the maximum allowable number for a numeric entry of a fillin.

HYTIME **Hytime**

Descriptor: AttributeFormat: NAME
Template Used: N/A

Description: Used by the **link** element, this attribute is a fixed default value (ie. cannot be changed by entry of another value). It is set to '**ilink**'.

ID **Identifier**

Descriptor: AttributeFormat: ID
Template Used: N/A

Description: Used by elements to hold a unique identifier of a specific element.

IF-NODE **If Node Template**

Descriptor: EntityFormat: Template
Template Used: N/A.

Description: The **if node** template provides a method of conditional branching within an interactive sequence. This template uses the same logic as the IF-THEN-ELSE statement in a programming language.

INDEX **Index**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: A signed integer value. Its meaning is dependent upon its sign. A positive value means an index position from the beginning of a **string** or **sequence.** A negative number means an index position counted back from the end of the **string** or **sequence.** A zero means the end of the string.

ITEMID **Item Identification**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in all **node** elements to identify the components of the system being repaired, as they relate to information elements. The item identification attribute specifies the reference designator(s) or other identifiable designator(s) of the system(s), subassemblies, or parts referred to by the element. The permissible values of this attribute are dependant upon the content specific application using this primitive.

LINK **Link**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element provides the capability for creating relational links with the data. It employs the HyTime "**ilink**" architectural form (template) and may contain "anchors" called (location elements) to identify two or more **linkends.** The **link** element may contain the name of the relation (e.g., **linkterm**).

LINKEND **Link End**

Descriptor: AttributeFormat: IDREFS
Template Used: N/A

Description: Used by the **link** element, this attribute contains one or more unique identifiers (IDREFs). The identifiers shall point to a CDM element or a location element which resolves at the desired data.

LINKTERM **Link Term**

Descriptor: AttributeFormat: IDREF
Template Used: N/A

Description: Used by the **link** element, this attribute contains the ID of a specific element. This attribute points to a data item which will be used to identify a link during presentation.

LOOP-NODE **Loop Node Template**

Descriptor: EntityFormat: Template
Template Used: N/A.

Description: The **loop node** template allows for the creation of iterative loops within an interactive sequence (**node-seq**) of elements.

LOW-BOUND **Low Bound**

Descriptor: ElementFormat: N/A
Template Used: N/A.

Description: This element is used to identify the minimum allowable entry for a numeric fillin.

MENU **Menu**

Descriptor: ElementFormat: N/A
Template Used: Node

Description: This element defines a **menu** for user interaction. It consists of a **prompt** followed by one or more **choice** elements.

MINSIZE **Minimum Size**

Descriptor: AttributeFormat: NUTOKENS
Template Used: N/A

Description: Used in the **graphic** and **grphprim** elements, the **minsize** attribute specifies the minimum viewing size at which the graphic should be displayed. The minimum is expressed as the width (in inches) at which the graphic should be displayed, assuming a 36 inch viewing distance.

MODE **Mode**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in the **parameter** element, this attribute is composed of character data containing permissible values of either '**in**','**out**', or '**in-out**'. The default value is '**in**'. It will indicate the method of parameter passing between the technical information and the software process.

NAME **Name**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in all **node** elements, this attribute holds the standard nomenclature for the element expressed as character data. The permissible values of this attribute depend on the specific element type.

NIL **An Empty Element**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element represents an undefined value. Any property can take on the **nil** value.

NODE **Node Template**

Descriptor: EntityFormat: Template
Template Used: N/A.

Description: The **node** is a template by which technical information is defined. The **node** template contains the "content" of the technical information. The **node** template creates hierarchy within the CDM. The **node** template also contains context filtering preconditions and postconditions. The **link** element within the **node** template provides the capability to cross reference to other technical information. The use of **link**, from the Hytime model, provides additional functionality by allowing a link to be made to a document outside the CDM specification boundary.

The **node** template provides the capability to create composite structures within the content specific layer. Composite structures may contain subcomponents that employ the **node, node alts,** or **node seq** templates. The **node** subcomponents may be composite structures themselves or they may be primitive nodes (**text, tables, graphics, audio, video, process**). Composite structures create hierarchy within the CDM. When composite nodes contain other composite nodes there is an implied hierarchy. The composite node in the content model is at a lower level in the hierarchy (e.g. a **task node** contains **step nodes** in its content model).

NODE-ALTS **Node Alternatives Template**

Descriptor: EntityFormat: Template
Template Used: N/A.

Description: This template shows you how to create context sensitive filtering. This element contains one or many elements using the **node** template. **Node-alts** (node alternatives) will contain a list of mutually exclusive nodes. Their grouping is due to the fact that they apply in different contextual situations. In this manner, the **node-alts** element is a logical reference that contains a set of **nodes** which might apply to different situations. An important fact in the **node-alts** structure is that no hierarchy is implied between the generic identifier and the content model **nodes** (e.g. a **task-alts** element will contain **task** nodes in its content model).

NODE-SEQ **Node Sequence Template**

Descriptor: EntityFormat: Template
Template Used: N/A.

Description: The **node seq** template provides the structure for creating interactive sequences with the user. The **node seq** template provides the capability to not only group elements together, but also to preserve any inherent order/sequence which may apply to the technical information. The **node seq** template also allows an author to define conditional branching and iteration within the technical information.

NUM-RANGE **Number Range**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element contains the maximum and minimum allowable values for a fillin.

PARAMETER **Parameter**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element will be used to pass parameters to or from an external software **process.** For example, the 1553 bus on an aircraft might require parameters concerning a given channel which requires look up. The **parameter** element will contain the channel required by the process.

PENPATT **Pen Pattern**

Descriptor: AttributeFormat: Character Date
Template Used: N/A

Description: Used in **graphic** and **grphprim** elements, this attribute represents the bit map pattern to be used as the pen for drawing lines, points, etc. for a particular **graphic.**

PENSHAPE **Pen Shape**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in **graphic** and **grphprim** elements, this attribute indicates the boundary shape for the pen for drawing lines, points, etc. for a particular **graphic.**

POSTCOND **Post Condition**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: The **postcond** element asserts the value of an **expression** to a **property** when the display system software presents a **dialog** node to the user, or when a user completes some action which needs to be recorded for later context filtering.

PRECOND **Precondition**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: A **precond** element must contain an expression which identifies the conditions which must be present to display the technical information.

PROCESS **External Software Process**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: This element will be used to represent an external software process.

aircrew station design shall be used

PROMPT **Prompt**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element defines the prompt to be displayed to the user for the presentation of a **fillin** or a **menu**. It allows the prompt to be either a text string (in the form of a question) or a graphic (a picture which requires an answer).

PROPERTY **Property**

Descriptor: ElementFormat: Parsable Character DataTemplate Used: N/A

Description: This element contains parsable character data which represents the property (variable) name. The value of a property may be obtained by finding the current value associated with the property name in the state table.

REF **Reference**

Descriptor: AttributeFormat: IDREF
Template Used: N/A

Description: Used in many elements, this attribute contains the ID of a specific element. The **ref** attribute utilizes the SGML #CONREF capability. A #CONREF attribute is only filled in when the element's content model is empty. In this case, the #CONREF attribute contains a reference which is a unique identifier to either an element employing the appropriate template or a location element that resolves to an element employing the appropriate template (see ISO/IEC IS10744).

REMOVE **Remove**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: For a **set** element the **remove** element returns a **set** with **value** removed. For a **sequence** using the binary operand form it returns a **sequence** which has the first instance of **value** removed. For a **sequence** or **string** as a unary operator remove must contain an **index value** which refers to the position from which the character in the **string** is to be removed or the value in the **sequence** is to be removed. The new string or sequence will be the old one up to but not including the index position concatenating with the old one after the index position.

ROW **Row Number**

Descriptor: AttributeFormat: NUTOKEN
Template Used: N/A

Description: Used in the **entry** element, the value of this attribute consists of the row number for that entry's tabular information.

ROWHDDEF **Row Header Definition**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element defines a row header for a specific row of tabular information.

SELECT **Select**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in the **menu** element, this attribute allows the author to designate the number of choices that may be selected by the user. The choices are either '**single**' or '**multiple**', with the default selection choice being '**single**'.

SELECTION **Selection**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element provides the capability of creating a special menu that allows selection within a given picture, text string or table.

SEQUENCE **Sequence**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element is defined as being an ordered sequence of data.

SET **Set**

Descriptor: ElementFormat: N/A
Template Used: N/A

Description: This element is defined as being an unordered sequence of data.

TABLE **Table**

Descriptor: ElementFormat: N/A
Template Used: Node, Node-alts

Description: This element defines how a table is constructed. A **table** will contain a column header followed by one or more entries. The combination of column header and entries may be repeated for as many columns as the **table** requires.

TEXT **Text**

Descriptor: ElementFormat: Parsable Character DataTemplate Used: Node, Node alts

Description: This element defines how text is constructed. Within a text string, there may be embedded **text** elements which allow the referencing of other elements or parts of elements through the link/location mechanism of HyTime.

TRANSFRM **Transformation Matrix**

Descriptor: AttributeFormat: NUTOKEN
Template Used: N/A

Description: Used in the **graphic** and **grphprim** elements, this attribute signifies a transformation matrix which specifies coordinate translations, scaling, or reflection and rotations in terms of homogenous coordinates.

TYPE **Type**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in all **node** elements, the information type attribute provides a more precise mechanism for classifying an element. The permissible values of this attribute are dependant upon the content specific application using this primitive.

UNOP **Unary Operator**

Descriptor: Entity Format: N/A
Template Used: N/A

Description: This entity enumerates all of the possible unary operators which may be used within an expression. This element could contain the following: **not, empty, size, head, tail, neg, remove, trunc, float, index, undef, max,** and **min.**

VALUE **Value**

Descriptor: EntityFormat: N/A
Template Used: N/A

Description: This entity defines an expression value. A **value** may be a **boolean, string, sequence, set, real, integer,** or **nil.**

VALUE-TYPE **Value Type**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in the **property** element, this attribute is used to denote the allowable data types which may be assigned to the property. The current legal values are any combination of the following: '**boolean**', '**integer**', '**real**', '**set**', '**sequence**', '**string**', and '**general**'. The default value is '**general**'.

VIDEO **Video Sequence**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: This element will be used to include a video sequence into technical information.

WINDOW **Window**

Descriptor: AttributeFormat: NUTOKENS
Template Used: N/A

Description: Used in the **graphic** and **grphprim** elements, this attribute indicates the subrectangle within a graphic which should be displayed in those cases where the author wishes to display only a portion of a large graphic to the user.

A CONTENT SPECIFIC LAYER TAG SET DESCRIPTIONS

10. SCOPE.

10.1 Scope. This appendix provides the detailed description of the IETM content specific elements and attributes to be included in an IETMDB. It is formulated as a description of possible tags or names for components in an IETMDB whose structure is defined by a DTD specified or developed in accordance with this specification. Unless otherwise specified by the procuring activity, this Appendix is a mandatory part of this specification. The information contained herein is intended for compliance.

20. APPLICABLE DOCUMENTS.

20.1 Government documents.

20.2 Non-Government publications. The following documents form a part of this document to the extent specified herein. Unless otherwise specified, the issues of the documents which are DoD adopted are those listed in the issue of the DODISS cited in the solicitation. Unless otherwise specified, the issues of documents not listed in the DODISS are the issues of the documents cited in the solicitation.

ISO 8879	Information Processing - Text and Office Systems - Standard Generalized Markup Language (SGML)
ISO/IEC IS10744	Information TechnologyHypermedia/TimebasedDocument Structuring Language (HyTime)

(Application for copies should be addressed to the American National Standards Institute, 1430 Broadway, New York, NY 10018.)

30. CONTENT SPECIFIC LAYER TAG SET DESCRIPTIONS.

30.1 Use of SGML. The markup tags described herein conform to rules defined in ISO 8879.

30.2 Tag set descriptions. Data elements shall be defined in accordance with the tag set descriptions included below (see 3.1.2).

ACTION **Action**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in the **rect** element, this attribute contains character data describing the type of maintenance action required to rectify, or fix, a fault. The action can be a '**swap**', which means it is a removal/replacement action, or it can be a '**maint**' action, which means it is an adjustment, alignment, or similar action. The default value is '**swap**'.

AGENT **Agent**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in the **rect** and **test** element, this attribute contains character data describing who performs a maintenance action. It can be either a '**human**' agent, or some valid computer system (e.g., 1553 bus) called '**machine**'. The default value is '**human**'.

ALERT **Alert**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: This element identifies an alert that may accompany a task or a step. The **type** attribute may designate an alert to be a warning, caution or note which may be displayed to the technician. A warning notifies the technician that a task or step may be harmful to himself or another human if not properly performed. A caution is used in technical information to emphasize a procedure that, if not strictly followed, or a condition that, if not strictly maintained, may result in damage to the equipment. A note signifies additional information which aids the technician in completing the step or task. A note is used in technical information to emphasize an especially important procedure or condition.

ATTACH-PART **Attaching Part**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: This element identifies all the attaching parts required for a given part information element.

CAGE **Commercial and Government Entity**

Descriptor: AttributeFormat: NUTOKENS
Template Used: N/A

Description: Used in the **consum, equip** and **partbase** elements, this attribute is a five character code assigned by the Defense Logistics Services Center (DLSC) to the design control activity or actual manufacturer of an item contained in the Cataloguing Handbook H4/H8 series.

CONNECTION **Connecting Part**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: This element is used to identify a connection between two part information elements (e.g., a connection between pin 123 and wire ABC).

CONSUM **Consumables**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: This element identifies all the consumable required for the completion of the task.

DESCINFO **Descriptive Information**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: The element **descinfo** is used to define general purpose, non-procedural, narrative information such as theory of operation, schematics, etc which are associated with a system component. The **descinfo** element is very flexible. It can be used to describe any arbitrary, hierarchical hypertext like node.

ESTTIME **Estimated Time**

Descriptor: AttributeFormat: NUTOKEN
Template Used: N/A

Description: Used in the **task** and **step** elements, the value of this attribute indicates the amount of time, in minutes, required for the corresponding task/step to be completed.

EQUIP **Equipment**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: An **equip** element identifies the equipment needed to perform a particular **task**. **Equip** usually refers to a piece of test equipment, support equipment, or a tool.

FAULT **Fault**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: The element **fault** is used to identify a potential failure which may occur on a weapon system.

FAULTINF **Fault Information**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: The **faultinf** element is used to define all the tests and faults associated with the system that references it.

FLTSTATE **Fault State**

Descriptor: ElementFormat: N/A
Template Used: Node, Node alts

Description: The **fltstate** element identifies a set of implicated or exculpated faults. Implicated faults are faults suspected of being bad; exculpated faults are faults known to be good. Each implicated fault will have a weight associated base on its likelihood of causing the discrepancy. The '**type**' attribute will designate whether the list of faults are '**implicated**' or '**exculpated**'.

FOLLOW-ON **Follow On Conditions**

Descriptor: ElementFormat: N/A
Template Used: Node
Node-alts

Description: A follow on condition is a maintenance condition which must be accomplished sometime following the completion of a task to clean up or undo actions performed during the task. For example, in order to fix a component a task might require that an access panel be removed. The panel would then need to be replaced as a follow on action. This task might be performed sometime after the repair task is completed, but not immediately after the repair task. Other maintenance tasks might be performed in the same area before the follow on task is accomplished. A **follow-on** element contains a set of preconditions which define the follow on maintenance condition which must be satisfied, relational links, a text element which verbally describes the follow on condition, a list of task(s)/step(s) which provide instructions for accomplishing the follow on condition, and a set of postconditions which define the state changes to be made once the follow on condition is accomplished.

FSC **Federal Stock Classification**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in the **partbase** element, the value of this attribute contains applicable Federal Stock Classification (FSC) codes.

GOVSTD **Government Standard**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in the **consum** element, the value of this attribute signifies a document that establishes engineering and technical requirements for processes, procedures, practices, and methods that have been adopted as standards. It also establishes requirements for selection, application, and design criteria for materials.

HCI **Hardness Critical Item**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in the **partbase** element, the value of this attribute represents a code which indicates that an item could degrade system survivability in a nuclear, biological, or chemically hostile environment if hardness were not considered.

ICC **Item Category Code**

Descriptor: AttributeFormat: Character Data
Template Used: N/A

Description: Used in the **equip** and **consum** elements, the value of this attribute signifies a code which identifies a type of item, and indicates categories into which support and test equipment, spares, repair parts, etc. may be divided.

Note: ICCs of "A," "B," and "C" should not be assigned to hardware items: these codes are reserved for grouping and selecting similar ICCs during automated data processing.

Peculiar Support Equipment and Tools not Currently in the DOD Inventory (ICC Group A):

Peculiar Support Equipment (Other)	7
Peculiar Tools	8
Peculiar Test Equipment	M
Peculiar Handling Equipment	D
Peculiar Automatic Test Equipment (ATE)	1

Common Support Equipment and Tools Currently in the DOD Inventory (ICC Group B):

Common Support Equipment (Other)	H
Common Tools	4
Common Test Equipment	5
Common Handling Equipment	6
Common Automatic Test Equipment (ATE)	2

Common Support Equipment and Tools Currently in the DOD Inventory but not Assigned to a Unit/Ship (ICC Group C):

Common Support Equipment (Other)	G
Common Tools	N
Common Test Equipment	P
Common Handling Equipment	R
Common Automatic Test Equipment (ATE)	3
Bulk Items	Q
Training material not currently in the DOD inventory	S
Training material currently in the DOD inventory	T
End Item	W
Spare (repairable support item)	X
Repair part (a nonrepairable consumable support item, component, assembly)	Y
Repair Parts Kit	Z
A repair part, component or assembly that is contained in a kit/set	9
Tool Kit/Set	V
Program (Embedded software)	E
Tech Manuals	F
Forms or records	J
Electrostatic Discharge-Sensitive Item	K
Electromagnetic-Sensitive Item	L
Facilities	U

System-Peculiar Spare Part	AA
Maintenance Significant Consumable	AB
Modified Hand Tool	AC
Maintenance Assist Module	AD

INDEXNUM **Index Number**

Descriptor: Attribute Format: NUTOKENS
Template Used: N/A

Description: Used in the **partinfo** element, the value of this attribute contains the index number for the part which represents a callout in a graphic output onto paper.

INPUT **Input Conditions**

Descriptor: Element Format: N/A
Template Used: Node, Node alts

Description: The **input** element contains the personnel required, the consumable used, the equipment used and the required conditions for accomplishing a given task.

LOCATION **Part Location**

Descriptor: Element Format: N/A
Template Used: Node, Node alts

Description: The **location** element provides information for physical assessment. It will contain x, y, z location(s) for a system with respect to the x, Fuselage Station (FS), y, Buttock Line (BL), and z, Water Line (WL) reference system. Where appropriate BL may be replaced by Wing Station (WS).

LOCATION-X **Location X**

Descriptor: Attribute Format: NUTOKENS
Template Used: N/A

Description: Used in the **location** element, the value of this attribute contains a number representing a position on the Fuselage Station (FS), which is used as the x-axis of the weapon system.

LOCATION-Y **Location Y**

Descriptor: Attribute Format: NUTOKENS
Template Used: N/A

Description: Used in the **location** element, the value of this attribute represents a position on the Buttock Line (BL), which is used as the y-axis of the weapon system.

LOCATION-Z **Location Z**

Descriptor: Attribute Format: NUTOKENS
Template Used: N/A

Description: Used in the **location** element, the value of this attribute contains a number representing a position on the Water Line (WL), which is used as the z-axis of the weapon system.

LRU **Line Replaceable Units**

Descriptor: Attribute Format: Character Data
Template Used: N/A

Description: Used in the **partinfo** element, this attribute signifies an essential support item that is removed and replaced at field level to restore the end item to its operationally ready condition. Allowable values are:

Item is a LRU	Y
Item is not a LRU	N

MFGCODE **Manufacturers Codes**

Descriptor: Attribute Format: Character Data
Template Used: N/A

Description: Used in the **consum** element, the value of this attribute indicates the in house code a manufacturer uses to represent parts.

MILSPEC **Military Specification**

Descriptor: Attribute Format: Character Data
Template Used: N/A

Description: Used in the **consum** element, the value of this attribute represents the exact specification for each item bought by the government.

MTBF **Mean Time Between Failure**

Descriptor: Attribute Format: Character Data
Template Used: N/A

Description: Used in the **fault** and **partinfo** elements, the value of this attribute signifies, for a particular interval, the total functional life of a population of an item divided by the total number of failures within the population during the measurement interval. The definition holds for time, rounds, miles, events, or other measure-of-life units.

NOUNID **Noun Identifier**

Descriptor: Attribute Format: NUTOKEN
Template Used: N/A

Description: Used in the **partinfo** element, the value of this attribute indicates a general name of a part.

NOUNTYPE **Noun Type**

Descriptor: Attribute Format: NUTOKEN
Template Used: N/A

Description: Used in the **partinfo** element, the value of this attribute signifies more specific descriptors which differentiate part names.

NSN **National Stock Number**

Descriptor: Attribute Format: Character Data
Template Used: N/A

Description: Used in the **equip**, **consum** and **partbase** elements, the value of this attribute is a number, assigned under the Federal Cataloguing Program and/or North Atlantic Treaty Organization (NATO) codification of equipment system to each approved item, which provides a unique identification of an item of supply within a specified FSC. The field consists of a three character prefix, a thirteen character NSN, and a four character suffix code. For applicable codes, see DOD 4100.38-M.

OPERABILITY **Operability**

Descriptor: Attribute Format: Character Data
Template Used: N/A

Description: Used in the **task** element, the value of this attribute is a code used to indicate the operational status and mission readiness of the system during the maintenance task. Allowable values are:

Full Mission-Capable: performance of the maintenance task does not degrade any mission capability.	C
Partial Mission-Capable: performance of the maintenance task degrades the mission capability of the system, but can perform at least one mission.	D
System Inoperable During Equipment Maintenance: system is not available to perform all normal operations.	A
System Operable During Equipment Maintenance: system is available to perform normal operations.	B
Not Mission-Capable: system cannot perform any missions.	E
Off-Equipment Maintenance: task is performed after the item under analysis has been removed from the system.	G
Turnaround: task occurs during normal turnaround operations, and does not affect the operability of the system.	F

OUTCOME **Outcome**

Descriptor: Element Format: N/A
Template Used: Node, Node alts

Description: This element represents a possible outcome from a test. It identifies a fault state (**fltstate**) for use in a dynamic fault model, and a **test** or **fault** for the static tree model.

PARA **Para**

Descriptor: Element Format: N/A
Template Used: Node, Node alts, Node seq, If node, Loop node

Description: Identifies a piece of text to be displayed under the **descinfo** element. The text may be "Theory of Operation", "General Information", etc.

PARTBASE **Part Base**

Descriptor: Element Format: N/A
Template Used: Node
Node-alts

Description: This element describes the supply system's view of the part information. It describes the item in terms of its part number.

PARTINFO **Part Information**

Descriptor: Element Format: N/A
Template Used: Node, Node alts

Description: This element describes the maintainer's view of the part information. It identifies parts information within its relative position in the weapon system.

PARTNUM **Part Number**

Descriptor: Attribute
Template Used: N/A

Format: Character Data

Description: Used in the **partbase** element, this attribute signifies any number, other than a government activity stock number, used to identify an item of production or supply.

PERSON **Person**

Descriptor: Element
Template Used: Node

Format: N/A

Node-alts

Description: This element is used to identify the personnel requirements for a given task.

QUANTITY **Quantity**

Descriptor: Attribute
Template Used: N/A

Format: NUTOKEN

Description: Used in the **person, equip, expend** and **consum** elements, the value of this attribute signifies the amount of the appropriate consumable, equipment, or people required for the associated **task/step.**

RANGE **Range**

Descriptor: Attribute
Template Used: N/A

Format: Character Data

Description: Used in the **test** element, this attribute represents the boundaries for valid choices or outcomes, according to the element containing the range.

INDEX

	PARAGRAPH	PAGE
A Content Specific Document Type Definition ..		B-1
A Content Specific Layer Tag Set Description .		D-1
Acquisition of Final Form IETMs	1.3.2	2
Acquisition of Fully Constructed IETMs	1.3.2	3
Acquisition of IETMDB	1.3.2	3
Acquisition Requirements	6.2	12
Acronyms	6.4.1	13
Additional Content Specific DTDs	3.1.4	6
Appendix A		A-1
Appendix B		B-1
Appendix C		C-1
Appendix D		D-1
Applicable Documents	2.	4
Attributes	3.2.1.1	7
Audio, Video and Process	3.2.3.4	8
Capabilities (IETMDB)	1.3.1.1	2
CDM Attribute	3.2.1.1.4	7
Classified Material	5.1.2	12
Commercial and Government Entity (CAGE)	3.3.3.5.2	11
CONREF Attribute	3.2.2.1	8
Content Data Model (CDM)	3.1	5
Content Specific - Document Type Definition (DTD)	3.3.1	9 B-1
Content Specific DTD for Organizational Level (O-Level) Maintenance	3.3.3	10
Content Specific Layer	3.3	9
Context Dependent Filtering.....	3.2.4	9
Control of Content Specific DTDs	3.3.1	9
Data Base Interchange Requirements	3.1.1	6
Data Base Structuring and Data Element Naming Requirements	3.1.2	6
Data Maintainability	3.1.3	6
Data Portability	1.4.1	3
Data Requirements	6.3	13
Definitions of Acronyms and Terms	6.4	13
Delivery, Preparation for	5.1	12
Descriptive Information	3.3.3.2	10
Development of Content Specific DTDs	3.3.2	9
Dialog Elements	3.2.3.5	8
Dialogs	3.2.3.5	8
Digital Product Packaging	5.1.1	12

INDEX

	PARAGRAPH	PAGE
Expressions	3.2.4.3	9
Fault Element	3.3.3.4.1	10
Fault State Elements	3.3.3.4.2	10
Fillin	3.2.3.5	8
Filtering	3.2.4	9
Final Form IETMs, Acquisition of	1.3.2	2
Format Free Technical Information	1.4	3
Fully Constructed IETMs, Acquisition of	1.3.2	2
General Requirements	3.1	5
Generic Layer	3.2	6
Generic Layer Tag Set Descriptions		C-1
Generic Layer Document Type Definition		A-1
Government Documents	2.1	4
Government Documents, Drawings and Publications	2.1.2	4
Graphics	3.2.3.3	8
Handbooks	2.1.1	4
Hardness Critical Item (HCI)	3.3.3.5.2	11
Hierarchy, Item/System	3.3.3.1	10
IETMDB, Acquisition of	1.3.2	2
IETMDB Capabilities	1.3.1.1	2
If Node Template	3.2.1.4	7
Inspection	4.1	11
Integration Support	1.4.2	4
Intended Use	6.1	12
Interactive Electronic Technical Manual (IETM)	6.4.2	14
Interchange, Data Base Requirements	3.1.1	6
Introduction	1.3	2
Item-Id Attribute	3.2.1.1.2	7
Item/System Hierarchy	3.3.3.1	10
Key Word List	6.5	14
Limited Applicability, Paragraphs	1.2	1
Links to Reduce Redundancy	3.2.2.1	8
Location Elements	3.2.2.2	8
Logistics Support and Task Analysis Link	3.2.2.3	8
Loop Node Template	3.2.1.5	7

INDEX

	PARAGRAPH	PAGE
Maintainability, data	3.1.3	6
Maintainer or Operator, Parts Information	3.3.3.5.1	11
Maintenance, Organizational Level	3.3.3	10
Mean Time Between Failures (MTBF)	3.3.3.5.1	11
Menu	3.2.3.5	9
Naming, data element	3.1.2	6
National Stock Number (NSN)	3.3.3.5.2	11
Nature and Purpose of a Revisable Source Data Base	1.3.1	2
Node Alternatives Template	3.2.1.2	7
Node Sequence Template	3.2.1.3	7
Node Template	3.2.1.1	7
CDM Attribute	3.2.1.1.4	7
Item-Id Attribute	3.2.1.1.2	7
Name Attribute	3.2.1.1.1	7
Ref Attribute	3.2.1.1.5	7
Type Attribute	3.2.1.1.3	7
Non-Government Publications	2.2	5
Notes	6.	12
Operator or Maintainer Parts Information	3.3.3.5.1	11
Options, Technical Information Procurement	1.3.2	2
Order of Precedence	2.3	5
Organization Level Maintenance	3.3.3	10
Other Government Documents, Drawings and Publications	2.1.2	4
Outcome Elements	3.3.3.4.4	11
Packaging	5.	12
Paragraphs With Limited Applicability	1.2	1
Parts Information	3.3.3.5	11
Maintainer/Operator Information	3.3.3.5.1	11
Supply Information	3.3.3.5.2	11
Parts Information Provided for Parts Supply	3.3.3.5.2	11
Postconditions	3.2.4.2	9
Precedence, Order of	2.3	5
Preconditions	3.2.4.1	9
Preparation for Delivery	5.1	12
Primitive Elements	3.2.3	8
Audio	3.2.3.4	8
Graphics	3.2.3.3	8
Process	3.2.3.4	8
Tables	3.2.3.2	8
Textual Information	3.2.3.1	8
Video	3.2.3.4	8

INDEX

	PARAGRAPH	PAGE
Procedural Information	3.3.3.3	10
Process, Audio, Video	3.2.3.4	8
Procurement Options, Technical Information	1.3.2	2
Quality Assurance	6.4.3	14
Quality Assurance Provisions	4.	11
Rectification Elements	3.3.3.4.5	11
Reduce Redundancy	3.2.2.1	8
Ref Attribute	3.2.1.1.5	7
Relational Links	3.2.2	7
Requirements	3.	5
Additional Content Specific DTDs	3.1.4	6
Data Base Interchange Requirements	3.1.1	6
Database Structuring and Data Element Naming Requirements	3.1.2	6
Data Maintainability	3.1.3	6
General Requirements	3.1	5
Responsibility for Compliance	4.1.1	11
Responsibility for Inspection	4.1	11
Revisable Source Data Base	1.3.1	2
Scope	1.1	1
Selection	3.2.3.5	8
Source Maintenance and Recoverability (SMR) Code	3.3.3.5.2	11
Specifications, Standards & Handbooks	2.1.1	4
Structuring, Data Base	3.1.2	6
Subject Term (Key Word) List	6.5	14
Supply Process	3.3.3.5.2	11
System/Item Hierarchy	3.3.3.1	10
Tables	3.2.3.2	8
Tag Set Descriptions, A Content Specific Layer		D-1
Tag Set Descriptions, Generic Layer		C-1
Task Analysis Link	3.2.2.3	8
Technical Information, Format Free	1.4	3
Technical Information Procurement Options	1.3.2	2
Templates	3.2.1	6
If Node Template	3.2.1.4	7
Loop Node Template	3.2.1.5	7
Node Alternatives Template	3.2.1.2	7
Node Sequence Template	3.2.1.3	7
Node Template	3.2.1.1	7
Terms and Acronyms, Definitions of	6.4	13

INDEX

	PARAGRAPH	PAGE
Test Element	3.3.3.4.3	11
Textual Information	3.2.3.1	8
Troubleshooting Information	3.3.3.4	10
Type Attribute	3.2.1.1.3	7
Usable-on code	3.3.3.5.1	11
Use of Generic DTD Primitive Elements	3.3.2.1	9
Use of Generic DTD Template elements	3.3.2.2	9
Video	3.2.3.4	8

STANDARDIZATION DOCUMENT IMPROVEMENT PROPOSAL

INSTRUCTIONS

1. The preparing activity must complete blocks 1, 2, 3, and 8. In block 1, both the document number and revision letter should be given.
2. The submitter of this form must complete blocks 4, 5, 6, and 7.
3. The preparing activity must provide a reply within 30 days from receipt of the form.

NOTE: This form may not be used to request copies of documents, nor to request waivers, or clarification of requirements on current contracts. Comments submitted on this form do not constitute or imply authorization to waive any portion of the referenced document(s) or to amend contractual requirements.

I RECOMMEND A CHANGE:	1. DOCUMENT NUMBER	2. DOCUMENT DATE (YYMMDD)
	MIL-D-87269	20 November 1992

3. DOCUMENT TITLE

Data Base, Revisable: Interactive Electronic Technical Manuals

4. NATURE OF CHANGE *(Identify paragraph number and include proposed rewrite, if possible. Attach extra sheets as needed.)*

5. REASON FOR RECOMMENDATION

6. SUBMITTER

a. NAME (Last, First, Middle Initial)	b. ORGANIZATION	
c. ADDRESS (Include Zip Code)	d. TELEPHONE (Include Area Code) (1) Commercial (2) AUTOVON (If applicable)	7. DATE SUBMITTED (YYMMDD)

8. PREPARING ACTIVITY

a. NAME	b. TELEPHONE (Include Area Code) (1) Commercial	(2) AUTOVON
HQ AFMC/ENCS (16)	513-257-3085	787-3085
c. ADDRESS (Include Zip Code) WRIGHT-PATTERSON AFB, OH 45433-5001	IF YOU DO NOT RECEIVE A REPLY WITHIN 45 DAYS, CONTACT: Defense Quality and Standardization Office 5203 Leesburg Pike, Suite 1403, Falls Church, VA 22041-3466 Telephone (703) 756-2340 AUTOVON 289-2340	