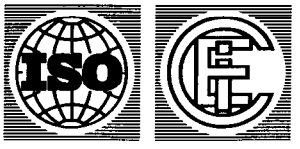
TECHNICAL REPORT

ISO/IEC TR 10037

First edition
1991-03-15

Information technology — SGML and Text-entry Systems — Guidelines for SGML Syntax-Directed Editing Systems

Traitement de l'information — Systèmes SGML — Recommendations pour les systèmes d'édition sensibles à la syntaxe SGML.



Reference number
ISO/IEC TR 10037:1991(E)

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

Contents

Page

1 Scope 1

2 Reference 2

3 Definitions 2

4 Document processing 2

5 Document Type Definition processing 9

Alphabetical index 10

© ISO/IEC 1991
All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

ISO/IEC Copyright Office • Case Postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The main task of technical committees is to prepare International Standards, but in exceptional circumstances a technical committee may propose the publication of a Technical Report of one of the following types:

— type 1, when the required support cannot be obtained for the publication of an International Standard, despite repeated efforts;

— type 2, when the subject is still under technical development or where for any other reason there is the future but not immediate possibility of an agreement on an International Standard;

— type 3, when a technical committee has collected data of a different kind from that which is normally published as an International Standard ("state of the art", for example).

Technical Reports of types 1 and 2 are subject to review within three years of publication, to decide whether they can be transformed into International Standards. Technical Reports of type 3 do not necessarily have to be reviewed until the data they provide are considered to be no longer valid or useful.

ISO/IEC/TR 10037, which is a Technical Report of type 3, was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology.*

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Introduction

This Technical Report specifies a series of guidelines for the capabilities of an SGML (Standard Generalized Markup Language, as defined in ISO 8879) Syntax-Directed Editing System. An *Editing System* is a combination of computer software and hardware that is used as a tool for the entry and modification of data. A *Syntax-Directed* Editing System is one that is aware of the syntax and structure of the data being edited, and can make use of that knowledge to ease the task of the person using that system. This Technical Report is therefore concerned with editing systems that are specifically tailored for the manipulation of documents marked up with SGML.

When electronic publishing first evolved, an author would create a document in either manuscript or typescript form, using various spacing conventions to indicate the structure of the document. An editor would then "blue pencil" it to mark the actions needed to lay out and format the document. The document would then be key-punched with the appropriate detailed formatting instructions inserted into the text.

With the advent of generic markup and availability of on-line editing systems, authors were given a list of tags and were expected to use these tags to mark up the document as it was being created, thus reducing the need for manual "blue pencilling". In some cases, having authors mark up the documents created more problems than it solved. For example:

— tags were used incorrectly

— generic identifiers (tag names) were misspelled

— tags were omitted.

With the development of SGML there is now a mechanism that allows markup errors to be found prior to composition — thus eliminating a cause of many subsequent composition errors. The drawback to this was that SGML requires that the author understand enough about the tag set to allow it to be used correctly. Many authors were unhappy about this and demanded assistance with the tagging.

Furthermore, SGML requires the creation and use of a Document Type Definition (DTD) which describes a document's structure. Creating DTDs is a difficult task that requires skill and is prone to errors; when an SGML document fails validation it is sometimes not clear whether the error is in the document or in the DTD.

It is apparent that tools are needed both to aid the document analyst in creating correct DTDs and the author in correctly using them. These tools are called *SGML Syntax-Directed Editing Systems.*

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Information technology — SGML and Text-entry Systems — Guidelines for SGML Syntax-Directed Editing Systems

1 Scope

1.1 What is in this report?

This Technical Report describes a set of functions which an SGML syntax-directed editing system may have in order to help users manipulate documents marked up according to the rules of SGML. These functions may be embodied in a special-purpose editing system, or they could be added to those functions already present in an existing text-entry or editing system. In either case the result would be a syntax-directed editing system that is optimized for the manipulation of SGML documents.

The Technical Report contains two major clauses. The first and larger clause describes the functions of an SGML syntax-directed editing system as applied to document processing (that is, the creation, viewing, and modification of an SGML source document instance). The second clause identifies those additional facilities felt to be appropriate for DTD processing. The editing of SGML declarations is not covered.

This Technical Report does not specify a "standard editing system". Any references to specific functions relating to general editing (such as insertion, deletion, or the changing of data) are made only for clarity. They do not imply any particular procedure for performing such functions, but simply indicate that their existence is presumed. The methodology for the implementation of editing functions is left to the ingenuity of the implementor.

Similarly, this Technical Report is primarily aimed at the creator of "text" documents. Although SGML can be used for describing a wide variety of data (including databases, spreadsheets, mathematics, and even music), these uses are not directly considered in this Technical Report.

1.2 Who should read this report?

This Technical Report is intended to be useful to two categories of readers:

a) People who are specifying or selecting an editing system for use with SGML documents.

b) People who are implementing an SGML syntax-directed editing system.

As such it describes a variety of facilities — some essential and some more esoteric — which may be found useful in an SGML syntax-directed editing system. By comparing the facilities available in an SGML editing system with those described here, the reader can gauge its capabilities. It is assumed that readers are familiar with the concepts and terminology of SGML; the definitions of many of the terms used in the Technical Report may be found in ISO 8879.

Most of the requirements for an SGML syntax-directed editing system are sufficiently generic that they can be implemented using a wide variety of display and entry devices. These include both text and graphics displays (whether monochrome or colour), keyboards with or without special "function" keys, mice and other pointing devices, and potentially even voice recognition and sound generation machinery. It is expected that an SGML syntax-directed editing system will take full advantage of the facilities of the devices available in a way that is natural to the users of the system.

Although the editing system may be primarily designed for the manipulation of SGML documents and DTDs, nothing in this Technical Report necessarily prevents its use with non-SGML documents (or, indeed, other data such as computer programs). However, it must be noted that the special SGML functions described in this Technical Report may not be appropriate for documents that do not use SGML syntax.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

2 Reference

The following standard contains provisions which, through reference in this text, constitute provisions of this Technical Report. At the time of publication, the edition indicated was valid. All standards are subject to revision, and parties to agreements based on this Technical Report are encouraged to investigate the possibility of applying the most recent edition of the standard indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO 8879:1986, *Information processing — Text and office systems — Standard Generalized Markup Language (SGML).*

3 Definitions

For the purposes of this Technical Report, the following definitions apply.

3.1 cueing: The giving of clues, by use of formatting, colour emphasis, etc., to the user to help identify the structure or other attributes of the data being edited.

3.2 currently open element: The open element whose *start-tag* occurred most recently (or was omitted through markup minimization).

3.3 current position: The place within the document where a subsequent insertion, deletion, or other modification will be made.

3.4 cursor: A visual indication on the display screen of the current position in data being edited.

3.5 dynamic validation: The parsing of a document or partial document for errors as part of (or very soon after) the creation or modification process (that is, takes place as the user enters or modifies the document). See also *partial validation*, *static validation.*

3.6 editing system: A combination of computer software and hardware that is used as a tool for the entry and modification of data.

3.7 free-standing element: An element that is created without reference to a containing element.

3.8 implicit unit: A *unit* that is inferred from some or all of the content of a document element, rather than from SGML rules or markup. For example, a word or sentence.

3.9 intrusive function: An SGML function or markup that intrudes upon the writing process.

3.10 partial validation: Incomplete *dynamic validation* or *static validation.*

3.11 selection: The process of selecting an item or action from a list of possibilities, or the choosing or marking of some part of a document.

3.12 selection device: The tool used to select items presented to the user by the system. For example, a mouse or function key.

3.13 static validation: The parsing of a document or partial document for errors at the end of the creation or modification process, or on specific user request. See also *dynamic validation*, *partial validation.*

3.14 structural cue: A clue, given by use of formatting, colour emphasis, etc., that helps identify the structure of the data being edited.

3.15 syntax-directed editing system: An editing system that is aware of the syntax and structure of the data being edited, and can make use of that knowledge to ease the task of the person using that system.

3.16 unit: A defined portion of the document that the SGML syntax-directed editing system user can request to be processed as an aggregate. For example, an element that is to be moved to some other position within the document would be considered a unit if it includes the start-tag, end-tag, and the element's character content.

3.17 validation services: The facilities provided by the SGML syntax-directed editing system for the validation of SGML documents or Document Type Definitions.

4 Document processing

An SGML encoded document consists of the following parts: an optional SGML declaration, a mandatory base document type definition, and the document instance. Of these, a document author would normally only be concerned with the document instance, and would use a DTD provided by a document designer. Therefore, an SGML syntax-directed editing system is primarily intended to allow the creation and modification of the document instance and its contained elements as it is the editing of document instances that is the most common task in the preparation of SGML encoded documents.

In this part of the Technical Report are described the facilities of an SGML syntax-directed editing system, as applied to the editing of document instances, in the following order:

a) the document *units* which should be "understood" by the editing system,

b) the presentation of the document,

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

c) document entry — the creation of new text and markup,

d) the manipulation and modification of an existing document,

e) desirable validation services (checks) that might be provided.

Some users of an SGML syntax-directed editing system need the ability to perform the same functions of creation and modification upon the DTD and to a lesser extent upon the SGML declaration. Those items which are specific to DTD processing are described in the next clause.

4.1 Document units

A document type definition (DTD) defines a variety of units by which it is natural to manipulate the document. The most obvious unit is a document element — the unit could be an element's start- and end-tag together with the content they enclose. Similarly, an entity reference would normally be treated as a unit. In some cases, it will be useful to treat the content of an element as a unit. Units may also consist solely of markup — it is possible that a start-tag, complete with the start-tag open, generic identifier, attribute specification list, and start-tag close could be manipulated as a single unit. An attribute may also need to be treated as a unit.

In addition to units described by the SGML syntax there may be appropriate *implicit units* that are not explicitly identified by markup, such as sentences, words, or numbers.

Units and their treatment are similar to blocks in a traditional editing system's block marking operations, where a block of characters is delineated for some manipulation; they differ in that the units described by markup are defined in the DTD and hence are subject to validation. Units may also affect the way in which the document is presented to the user.

Units supplement rather than replace the traditional editing operations that rely upon a mark being set. For example, the system would still need to be capable of the deletion and insertion of arbitrary characters in the text irrespective of any markup that might be included.

4.1.1 Elements as units

Elements form the most natural units because their structure is precisely defined by the rules of SGML and by the DTD. They can be identified by their start- and end-tags even if their presence has been inferred using SGML's rules for omitted tag or DATATAG minimization. The use of minimization will still allow an element's content to be manipulated as a single unit, even if neither the start-tag nor the end-tag is present in the source document and it may not be (immediately) apparent to the user where the start- and end-tags occur.

The identification of an element as a unit can be confusing in some circumstances when minimization is used; this confusion can be alleviated by the use of structural cues when the document is presented.

Another cause of confusion is the use of short references and maps. The occurrence of a particular sequence of characters could have different maps associated with it in different elements where the elements are presented to the user in a similar or identical style — a selection action in one place may therefore have a very different result from another, apparently similar, action. This kind of confusion is best avoided by careful DTD design.

4.1.2 Entity references

Entity references are a natural form of unit; depending on the kind of reference and the particular application these might be treated in different ways.

For example, it *may* be appropriate to let the user dynamically generate entity declarations for inclusion within the source document — these might just be for simple CDATA entities which act as straightforward text substitutions or shorthand, or they might be capable of defining more complicated entities, such as tables defined as sub-documents or text entered using special coding notations.

Allowing text substitution as shorthand during editing is a service that an editing system may provide. It is not always appropriate to use the SGML SHORTREF feature to obtain similar results.

4.1.2.1 External entities

The ability of the SGML syntax-directed editing system to treat the content of external entities as units will depend upon the workstation support for displaying and handling CDATA, SDATA, NDATA, and SUBDOC entities, and entities with associated NOTATION declarations.

In many cases, such as when the external entity is not available at the time of editing (or when the content cannot be displayed), the SGML syntax-directed editing system will only be able to give an indication of the reference.

4.1.3 Units related to other markup

Four other forms of markup lead to natural units. These are processing instructions, marked sections, suppression of markup recognition, and comment declarations.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

4.1.3.1 Processing instructions

A processing instruction (PI) would normally be treated as a single unit; such treatment is particularly convenient when removing all or selected PIs from a document.

4.1.3.2 Marked sections

It is useful to treat marked sections as units; some SGML syntax-directed editing systems may support sophisticated "versioning" mechanisms using marked sections. The effective status of a marked section may suppress the recognition of some types of units.

4.1.3.3 Markup recognition suppression

If markup recognition suppression has been enabled in the particular concrete syntax by assigning characters to the appropriate classes (MSICHAR, MSOCHAR, and MSSCHAR) and characters in these classes occur in the document their occurrences will modify the boundaries of units and may cause the recognition of recursive units to be ignored.

4.1.3.4 Comment declarations

Comments, being a markup declaration, would normally be treated as units.

4.2 Presentation of documents

Before a user can interact with an SGML document it must be displayed or presented in some form. An SGML syntax-directed editing system can make good use of its knowledge of the structure of the document (derived from the DTD) and of the document units described above to ease the user's task. For example, the editing system may permit the selection and viewing of sections of the document, selected from a contents list. It could show parts of the document in the context of the surrounding structure. It might permit the editing of partial documents or document fragments. Elements whose function is to annotate the source document (but which would not normally appear in the final formatted form) might be displayed or not displayed on demand, and given colour emphasis to differentiate them from other content.

The SGML structure can therefore considerably influence the presentation of the document to the user. This influence is referred to as an intrusion when it disturbs the normal writing process, and as *cueing* (as in giving clues) when the influence assists the normal writing process. Independent of this, SGML aspects (such as markup) can be more or less obvious to the user, and the degree to which the markup is visible to the user may well be varied to suit the user or the application.

For example, in highly structured applications (such as the editing of dictionary entries) any special formatting could be primarily for the purpose of emphasizing the various parts of a document's structure; the user is free to enter any text and markup at any point in the document (with validation of the document being either immediate or delayed, as appropriate). As an alternative, an editing system could use a representation of the document and its structure which is not that which is defined in ISO 8879. Formatting, emphasis, and structural cues can be used to convey the structure of the document to the user. The user's perceived model of the editing process need not necessarily be that of an SGML document, even though the editing system will ensure that the final result is indeed a valid document as described in a specific DTD.

A particular SGML syntax-directed editing system may be able to vary the presentation of the document in various ways; in general it is an advantage if the presentation mechanisms in an editing system are customizable to suit the application and the personal preferences of the user.

The general patterns of cueing are described below.

4.2.1 Overt tag cueing

The most simple form of presentation allows some or all of the tags (and other markup) in a document to be explicit (that is, visible to the user). In this form the text and its markup on the screen is neither translated to an alternative representation (such as graphical or other devices), nor is it hidden in any manner.

If, as presented, the document appears as a valid SGML document (that is, with all necessary tags and markup displayed) the document is referred to as being in the unformatted state.

4.2.2 Covert structure cueing

SGML syntax-directed editing systems can have a formatted state where the text and its markup are presented on a display screen with the SGML descriptive structure more or less transformed to something deemed to be more understandable by the user. Screen representation has at least three forms:

a) **Positional Transformation**

The structural relationships between SGML elements can be represented on-screen by their relative vertical or horizontal position with respect to each other. Typical positional constructs are column positioning, left and right indention, and vertical spacing between elements.

b) **Graphical Transformation**

Visual cues can be used to signal to the user some characteristic of the text that is being displayed. This may be done by attaching some extra signal feature to (or near to) the character or characters,

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

thus distinguishing them from their simplest representation. These graphical cues may or may not be related to the representation on some final output medium. Typical graphical cues include font changes, colour emphasis, the use of icons, and various character enhancement techniques.

Similarly, entity references may be identified or enhanced by appropriate graphical cues.

c) **Semantic Translation**

Semantic translation is the process by which the meaning of an SGML element is translated textually into a more acceptable natural language form for the user. Typically a phrase made up of ordinary words is used, the meaning of which helps the user appreciate the structure of the document.

This form of cueing is especially useful for EMPTY elements, though it is also useful whenever an element would normally "generate" text when a document is formatted.

4.2.3 Specification of structural cues

Some additional information to that defined for the DTD is required to describe the display formatting and representation of each element. The syntax and location of this information is not within the scope of this Technical Report; however, certain basic semantics such as display techniques, flowed or unflowed formatting, and so on, will be supported by any SGML syntax-directed editing system which supports covert structure cueing.

The scope and accuracy of on-screen formatting will be limited by the speed and sophistication of the supporting hardware, and by the application-specific requirements for the system. While nothing in this Technical Report prohibits the full formatting of the document to the point where it mimics some final output form, the primary reason for on-screen formatting in an SGML-based text entry system is input assistance and verification. The author is given rapid feedback and added assurance that the markup as entered is correct; review of the document is enhanced and the probability of the incorrect entry of data is reduced.

Input **is** facilitated by rapid responses while it is **not** necessarily facilitated by certain formatting techniques such as line justification (especially if the formatting comes at the expense of performance), nor does input facilitation mean that an attempt should always be made to emulate, however slightly, composition output.

NOTE 1 For example, the start-tag <**company**> might cause its element content to be composed *emphasized* and with a certain point size, yet during data entry it might be more appropriate to map the beginning of the element to the prompt "please enter company name".

4.3 Document entry — creating new text and markup

Creating a new document (or adding to an existing document) involves both writing the textual content of the work and indicating its structure. The goal of an SGML syntax-directed editing system is to make the second of these as easy as possible without intruding on the first.

There are a variety of ways in which the structure can be communicated to the editing system: an input methodology based on that defined in ISO 8879 (using a text representation and the various minimization and short reference techniques) can be used, or alternatively a new methodology (for example, the use of pictorial representations) might be appropriate, depending on the user and on the application. In many cases it will be appropriate to allow the user a choice of methods. Some possible methods are:

— Character-oriented direct tag input. The tags are typed in manually, either in the final SGML form or in some alternative representation.

— Single-stroke tag input. Commonly-used tags are entered by a single keystroke, perhaps mnemonically related to alphabetic keys or perhaps by use of special function keys.

— Selection from a presented list of tags.

— Selection from a presented list of semantically translated structural cues (for example, "Start of Chapter" rather than <**h1**>).

There should normally be support within the editing system to guard against the creation of undesirably recursive units. For example, the entry of a marked section declaration or an MSC in an ignore marked section would often be undesirable.

4.3.1 Selection mechanism

When markup or some other document unit is to be selected, a number of possibilities and considerations apply:

— The editing system might map possible markup selections on to a set of keys, following a set of rules for choosing the markup most likely to be required.

— The editing system might display a list of some or all of the legal tags (including attributes, if appropriate) at any state in the document and provide the user with a convenient way to select them.

— A key might insert both the start- and end-tags for non-minimized elements whose content is PCDATA or mixed, and position the cursor appropriately.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

— It may be possible to present the mapped character sequence or sequences for an element defined as a DATATAG as selections.

— The editing system might either omit some or all of the tags, as permitted by the DTD, or it might provide for the inclusion of all start- and end-tags. This latter possibility can be useful for the export of documents to SGML processing systems that do not support the OMITTAG feature.

— The selection of many kinds of document units will probably be possible; these should be provided for in a manner consistent with that used for selecting element markup.

4.4 Manipulation of SGML structures

4.4.1 Cursor movement

Movement about the document can either be between character positions or between units. Alterations to the current position in a document (usually indicated by some form of cursor) may be achieved by moving to the next (or previous) character or unit. As with other text editing systems, it is usual that an SGML syntax-directed editing system will allow movement by implicit units (such as words, numbers, and sentences) as well.

Arbitrary movements within the document would also normally be allowed (for example, to some previously marked point, or while scrolling rapidly through a document).

4.4.1.1 Ambiguous cursor positioning

Certain representations of cursor position can be ambiguous as to where a specific action (such as insertion or deletion) will take place. As is appropriate for all editing systems, an SGML syntax-directed editing system should use a cursor representation that is always self-consistent and which clearly shows the point at which actions are to take place.

When some or all of the markup in a document is hidden there may be ambiguous presentations in which the actual current position cannot be deduced from the displayed position of the cursor. For example, if the underlying document content included the sequence

```
abc<q><e>def</e>ghi</q>jkl
```

then if the markup were hidden from view the display might (if the <**e**> element signified emphasis, represented by the use of underscored type) show

abc<u>def</u>ghijkl

and in this case a cursor positioned between the "c" and the "d" might be ambiguous — is it next to one of those characters, or between the two tags?

When this kind of ambiguity occurs, some alternate method for indicating the underlying structure may need to be employed. For example, the elements could be displayed directly, or assigned an alternative cue that represents the structure; this could be used when the cursor is in the immediate vicinity of the start- or end-tag. Other techniques for markup representation can help, too: if the <**q**> element in the example indicated a quotation then that element's start and finish could be appropriately indicated with quotation marks (which are not in any sense part of the document content).

4.4.2 Manipulation of document units

The two fundamental modification functions of editing systems are deletion and insertion. When applied in an SGML syntax-directed editing system there are two categories of modification functions:

4.4.2.1 Structural modification

Here the object of insertion and deletion activities is structural, such as a particular group of one or more contiguous elements (and their content) or some other unit. It is useful if control over which units may be modified is available — for example, when the goal of the editing is translation (from one language to another) it may be required that the overall structure of a document be protected against change.

4.4.2.2 Character modification

Here the basic object of insertion and deletion activities is the character (or groups of characters), irrespective of element boundaries.

This action may create a non-conforming document in the case where inserting or deleting text across element boundaries results in misplaced or missing tags, or invalid (perhaps recursive) document units. Depending on the application and the validation (checking) in effect, some indication may be made of error conditions like these.

4.4.3 Entity processing

4.4.3.1 General requirements

When entities are used, the SGML syntax-directed editing system should, if possible, verify that the entity has been declared. External entity declarations may not be available at the time of editing and so cannot always be verified, though it may be possible to check the syntax of PUBLIC entity declarations. If external entity references have not been validated because the declaration does not yet exist, the user should be warned that the SGML document instance

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

may be invalid at the end of the editing session, if the editing system supports validation.

The presentation and selection of entities will vary according to the application, but should be consistent with the treatment of other document units (for example, it may be possible to generate an entity reference by single keystrokes, or by selection from lists). However, since entity references generally refer to or include part of the content of the document they will not usually be hidden even when other markup is hidden.

4.4.3.2 Internal entities

The SGML syntax-directed editing system should be able to display an internal entity reference with the declared value of the entity, as well as optionally providing access to the entity reference itself.

It may also be useful to be able to request a true replace of the reference by its value, but it should be noted that entity references are often used in situations where the value of the entity would create an invalid or modified document. This is especially true for CDATA or SDATA entities. For example, it is reasonable to have an SDATA entity whose value is a valid tag. As an entity reference it would have no effect on the structure of the document. If it were replaced in the document instance with its expansion it could radically affect the structure or validity of the document.

4.4.3.3 External entities

The system should provide the ability to display external data entities, provided that the entity can be found and is in a form that can be displayed by the editing system.

Again, it may also be useful to be able to request a true replace of the external entity reference by its value, with the same caveat as for internal entity reference replacement.

The user could optionally be warned if, during validation, an external entity cannot be found (though this is not necessarily an error).

NOTES

2 The SGML syntax-directed editing system may be able to interpret the system parameter literal, public identifier, and entity name in an external entity declaration as a file or object name in the editing system's permanent storage subsystem. If so, it may be helpful to the user to provide this information.

3 An external entity (assuming it is available) may need to be parsed in accordance with the rules of SGML.

Different processing is required for references in content to SGML and non-SGML entities. The content of SGML entities affects the validity of a document that refers to them; their content will need to be parsed by a validating editing system. A document that refers to unavailable SGML entities cannot be fully validated. Note that an editing system has to be able to interpret the system identifier, public identifier, and entity name of an SGML external entity in order to be able to read and parse it at the point of reference.

The content of non-SGML entities does not affect the validity of a document that refers to them. They may, in fact, be prepared independently of the SGML document, and it is quite possible that the editing system may not be able to interpret a non-SGML entity's system identifier, public identifier, and name. A warning may or may not be appropriate, depending on the circumstances of the particular system.

4.4.4 Other considerations

4.4.4.1 Subdocuments

A subdocument is validated independently of any document that refers to it. Whether a subdocument is, or can optionally be, validated at the time a reference to it is encountered or at some other time (it may already be known to be valid, for example) is a decision that can be made as part of the design of a specific editor.

4.4.4.2 Notations

If the SGML syntax-directed editing system does not support a particular notation or notations specified for an element or entity, it should permit the content to be considered for treatment as ordinary text for display (and modification, if allowed).

It should be noted that elements that have specified notation attributes are necessarily editable, because their content is part of the document instance. References to external entities that have notations, on the other hand, are subject to the same criteria discussed earlier.

4.4.4.3 Concurrent documents

The SGML syntax-directed editing system, if supporting concurrent documents, should understand the rules for concurrent documents as defined in ISO 8879.

4.4.4.4 Application-dependent help

In addition to the aids normally found in an editing system, the SGML syntax-directed editing system could include a help facility which enables the logical structure of the elements to be described. The help facility could also permit the user to examine an appropriate display of the current structure of the document. This could be a simple "table of contents" of a document, or it might provide access to the details of the SGML markup.

For example, the content model of the current element could be displayed in a suitable manner. An

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

overview of the DTD or the currently applicable declarations (both element and attribute list declarations) could also be displayed, using the current element's declaration as a starting point. The user may be able to scroll through this overview examining higher or lower declarations or to select a token with a content model and examine the corresponding declarations.

4.5 Validation services

An SGML syntax-directed editing system is a tool for producing a document which is correctly marked up according to a specified DTD. It is the role of the *validation services* to aid in the prevention, detection, and correction of markup errors as the document is being created and modified.

Depending on the editing system and the application, validation may be *dynamic* (that is, takes place as the user enters or modifies the document) or it may be *static* (validation is carried out on demand). In general dynamic validation is preferred, though it may be that only partial (that is, incomplete) dynamic validation is possible within the constraints of the resources available. For example, it may only be possible to validate the document from the top down to the current position.

Markup errors have many forms but are mainly caused by missing, misplaced, or misspelled tags, attributes, and entities. Many occurrences of such errors can be prevented, in various ways. For example, misspelling of tag and entity names can be avoided by allowing them to be inserted from a presented list. In addition, such lists can be restricted to those elements (and, sometimes, entities) that will maintain the validity of the document.

However, when generating a restricted list, the SGML syntax-directed editing system may need to use a more flexible definition of "valid" from that used by a validating SGML parser because a document being edited is rarely in a truly valid state. For example, consider an element whose content model contains a required sub-element. Inserting such an element immediately creates an invalid document since the required sub-element is missing. Also, a document is normally parsed top-down but many are created simply from beginning to end and may not be valid until almost completed.

For the validation services to be useful, then, they must continue to function even though the document is invalid. That is, they must continue to aid in the prevention of further errors in the presence of existing errors.

When errors are detected it is important that only actual, as opposed to propagated, errors be reported. For example, a misplaced element will cause the following content of the enclosing element to be invalid, yet this should be reported as only a single error.

The simplest way to avoid reporting propagated errors is to report only the first error encountered; this forces the user to correct the document in a top-down manner. More sophisticated algorithms may be applied to allow more than one real error to be correctly identified. In either case, as long as there remains at least one error in the document there may be other errors not yet revealed. The SGML syntax-directed editing system should normally indicate when the document is completely free of errors (if there are external entity references this may be conditional on their content being available).

4.5.1 Entity validation

Part of the purpose of a DTD is to control the type of markup applied to a document. If a user is to be allowed to dynamically create new entities (for shorthand or any other purposes) then some restrictions will apply.

a) The entity must be declared within a document subset of a DOCTYPE declaration, or at an appropriate position in a complete DTD.

b) The editing system should ensure that the SGML document is valid when it is written out by, at a minimum, ensuring that a correct prologue complete with the new entity declarations is written with the document instance. This is necessary because the document instance may not be valid with respect to any pre-existing DTD.

c) At the time of the dynamic declaration of the entity, the DTD will need to be revalidated in order to ensure that it has not just become invalid.

NOTE 4 Revalidation of the DTD may be as simple as checking that the entity has not been declared in the DTD already and that there has been no reference to the new entity in the document instance (which may have been resolved by a default general entity).

4.5.2 Recognizable errors

There are a variety of common errors which an SGML syntax-directed editing system should be able to detect and report. Any violation of the SGML rules, as defined for GENERAL validation in ISO 8879, should normally be indicated, but the following errors may be sufficiently common — especially if direct entry of markup is allowed — that special presentation and treatment is justified:

— **Element is Unknown**

A tag was entered manually that does not identify an element in the DTD. It may be possible to suggest a list of probable elements, using a combination of knowledge of the structure of the document and spelling-checker techniques.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

— **Element is Out of Place**

An element has been found or inserted in a position not allowed by the DTD. Again, it may be possible to suggest some appropriate corrective action.

NOTE 5 The physical position of the tag may be correct; the error may be in a preceding element.

— **Element is not Closed**

The document is incomplete, or a required end-tag has been omitted. The identifiers of unclosed elements, and the position of their start-tags, can be shown.

— **Missing Required Attribute**

A required attribute is missing from a start-tag. The name of the required attribute (and possible values, if known) can be shown.

— **Unknown Attribute**

An attribute that has been specified is not defined in the DTD. The unknown attribute can be indicated.

— **Bad Attribute Value**

The value of an attribute does not match any of the possibilities defined in the DTD. The name of the attribute, the bad value, and valid values can be shown.

4.5.3 Free-standing validation

An SGML syntax-directed editing system might provide a means to validate free-standing elements (document fragments) in accordance with the appropriate content model isolated from the remainder of the DTD.

NOTE 6 An element validated in such a manner may prove to be contextually invalid when included within a surrounding document structure.

5 Document Type Definition processing

As described above, the majority of users of an SGML syntax-directed editing system will be using it for the creation and modification of documents. A small proportion, however, will wish to use it for editing DTDs — the designer of a DTD needs as much assistance as an author to ensure that his or her work conforms to the specification of SGML as described in ISO 8879.

Some of the facilities already described will apply to the editing of DTDs as well as to the editing of documents; this part of the Technical Report details those additional facilities that are specific to the treatment of DTDs by an SGML syntax-directed editing system.

NOTE 7 The editing of SGML declarations is not within the scope of this report; users should, however, be aware that editing an SGML declaration may require the complete re-validation of all dependent DTDs and document instances.

5.1 Creation of Document Type Definition

A DTD can be created by simply treating it as textual material, similar to the way in which a document is handled, and using the same facilities and services available when editing documents. Alternatively, there are a variety of graphical techniques, such as "tree" diagrams, that might be used to aid the specification of the DTD. All SGML syntax-directed editing systems should allow the textual treatment of a DTD; more sophisticated methods may be appropriate for certain applications and uses of SGML. For example, automatic proposal of parameter entities for common content models could well be helpful to the DTD designer.

The prime requirement during DTD creation is that the DTD be validated and therefore the concepts developed in 4.5 are especially relevant to that part of an SGML syntax-directed editing system that deals with creation and manipulation of DTDs; specific DTD concerns are detailed below. As before, the validation might be wholly or partly dynamic. The concept of units as described in 4.1 is also relevant when working on a DTD.

5.2 Validation of Document Type Definition

An SGML syntax-directed editing system that supports the editing and validation of DTDs must be able to verify that the DTD conforms to the syntax specified in ISO 8879 and be able to validate GENERAL. Any SGML features supported for document editing should also be supported during DTD validation; the optional validation services that apply to DTD's are CAPACITY, EXCLUDE, MODEL, and FORMAL.

An additional condition that could be checked for is the following exceptional situation:

5.2.1 Identification of missing elements

The system could provide an optional warning if any element mentioned in a content model is not defined inside the type definition. The system should also provide a warning when it discovers elements that are defined in a type definition and not referenced in any content model (normally there will be at least one such element — the document element, which will have the same name as the DTD).

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Alphabetical index

A

Annotations 4.2
Application-dependent help 4.4.4.4
Attributes 4.1, 4.5.2
— as units 4.1
— bad values 4.5.2
— missing 4.5.2
— unknown 4.5.2

B

Background Introduction
Blocks 4.1
Blue pencilling Introduction

C

CDATA entities 4.1.2, 4.1.2.1
Character modification 4.4.2.2
Checking documents 4.5
Colour emphasis 4.2, 4.2.2
Comment declarations 4.1.3.4
Concurrent documents 4.4.4.3
Content model 4.4.4.4, 5.2.1
— display of 4.4.4.4
— in DTD validation 5.2.1
Contents lists 4.2
Correction of errors 4.5
Covert structure cueing 4.2.2
Creating new documents 4.3
Cueing 3.1, 4.2
Cues 4.2.1, 4.2.2
— graphical 4.2.2
— overt tags 4.2.1
— positional 4.2.2
— semantic 4.2.2
Current position 3.3, 4.4.1
— altering 4.4.1
Currently open element 3.2
Cursor 3.4, 4.4.1, 4.4.1.1
— movement 4.4.1
— position, ambiguous 4.4.1.1

D

DATATAG 4.1.1, 4.3.1
— entry 4.3.1
— minimization 4.1.1
Deletion 4.4.2.1, 4.4.2.2
— of characters 4.4.2.2
— of units 4.4.2.1
Detection of errors 4.5
Display of documents 4.2
Document Introduction, 4, 4.1, 4.3, 4.5.3
— entry 4.3
— fragments, validation of 4.5.3
— generic markup Introduction
— new 4.3
— processing 4
— units 4.1
Document instance 4
Document Type Definition
— See DTD
DTD Introduction, 4.1.1, 5, 5.1, 5.2
— creation of 5.1
— design can reduce confusion 4.1.1
— processing of 5
— validation of 5.1, 5.2
Dynamic validation 3.5, 4.5

E

Editing system Introduction, 3.6
Elements 4.1, 4.1.1, 4.3, 4.5.2, 5.2.1
— as units 4.1, 4.1.1
— input methods 4.3
— missing from DTD 5.2.1
— open 4.5.2
— out of place 4.5.2
— unknown 4.5.2
Entities
— See Entity references
Entity declarations 4.1.2
— generation of 4.1.2
Entity processing 4.4.3
Entity references 4.1, 4.1.2, 4.1.2.1, 4.4.3.1, 4.4.3.2, 4.4.3.3
— as units 4.1
— external 4.1.2.1, 4.4.3.3
— internal 4.4.3.2
Entity validation 4.5.1
Errors, recognized during validation 4.5.2
External entities 4.1.2.1

F

Feedback to user 4.2.3
Font changes, use of 4.2.2
Formatting for display 4.2, 4.2.3
Fragments, validation of 4.5.3
free-standing element 3.7
Free-standing elements 3.7, 4.5.3
— validation 4.5.3

G

Generic markup Introduction
Graphical transformation cues 4.2.2

H

Help for the user 4.4.4.4
Hiding markup 4.2

I

Icons, use of 4.2.2
implicit unit 3.8
Implicit units 3.8, 4.1
Indention, use of 4.2.2
Input verification 4.2.3
Insertion 4.4.2.1, 4.4.2.2
— of characters 4.4.2.2
— of units 4.4.2.1
Instance, of a document 4
Intrusive function 3.9

K

Keys, mapping of 4.3.1

L

Lists of elements and entities 4.5

M

Mark operations 4.1
Marked sections 4.1.3.2, 4.3
Markup 4.1.3.3, 4.2, 4.3, 4.5
— errors 4.5
— input methods 4.3
— recognition suppression 4.1.3.3
— visibility 4.2
Minimization 4.1.1
— DATATAG 4.1.1
— omitted tag 4.1.1
Modification 4.4.2.1, 4.4.2.2
— of characters 4.4.2.2
— of units 4.4.2.1

N

NDATA entities 4.1.2.1
New documents 4.3
NOTATION declarations 4.1.2.1
Notations 4.4.4.2
Numbers, as units 4.1

O

OMITTAG feature 4.3.1
Omitted tag minimization 4.1.1
Overt tag cueing 4.2.1

P

Partial validation 3.10
Positional transformation cues 4.2.2
Presentation of documents 4.2
Prevention of errors 4.5
Processing instructions 4.1.3.1
Prompts, use of 4.2.3
Public identifiers 4.4.3.3
— of external entities 4.4.3.3

R

Recognizable errors 4.5.2
Required attributes, missing 4.5.2
Response time, significance of 4.2.3

S

SDATA entities 4.1.2.1
Selection 3.11, 3.12, 4.3.1
— devices 3.12
— mechanisms 4.3.1
selection device 3.12
Semantic transformation cues 4.2.2
Sentences, as units 4.1
SGML 4
— declaration 4
Short references 4.1.1
— confusion caused by 4.1.1
Specification of structural cues 4.2.3
Static validation 3.13, 4.5
Structural cue 3.14

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Structural relationships 4.2.2
Structures 4.2, 4.2.3, 4.4, 4.4.4.4
— cueing 4.2
— *display of* 4.4.4.4
— manipulation of 4.4
— specification of cues 4.2.3
SUBDOC entities 4.1.2.1, 4.4.4.1
Subdocuments 4.4.4.1
Syntax-Directed editing system Introduction, 3.15
System parameter literal 4.4.3.3

T

Tags 4.1, 4.3
— *as units* 4.1
— input methods 4.3

U

Unformatted document state 4.2.1
unit 3.16
Units 3.16, 4.1, 4.4, 4.4.2
— document 4.1
— manipulation of 4.4, 4.4.2

V

Validation 3.17, 4.5, 4.5.1, 4.5.3, 5.2
— of document fragments 4.5.3
— of DTD 5.2
— of entities 4.5.1
— services 3.17, 4.5
validation services 3.17
Vertical spacing, use of 4.2.2

W

Words, as units 4.1

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

UDC 681.3.06:655.53:(083.13)

Descriptors: data processing, text processing, information interchange, documentation, documents, logical structure, programming (computers), artificial languages, programming languages.

Price based on 11 pages

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale