INTERNATIONAL STANDARD ISO 8879:1986
TECHNICAL CORRIGENDUM 2

Published 1999-11-01

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION
INTERNATIONAL ELECTROTECHNICAL COMMISSION • МЕЖДУНАРОДНАЯ ЭЛЕКТРОТЕХНИЧЕСКАЯ КОМИССИЯ • COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

Information processing — Text and office systems — Standard Generalized Markup Language (SGML)

TECHNICAL CORRIGENDUM 2

Traitement de l'information — Systèmes bureautiques — Langage normalisé de balisage généralisé (SGML)

RECTIFICATIF TECHNIQUE 2

Technical Corrigendum 2 to International Standard ISO 8879:1986 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 34, *Document description and processing languages*.

All occurrences of "element declaration" in this International Standard are changed to "element type declaration".

All occurrences of "element set" in this International Standard are changed to "element type set".

Replace 4.223 with:

4.223 owner identifier: The portion of a public identifier that identifies its owner.

Note: The owner of a public identifier is not necessarily the owner of the object that it identifies.

Add the following normative annex K and informative annex L to ISO 8879.

ICS 35.240.30 **Ref. No. ISO 8879:1986/Cor.2:1999(E)**

Annex K
(normative)
Web SGML Adaptations

This annex remedies defects revealed by the multiple adaptations of SGML for the World Wide Web (WWW), intranets, and extranets. The annex corrects errors, resolves ambiguities for which there is a clear resolution that does not cause existing conforming documents to become non-conforming, and provides a choice of alternative resolutions for other ambiguities. Although motivated by the World Wide Web, applicability of this annex extends to all uses of SGML.

This annex makes reference to groves and property sets, which are defined in the SGML Extended Facilities of the 2nd Edition of HyTime (ISO/IEC 10744) and also in DSSSL (ISO/IEC 10179).

K.1 Conformance

The decision to conform to this annex is made for individual documents and is indicated in the SGML declaration, as described below (K.3.2). An SGML system need not support this annex in order to be a conforming SGML system.

This annex is organized as a set of replacement and new syntax productions and text, and is phrased in terms of modifications to be made to the body of this International Standard, though the numbers of the affected clauses are not necessarily cited. However, these modifications are applicable only when conforming to this annex.

An SGML parser that supports this annex shall be able to parse conforming SGML documents that do not claim conformance to it. The parsing of such documents must produce the same grove as would a conforming SGML parser that does not support this annex. Error reporting for such documents, however, is with respect to the standard as modified by this annex; errors with respect to the unmodified standard need not be reported.

K.2 Definitions

K.2.1 Definitions related to document type declarations

K.2.1.1 DTD declarations:

Markup declarations that occur in a document type declaration subset.

K.2.1.2 external subset:

The portion of a document type declaration subset located within the external subset entity.

K.2.1.3 internal subset:

The portion of a document type declaration subset that occurs between the dso and dsc of a document type declaration.

If the dso and dsc are omitted, the internal subset is empty (rather than non-existent).

K.2.1.4 external subset entity:

The entity that is declared by the external identifier parameter of a document type declaration and referenced (implicitly) at the end of the internal subset.

K.2.2 Definitions related to document instances

K.2.2.1 fully-declared document instance:

A document instance whose associated document type declaration contains sufficient markup declarations to express all of the DTD properties required for the instance by the body of this International Standard, and that does not contain DTD data entities that modify those properties.

The document type declaration itself could be an implied declaration, as provided in clause K.4.9.

Note 1: An SGML declaration requires document instances to be fully-declared if it specifies IMPLYDEF ATTLIST NO and ELEMENT NO ENTITY NO NOTATION NO. A system should offer means, such as a parameter to the invocation of processing, to request validation of whether an instance is fully-declared even when the SGML declaration does not require it to be.

A document instance that is not fully-declared must be either a fully-tagged document instance or an amply-tagged document instance.

K.2.2.2 fully-tagged document instance:

A document instance in which a start-tag with a generic identifier, and an end-tag, are present for every element, and the attribute name is present in every attribute specification in the start-tag.

Note 2: An SGML declaration requires document instances to be fully-tagged if it specifies DATATAG NO, RANK NO, OMITTAG NO, SHORTTAG STARTTAG EMPTY NO, and SHORTTAG ATTRIB OMITNAME NO. A system should offer means, such as a parameter to the invocation of processing, to request validation of whether an instance is fully-tagged even when the SGML declaration does not require it to be.

K.2.2.3 type-valid document instance:

A document instance that conforms to such DTD properties as are required for it by the body of this International Standard and that are expressed in its associated document type declaration, either by markup declarations, DTD data entities, or both.

K.2.2.4 amply-tagged document instance:

A document instance whose use of markup minimization does not require access to a document type declaration.

Note 3: An SGML declaration requires document instances to be amply-tagged, insofar as they use minimization, if it specifies DATATAG NO, RANK NO, SHORTTAG ATTRIB OMITNAME NO, and either IMPLYDEF ELEMENT NO or IMPLYDEF ELEMENT ANYOTHER. A system should offer means, such as a parameter to the invocation of processing, to request validation of whether an instance is amply-tagged even when the SGML declaration does not require it to be.

Note 4:

End-tags can be omitted in an amply-tagged document instance, even for element types that have been declared by default as provided for in K.3.7.

More precisely, an amply-tagged document instance can omit the end-tag for an element of an implicitly-declared type by satisfying any of the three conditions -- a), b), or c) -- specified in 7.3.1.2. However, condition c) can be satisfied only when "IMPLYDEF ELEMENT ANYOTHER" is specified in the SGML declaration and the element is followed by another element of the same type.

The following amply-tagged document instance illustrates these possibilities when "IMPLYDEF ELEMENT ANYOTHER" has been specified and there are no DTD declarations:

```
<chapter>
<title>Chapter title</title>
<list>
<item>First item in the list
<item>Second item in the list
</list>
```

The end-tag for the first item in the list can be omitted because of condition c): This element is followed by an element that is not allowed in its content--in particular, another item.

The end-tag for the second item in the list can be omitted because of condition b): This element is followed by the end-tag of another open element (the list).

The end-tag for the chapter can be omitted because of condition a): This element is followed by the end of the SGML document entity.

K.2.2.5 immediately recursive element:

An element whose immediate content includes a child subelement of the same type.

Note 5: In the following fully-tagged text, the first section element is immediately recursive while the first list element is not.

```
<section>
<title>The title</title>
<section>
<p>The data.</p>
</section>
</section>

<list>
<item>First outer list item</item>
<item>Second outer list item, which contains an inner list.
<list>
<item>Inner list item</item>
</list>
</item>
</list>
```

Note 6: A system should offer means, such as a parameter to the invocation of processing, to request validation that an instance contains no immediately recursive elements even when the DTD would permit them.

K.2.3 Definitions related to entity constraints

Note 7: An SGML system that supports unconstrained SGML documents must be able to parse DTD declarations and resolve both internal and external entity references. If it continues parsing (as a form of error-recovery) after failing to access a referenced entity, the results are unpredictable. Observing one or more of the entity constraints defined in this International Standard may cause a document to be more amenable to processing by a simpler SGML system, or in an environment (such as a network) where access to external entities may be slow or unreliable.

K.2.3.1 integrally-stored document instance:

A document instance in which every element and marked section ends in the entity in which it begins.

Note 8: This constraint makes it possible, as a form of error-recovery, for parsing to continue in a fully-tagged document instance after a failure to access a referenced entity. The resulting grove will be the same for the parsed text, except for the tree addresses of younger siblings of the nodes in the inaccessible entity. The constraint also has implementation benefits for editors, "lazy" replacement of entity references, and sharing of grove portions when entities are reused.

K.2.3.2 reference-free document:

An SGML document that has no entity references other than references to predefined data character entities.

Note 9: A reference-free document can be parsed by conforming SGML systems that cannot resolve entity references.

K.2.3.3 external-reference-free document:

An SGML document that has no external entity references.

Note 10: An external-reference-free document could have attribute values that contain names of external entities or that otherwise might cause an application to access an external entity by means other than entity references.

Note 11: External-reference-free documents can be parsed by systems that cannot resolve external entity-references.

K.2.4 Other definitions

K.2.4.1 predefined data character entity:

A general entity, associated with a character number in the syntax-reference character set, that is used to reference significant SGML characters as data.

Note 12: In order to allow delimiter escaping when parsing without respect to DTD declarations, there should be a predefined data character entity for the first character of each delimiter string that can be recognized in a mode where data can occur.

K.2.4.2 white space:

The characters assigned to the SEPCHAR, SPACE, RE, and RS functions.

K.2.4.3 mandatorily empty element:

An element of a type declared EMPTY, or that is forced to be EMPTY by an explicit content reference attribute.

K.2.5 Definitions related to DTD notations

K.2.5.1 DTD notation:

A data notation that is capable of expressing DTD properties.

Note 13: This International Standard provides markup declarations as the means of expressing DTD properties. It also allows the use of DTD notations, but does not define any.

Note 14: A DTD notation can also be capable of expressing other information. For example, it could express constraints on a document's data content or structure that cannot be expressed by markup declarations but could be validated by an application. However, it is not a reportable markup error if the document fails to conform to such additional constraints.

Note 15: DTD notations are used in DTD data entities.

K.2.5.2 DTD properties:

Classes and properties that are specifiable by markup declarations. They are defined by the following grove plan:

```
<!DOCTYPE grovplan PUBLIC "ISO/IEC 10744:1997//DTD Grove
Plan//EN">
<grovplan propset=SGMLProp id=dtdprops>
<title>DTD Properties Grove Plan</title>
<desc>
Classes and properties that are specifiable by DTD declarations. They
are needed to parse and validate a document instance.
</desc>
<inclmod>
prlgabs0 prlgabs1 dtgabs rankabs srabs subdcabs
fpiabs arcabs fsiabs dafeabs gadcabs pelement
</inclmod>
<inclclas>
sgmldoc doctpdcl
</inclclas>
<omitprop classes="sgmldoc">
appinfo epilog
</omitprop>
</grovplan>
```

K.2.5.3 Parameter entity (4.225)

An entity that is either the external subset entity, or that is declared with a parameter entity name.

K.2.5.4 DTD data entity:

An external parameter entity whose declaration includes a notation name.

Note 16: The notation name should be that of a DTD notation and the entity content should describe DTD properties.

K.3 SGML declaration

```
[171] SGML declaration =
"<!SGML", ps+,
(SGML declaration reference | SGML declaration body),
ps*, ">"
```

In order for SGML documents to be self-identifying, it is strongly recommended that all conforming SGML documents contain one of the forms of SGML declaration.

K.3.1 SGML declaration reference

```
[171.1] SGML declaration reference =
name, external identifier?
where:
"name" is a name in the reference concrete syntax.
"external identifier" must reference an SGML declaration body. If
  omitted, the external identifier is "SYSTEM".
```

An external identifier, if it is a formal public identifier, should use "SD" as the public text class to identify an SGML declaration body, rather than one of those listed in 10.2.2.1.

Note 17: For example:

```
<!SGML HTML3.2 PUBLIC "+//IDN W3C.ORG//SD HTML Version 3.2//EN">
```

In 10.2.2.1 replace the phrase "The name must be" with "The name should be".

K.3.2 SGML declaration body

```
[171.2] SGML declaration body =
minimum literal, ps+,
document character set, ps+,
capacity set, ps+,
concrete syntax scope, ps+,
concrete syntax, ps+,
feature use, ps+,
application-specific information,
(ps+, added requirements)?
```

The minimum data of the minimum literal shall be one of the following:

- "ISO 8879:1986"
- "ISO 8879:1986 (ENR)"
- "ISO 8879:1986 (WWW)"

The ENR suffix indicates that the document conforms to Annex J of this International Standard.

The WWW suffix indicates that the document conforms to Annex K (this annex) of this International Standard, which incorporates by reference the provisions of Annex J of this International Standard and supersedes any contradictory provisions elsewhere in this International Standard.

Certain parameters of the SGML declaration body are omissible in order to allow documents to declare conformance to this annex by a simple change to the minimum data. An omitted parameter has no effect on the document; that is, with respect to the subject of the omitted parameter, the document is the same as if it did not conform to this annex.

Note 18: Nevertheless, it is recommended that all parameters be specified.

K.3.3 Capacity Set

```
[180] capacity set =
"CAPACITY", ps+,
(("PUBLIC", ps+, public identifier)|
"NONE"|
("SGMLREF", (ps+, name, ps+, number)*))
```

NONE indicates that no capacities are specified.

Specifying NONE does not require a system to support capacities greater than those specified in its system declaration.

K.3.4 Concrete syntax

```
[182] concrete syntax =
"SYNTAX", ps+,
(public concrete syntax |
(shunned character number identification, ps+,
syntax-reference character set, ps+,
function character identification, ps+,
naming rules, ps+,
delimiter set, ps+,
reserved name use, ps+,
quantity set,
(ps+, predefined data character entities)?))
```

K.3.4.1 Quantity Set

```
[194] quantity set =
"QUANTITY", ps+,
("NONE"|
("SGMLREF", (ps+, name, ps+, number)*))
```

NONE indicates that no quantities are specified.

Specifying NONE does not require a system to support quantities greater than those specified in its system declaration.

K.3.4.2 Predefined data character entities

```
[194.1] predefined data character entities =
"ENTITIES", ps+,
("NONE"|(parameter literal, ps+, character number)+)
```

Each interpreted "parameter literal" must be a valid general entity name in the syntax being defined, and is associated with a character number in the syntax-reference character set. When the named entity is referenced, the replacement text is a numeric character reference to the corresponding character. Predefined data character entities are treated as though defined at the start of the internal subset of all documents in which the concrete syntax is used.

Note 19: For example:

```
ENTITIES "amp" 38 "lt" 60 "gt" 62 "quot" 34 "apos" 39
```

K.3.5 Markup minimization features

```
[196] markup minimization features =
"MINIMIZE", ps+,
"DATATAG", ps+, ("NO"|"YES"), ps+,
"OMITTAG", ps+, ("NO"|"YES"), ps+,
"RANK", ps+, ("NO"|"YES"), ps+,
"SHORTTAG", ps+, ("NO"|"YES"|
(start-tag options, ps+, end-tag options, ps+, attribute-options)),
(ps+, empty element ending rules,
ps+, implied default declarations)?
```

Note 20: Use of an enabled markup minimization feature may be affected by the operation of other provisions of this International Standard, including other markup minimization features.

K.3.5.1 SHORTTAG start-tag options

```
[196.1] start-tag options =
"STARTTAG", ps+,
"EMPTY", ps+, ("NO"|"YES"), ps+,
"UNCLOSED", ps+, ("NO"|"YES"), ps+,
"NETENABL", ps+, ("NO"|"ALL"|"IMMEDNET")
where:
EMPTY YES enables empty start-tags.
UNCLOSED YES enables unclosed start-tags.
NETENABL ALL enables NET-enabling start-tags for beginning all elements
  that are permitted to have end-tags. An element so begun may, but
  need not, be ended with a null end-tag.
NETENABL IMMEDNET ("immediate NET") enables NET-enabling start-tags for
  beginning elements that are permitted to have end-tags and that have
  no text between the start-tag and the end-tag, provided that each
  element so begun is ended with a null end-tag.
```

Note 21: An element with no text between its start-tag and end-tag is not necessarily a mandatorily empty element.

Note 22: A null end-tag can be used for a mandatorily empty element only when EMPTYNRM YES is specified (see K.3.6).

Note 23: For example, if the **nestc** is "/" and the **net** is ">", and "img" is either not a mandatorily empty element or is a mandatorily empty element and EMPTYNRM YES is specified, then:

```
<img/>
```

can be used as an alternative to

```
<img></img>
```

K.3.5.2 SHORTTAG end-tag options

```
[196.2] end-tag options =
"ENDTAG", ps+,
"EMPTY", ps+, ("NO"|"YES"), ps+,
"UNCLOSED", ps+, ("NO"|"YES")
where:
EMPTY YES enables empty end-tags.
UNCLOSED YES enables unclosed end-tags.
```

K.3.5.3 SHORTTAG attribute options

```
[196.3] attribute options =
"ATTRIB", ps+,
"DEFAULT", ps+, ("NO"|"YES"), ps+,
"OMITNAME", ps+, ("NO"|"YES"), ps+,
"VALUE", ps+, ("NO"|"YES")
```

```
where:
DEFAULT YES enables attribute value defaulting (7.9.1.1).
OMITNAME YES enables attribute names and vi to be omitted for
  unique NMTOKEN values (7.9.1.2).
VALUE YES enables some attribute values to be specified without
  delimiters, rather than as literals (7.9.3.1).
```

Note 24: DEFAULT NO does not bar default values from attribute definition list declarations.

K.3.6 Empty element ending rules

```
[196.4] empty element ending rules =
"EMPTYNRM", ps+, ("NO"|"YES")
where:
EMPTYNRM YES applies normal rules for the presence of end-tags, including
  markup minimization rules, to elements of a type declared EMPTY, or
  that are forced to be EMPTY by an explicit content reference
  attribute (7.3.).
```

Note 25: Specifying EMPTYNRM YES applies to mandatorily empty elements the same rules about the presence of end-tags that apply to other kinds of elements.

K.3.7 Implied default declarations

```
[196.5] implied default declarations =
"IMPLYDEF", ps+,
"ATTLIST", ps+, ("NO"|"YES"), ps+,
"DOCTYPE", ps+, ("NO"|"YES"), ps+,
"ELEMENT", ps+, ("NO"|"YES"|"ANYOTHER"), ps+,
"ENTITY", ps+, ("NO"|"YES"), ps+,
"NOTATION", ps+, ("NO"|"YES")
where:
DOCTYPE YES means that an implied document type declaration includes
  an implied declaration for an external subset entity (see K.4.9); and
YES for the other parameters allows information of the specified type
  to be used in a fully-tagged or amply-tagged document instance
  without an explicit declaration. Instead, a declaration is implied,
  as follows:
ATTLIST YES means an undeclared attribute is declared as: CDATA #IMPLIED
ELEMENT YES means an undeclared element type is declared as: - O ANY
ELEMENT ANYOTHER means an undeclared element type is declared as: - O ANY
  but immediately recursive elements of that type are prohibited.
ENTITY YES means an undeclared general entity is declared as: SYSTEM
NOTATION YES means an undeclared notation is declared as: SYSTEM
```

Note 26: IMPLYDEF DOCTYPE YES implies only a declaration for an external subset entity. Other declarations may be implied if needed, and if permitted by the applicable IMPLYDEF parameters.

When IMPLYDEF ELEMENT ANYOTHER and OMITTAG YES are specified, the end-tag of an element of an undeclared type can be omitted from an amply-tagged instance when the element is followed by the start of an element of the same type.

When IMPLYDEF ENTITY YES is specified, #DEFAULT cannot be specified as a general entity name.

Implicitly declared definitions occur in the grove in the order in which it is necessary to imply their declarations. No attribute assignment node is created for an attribute unless the attribute is specified.

Note 27: An implied declaration is not necessarily the same as an explicit declaration for the same object that was ignored during parsing (Note 38). Nor does it constrain any explicit declaration that might be created for that object as a result of processing (for example, when generating an explicit DTD for a document that has none).

K.3.8 Other features

```
[198] other features =
"OTHER", ps+,
"CONCUR", ps+, ("NO"|("YES", ps+, number)), ps+,
"SUBDOC", ps+, ("NO"|("YES", ps+, number)), ps+,
"FORMAL", ps+, ("NO"|"YES"),
(urn feature, keeprsre feature, validity feature, entities feature)?
```

K.3.8.1 Universal Resource Names

```
[198.1] urn feature =
ps+, "URN", ps+, ("NO"|"YES")
where:
URN YES means public identifiers are interpreted according to the
  applicable Internet Engineering Task Force RFC2141 governing
  Universal Resource Names.
```

If both URN and FORMAL are YES, public identifiers are interpreted either as formal public identifiers or as URNs.

K.3.8.2 White space in content

```
[198.2] keeprsre feature =
ps+, "KEEPRSRE", ps+, ("YES"|"NO")
where:
KEEPRSRE YES means clause 7.6.1 does not apply.
```

If KEEPRSRE YES is specified, all white space in mixed content is included in the grove as datachar nodes and all white space in element content is included in the grove as s separator nodes.

Note 28: This option does not affect delimited strings, such as attribute value literals, which have their own rules for normalizing white space (and which, in any case, do not occur in content).

K.3.8.3 Assertions

```
[198.3] validity feature = ps+,
"VALIDITY", ps+, ("NOASSERT"|"TYPE")
where:
NOASSERT makes no validity assertion.
TYPE asserts that document instances are type-valid.
If the parameter is omitted, TYPE is assumed.

[198.4] entities feature = ps+,
"ENTITIES", ps+, ("NOASSERT"|
                  ("REF", ps+, ("NONE"|"INTERNAL"|"ANY"), ps+,
                  "INTEGRAL", ps+, ("NO"|"YES")))
where:
NOASSERT makes no validity assertion.
REF asserts the document either has unconstrained entity references (ANY),
    is external-reference-free (INTERNAL), or is reference-free (NONE).
INTEGRAL YES asserts document instances are integrally-stored.
If the parameter is omitted, NOASSERT is assumed.
```

It is a reportable markup error if a document is less constrained than is asserted.

Note 29: For example, if an otherwise conforming document incorrectly asserts that it is integrally-stored, the document is non-conforming. Had the assertion not been made, the document would have been conforming.

Note 30: A system should offer means, such as a parameter to the invocation of processing, to check whether a particular VALIDITY or ENTITIES assertion could correctly be made for a document, even when the SGML declaration does not make that assertion.

Note 31: To satisfy the classical requirement for SGML conformance, a document instance must be fully-declared as well as type-valid.

K.3.9 Added requirements

```
[199.1] added requirements =
"SEEALSO", ps+, ("NONE"| (ps+, public identifier)+)
```

The public identifiers identify additional requirements for the document, including requirements unrelated to the SGML language. These requirements are in addition to, and must not contradict, the requirements of this International Standard. Failure to satisfy the added requirements is not a reportable markup error.

Note 32: For example, this parameter could be used by an SGML system to signal the existence of requirements for specific entity constraint assertions, formatting conventions for specified element types, or data restrictions, such as that the number of cells in a table row does not exceed the number specified in some attribute. It is not a reportable markup error if the added required entity assertions are not present in the SGML declaration; if they are present, it is a reportable markup error if the document fails to satisfy them, as that would have been the case even in the absence of the added requirements.

It is not an error if the system is unable to access an object named by the public identifiers.

Note 33: See Annex L "Added Requirements for XML" for an example.

K.4 General

K.4.1 Hexadecimal character reference

```
[62] character reference =
named character reference |
numeric character reference |
hex character reference

[62.1] named character reference =
cro, function name, reference end

[62.2] numeric character reference =
cro, character number, reference end

[62.3] hex character reference =
hcro, hexdigit+, reference end
```

A hexdigit is a digit or lowercase a-f or uppercase A-F. The length of the hex digit string cannot exceed NAMELEN.

K.4.2 Delimiters (9.6)

```
Name  String  Number     Mode        Constraint  Description

HCRO  (none)  (none)     CON LIT     HEX         Hex character reference open
NESTC (NET)   (none)     TAG                     NET-enabling start-tag close
NET   /       47         CON         ELEM        Null end-tag
```

HEX constraint: must be followed by a hexdigit.

If no assignment is made to NESTC, the NET is used.

K.4.3 NET-enabling start-tag

```
[18] net-enabling start-tag =
stago,
generic identifier specification, attribute specification list, s*,
nestc
```

K.4.4 Attribute definitions

```
[141] attribute definition list declaration =
mdo, "ATTLIST", ps+,
((associated element type | (rni, ("IMPLICIT"|"ALL")))|
(rni, "NOTATION", ps+,
(associated notation name | (rni, ("IMPLICIT"|"ALL"))))), ps+,
attribute definition list,
ps*, mdc
where the keywords indicate the objects associated with the attributes:
NOTATION indicates that the associated objects are notations.
IMPLICIT means all the implicitly declared element types or notations
  and those whose DTD properties are expressed in DTD data entities.
ALL means all element types or notations.

[149.1] associated notation name =
(notation name | name group)
```

IMPLICIT and ALL are the equivalent of name groups.

The same element type or notation may be the associated object in multiple ATTLIST declarations. An attempt to redefine an attribute that was previously defined for an associated object is not an error; the earliest definition prevails (just as for entity declarations).

However, a definition associated with ALL can be overridden by subsequent attribute declarations for specific element types or notations (including declarations specified with IMPLICIT), except when the attribute has already been specified. It is therefore an error to specify a data attribute that was declared with ALL and then to attempt to redeclare that attribute.

K.4.4.1 Possibly empty attribute definition list

```
[142] attribute definition list =
(attribute definition, (ps+, attribute definition)*)?
```

K.4.4.2 Declared value

```
[145] declared value =
"CDATA"|"ENTITY"|"ENTITIES"|"ID"|"IDREF"|"IDREFS"|"NAME"|"NAMES"|
"NMTOKEN"|"NMTOKENS"|"NUMBER"|"NUMBERS"|"NUTOKEN"|"NUTOKENS"|
notation|name token group|data specification
```

Replace the penultimate paragraph of 11.3.3 with:

> A token can occur more than once in an attribute definition list provided that each occurrence of the token is in the *name token group* or *name group* of a different attribute definition. In an attribute specification where the value is such a duplicated token, omitted attribute name minimization cannot be used.

K.4.4.3 Data specification

```
[145.1] data specification =
"DATA", ps+, notation name, data attribute specification?
where:
"notation name" identifies the data content notation of the
  attribute's value
"data attribute specification" specifies the corresponding data
  attributes
```

The value of an attribute whose declared value is "data specification" is character data.

Note 34: An application may wish to verify that the value of a data specification attribute is meaningful in light of the specified notation name and data attributes, but it is not a reportable markup error if this is not the case.

K.4.5 Implied document type name

```
[111] document type name =
(generic identifier | (rni, "IMPLIED"))
where:
IMPLIED means the document element can have any valid element type
name.
```

It is a reportable markup error if IMPLIED is specified and the start-tag of the document element is omitted or does not contain a generic identifier.

IMPLIED cannot be specified if LINK EXPLICIT YES or CONCUR YES are specified in the SGML declaration.

K.4.6 Internet domain names in public identifiers

```
[80] owner identifier =
ISO owner identifier |
registered owner identifier |
unregistered owner identifier |
internet domain name owner identifier

[83.1] internet domain name owner identifier =
"+//IDN ", minimum data
where the minimum data must begin with an Internet domain name.
```

Note 35: A string like "IDN domain.name" or "IDN domain.name/sub-domain/sub-domain" is treated as an ISO/IEC 9070 "registered owner prefix". Any sub-domains of a domain could also be identified using owner name components. For example, the Internet domain named "someisp.net" and its sub-domains in the URL "http://www.someisp.net/users/mtb" could occur in an FPI as:

```
+//IDN someisp.net::www::users::mtb
```

or as:

```
+//IDN www.someisp.net/users/mtb
```

Note 36: When constructing a public text owner identifier using an Internet domain name, users may wish to consider the name's potential lifespan and the lifespans of the objects to be identified.

Semicolon, exclamation point, asterisk, number sign, commercial at sign, dollar sign, underscore, and percent sign are members of the abstract character class "special", which is usable in minimum data.

K.4.7 Elements

A mandatorily empty element cannot contain any text, including white space, other markup, or included subelements. Markup minimization can be applied to such elements.

K.4.8 Entities

K.4.8.1 Token separators (10.1.3)

A referenced parameter entity must consist of zero or more of the consecutive complete tokens that follow the ts in which the reference occurs in the same group (i.e., at the same nesting level), together with any intervening ts separators and connectors, and optionally with surrounding separators and connectors. The entity must end within the same group.

An Ee can occur in a ts only if the reference to the entity the Ee terminates occurs in the same group (i.e., at the same nesting level).

K.4.8.2 Entity set

```
[113] entity set =
(entity declaration |
notation declaration |
attribute definition list declaration |
ds)*
where:
The keyword "NOTATION" must be specified in each attribute
definition list declaration.
```

K.4.8.3 SGML subdocument entity

```
[3] SGML subdocument entity =
(s*, SGML declaration)?, prolog, document instance set, Ee
```

K.4.9 Omitted prolog (7.1)

If LINK EXPLICIT NO and CONCUR NO are specified in the SGML declaration and either or both of IMPLYDEF DOCTYPE YES or IMPLYDEF ELEMENT YES are specified, and if the document instance is either fully-tagged or amply-tagged, the SGML document entity need not contain a prolog. In such cases, a prolog will be implied consisting solely of a single implied document type declaration, as follows:

- If IMPLYDEF DOCTYPE YES is specified:
  `<!DOCTYPE #IMPLIED SYSTEM>`
- If IMPLYDEF DOCTYPE NO is specified:
  `<!DOCTYPE #IMPLIED>`

Note 37: When both the document type name and the external subset entity are implied in a document type declaration, a system may be able to locate an appropriate external subset by considering the storage identifier of the SGML document entity and/or the generic identifier of the document element.

If permitted by the implied default declarations parameter of the SGML declaration, a document type declaration (whether explicit or implied) may lack declarations for element types, attributes, notations, and/or general entities. Declarations are implied for them as provided in K.3.7.

Note 38: If some or all of an instance's associated document type declaration is unavailable or for other reasons is ignored during parsing, then:

1. It is recommended that the constructed grove be the same as if implied declarations had replaced the ignored ones; and
2. Whether or not the instance is fully-declared, the constructed grove could well differ from the grove that would be constructed by a conforming SGML parser that processes all the declarations that are present.

K.4.10 DTD data entities

K.4.10.1 DTD notations

To the extent that an SGML system can recognize properties expressed by DTD notations, it shall treat them identically to DTD properties that are expressed by markup declarations.

It is a reportable markup error if a DTD data entity is referenced and its representation of required DTD properties is not understood by the system and it is not permissible to imply default declarations for them.

Note 39: Therefore, when implied default declarations are permitted, the use of a DTD notation that cannot be understood has the same effect as omitting the equivalent markup declarations.

K.4.10.2 External parameter entity declaration (10.5.5)

An entity declaration for an external parameter entity can include an entity type that specifies the name of a data notation.

K.4.10.3 External subset entity

Production 110 is replaced with:

```
[110] document type declaration =
mdo, "DOCTYPE", ps+, document type name,
(ps+, external identifier,
(ps+, ("CDATA"|"NDATA"|"SDATA"), ps+, notation name)?)?,
(ps+, dso, document type declaration subset, dsc)?,
ps*, mdc
```

The notation name must be declared within the internal subset of the document type declaration.

Note 40: If data attributes are to be specified for a DTD data entity, the entity must be declared by an entity declaration; that is, the entity cannot be the external subset entity.

K.4.11 Content model (11.2.4, 11.2.5)

```
[129] primitive content token =
( rni , ("PCDATA" | "ALL" | "IMPLICIT")) | element token | data tag group
where:
ALL and IMPLICIT have the same meaning as in K.4.4 and are equivalent
to optional repeatable OR groups.
```

#ALL and #IMPLICIT are allowed in the name groups in [139] and [140]. They have the same meaning as in K.4.4.

Annex L
(informative)
Added Requirements for XML

This annex illustrates the relationship of SGML declarations to "added requirements" by means of a real-world example, XML. It is not intended as a specification for XML.

L.1 Application summary

The Extensible Markup Language (XML) is the core subset of SGML functionality developed by the World Wide Web Consortium (W3C) for exchanging SGML documents over the World Wide Web. The current specification for XML can be found at the W3C web site at "http://www.w3.org/TR/" under the title "Extensible Markup Language (XML)".

The XML specification covers the following aspects of an SGML application and system:

1. Restrictions on the use of some SGML language constructs.
2. A required character set and mandatory encodings.
3. Rules for the behavior of the entity manager.
4. Application semantics of the type appropriate for an enabling architecture; that is, less than a complete DTD and associated semantics. In contrast, SGML applications, such as HTML, would normally have a complete DTD with semantics defined for all element types and attributes.
5. Details of the interface between applications and an SGML/XML parser beyond what is addressed in ISO 8879 and related standards.

XML distinguishes two classes of conforming documents:

- An XML document must be "well-formed" by XML rules: it is a conforming SGML document whose document instance is fully-tagged and integrally-stored.
- A well-formed XML document may also be "valid" by XML rules: its document instance is also type-valid and fully-declared.

A full-SGML validating parser cannot validate documents for conformance to XML unless it is specially modified to support XML. That is because some of XML's language restrictions cannot be expressed in the SGML declaration. Those restrictions can be found in http://www.w3.org/TR/NOTE-sgml-xml.

L.2 SGML Declaration for XML

XML documents implicitly contain the following SGML declaration, which does not require the document to be type-valid. However, type-validity is asserted when a "validating processor" is invoked.

```
<!SGML -- SGML Declaration for XML --
     "ISO 8879:1986 (WWW)"
CHARSET
  BASESET
    "ISO Registration Number 176//CHARSET
    ISO/IEC 10646-1:1993 UCS-4 with implementation
    level 3//ESC 2/5 2/15 4/6"
  DESCSET
    0       9       UNUSED
    9       2       9
    11      2       UNUSED
    13      1       13
    14      18      UNUSED
    32      95      32
    127     1       UNUSED
    128     32      UNUSED
```

```
    160     55136   160
    55296   2048    UNUSED  -- surrogates --
    57344   8190    57344
    65534   2       UNUSED  -- FFFE and FFFF --
    65536   1048576 65536
CAPACITY NONE

SCOPE DOCUMENT

SYNTAX
  SHUNCHAR NONE
  BASESET "ISO Registration Number 176//CHARSET
          ISO/IEC 10646-1:1993 UCS-4 with implementation
          level 3//ESC 2/5 2/15 4/6"
  DESCSET
    0 1114112 0
  FUNCTION
    RE    13
    RS    10
    SPACE 32
    TAB   SEPCHAR 9
  NAMING
    LCNMSTRT ""
    UCNMSTRT ""
    NAMESTRT
      58 95 192-214 216-246 248-305 308-318 321-328
      330-382 384-451 461-496 500-501 506-535 592-680
      699-705 902 904-906 908 910-929 931-974 976-982
      986 988 990 992 994-1011 1025-1036 1038-1103
      1105-1116 1118-1153 1168-1220 1223-1224
      1227-1228 1232-1259 1262-1269 1272-1273
      1329-1366 1369 1377-1414 1488-1514 1520-1522
      1569-1594 1601-1610 1649-1719 1722-1726
      1728-1742 1744-1747 1749 1765-1766 2309-2361
      2365 2392-2401 2437-2444 2447-2448 2451-2472
      2474-2480 2482 2486-2489 2524-2525 2527-2529
      2544-2545 2565-2570 2575-2576 2579-2600
      2602-2608 2610-2611 2613-2614 2616-2617
      2649-2652 2654 2674-2676 2693-2699 2701
      2703-2705 2707-2728 2730-2736 2738-2739
      2741-2745 2749 2784 2821-2828 2831-2832
      2835-2856 2858-2864 2866-2867 2870-2873 2877
      2908-2909 2911-2913 2949-2954 2958-2960
      2962-2965 2969-2970 2972 2974-2975 2979-2980
      2984-2986 2990-2997 2999-3001 3077-3084
      3086-3088 3090-3112 3114-3123 3125-3129
      3168-3169 3205-3212 3214-3216 3218-3240
      3242-3251 3253-3257 3294 3296-3297 3333-3340
      3342-3344 3346-3368 3370-3385 3424-3425
      3585-3630 3632 3634-3635 3648-3653 3713-3714
      3716 3719-3720 3722 3725 3732-3735 3737-3743
      3745-3747 3749 3751 3754-3755 3757-3758 3760
      3762-3763 3773 3776-3780 3904-3911 3913-3945
      4256-4293 4304-4342 4352 4354-4355 4357-4359
      4361 4363-4364 4366-4370 4412 4414 4416 4428
      4430 4432 4436-4437 4441 4447-4449 4451 4453
      4455 4457 4461-4462 4466-4467 4469 4510 4520
      4523 4526-4527 4535-4536 4538 4540-4546 4587
      4592 4601 7680-7835 7840-7929 7936-7957
      7960-7965 7968-8005 8008-8013 8016-8023 8025
      8027 8029 8031-8061 8064-8116 8118-8124 8126
      8130-8132 8134-8140 8144-8147 8150-8155
      8160-8172 8178-8180 8182-8188 8486 8490-8491
      8494 8576-8578 12295 12321-12329 12353-12436
      12449-12538 12549-12588 19968-40869 44032-55203
```

```
    LCNMCHAR ""
    UCNMCHAR ""
    NAMECHAR
      45-46 183 720-721 768-837 864-865 903 1155-1158
      1425-1441 1443-1465 1467-1469 1471 1473-1474
      1476 1600 1611-1618 1632-1641 1648 1750-1764
      1767-1768 1770-1773 1776-1785 2305-2307 2364
      2366-2381 2385-2388 2402-2403 2406-2415
      2433-2435 2492 2494-2500 2503-2504 2507-2509
      2519 2530-2531 2534-2543 2562 2620 2622-2626
      2631-2632 2635-2637 2662-2673 2689-2691 2748
      2750-2757 2759-2761 2763-2765 2790-2799
      2817-2819 2876 2878-2883 2887-2888 2891-2893
      2902-2903 2918-2927 2946-2947 3006-3010
      3014-3016 3018-3021 3031 3047-3055 3073-3075
      3134-3140 3142-3144 3146-3149 3157-3158
      3174-3183 3202-3203 3262-3268 3270-3272
      3274-3277 3285-3286 3302-3311 3330-3331
      3390-3395 3398-3400 3402-3405 3415 3430-3439
      3633 3636-3642 3654-3662 3664-3673 3761
      3764-3769 3771-3772 3782 3784-3789 3792-3801
      3864-3865 3872-3881 3893 3895 3897 3902-3903
      3953-3972 3974-3979 3984-3989 3991 3993-4013
      4017-4023 4025 8400-8412 8417 12293 12330-12335
      12337-12341 12441-12442 12445-12446 12540-12542
  NAMECASE GENERAL NO  ENTITY NO
  DELIM
      GENERAL  SGMLREF
      HCRO     "&#38;#x"
               -- Ampersand followed by "#x" (without quotes) --
      NESTC    "/"
      NET      ">"
      PIC      "?>"
      SHORTREF NONE
  NAMES     SGMLREF
  QUANTITY  NONE
  ENTITIES  "amp" 38 "lt" 60 "gt" 62 "quot" 34 "apos" 39

FEATURES
  MINIMIZE
    DATATAG NO
    OMITTAG NO
    RANK    NO
    SHORTTAG
      STARTTAG EMPTY    NO  UNCLOSED NO  NETENABL IMMEDNET
      ENDTAG   EMPTY    NO  UNCLOSED NO
      ATTRIB   DEFAULT  YES OMITNAME NO  VALUE    NO
    EMPTYNRM YES
    IMPLYDEF ATTLIST  YES DOCTYPE  NO  ELEMENT  YES
             ENTITY   NO  NOTATION YES
  LINK  SIMPLE   NO  IMPLICIT NO  EXPLICIT NO
  OTHER CONCUR   NO  SUBDOC   NO  FORMAL   NO  URN  NO
        KEEPRSRE YES VALIDITY NOASSERT ENTITIES REF ANY  INTEGRAL YES

APPINFO NONE

SEEALSO "ISO 8879//NOTATION
         Extensible Markup Language (XML) 1.0//EN"
>
```

Note that the public identifier in the SEEALSO parameter identifies the XML 1.0 Recommendation, which is owned by the W3C. The string "ISO 8879" merely indicates that this particular public identifier of that document was created in this International Standard.