TECHNICAL REPORT

ISO/IEC TR 24716

First edition
2007-12-15

Information technology — Programming languages, their environment and system software interfaces — Native COBOL Syntax for XML Support

Technologies de l'information — Langages de programmation, leur environnement et interfaces du logiciel système — Syntaxe COBOL native pour support XML

Reference number
ISO/IEC TR 24716:2007(E)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

In exceptional circumstances, the joint 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 the joint 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.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC TR 24716, which is a Technical Report of type 2, was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*, in collaboration with INCITS Techmical Committee J4, Programming language COBOL.

Introduction

This Technical Report provides extensions so that COBOL can process XML documents as easily as it can read files. The new syntax to process XML documents,

- is based on the familiar approach used with COBOL I/O support,
- provides Document Object Model (DOM) style parsing,
- handles multiple input sources to handle XML in an extremely flexible manner,
- reads, updates, and writes XML documents,
- checks that XML documents are well-formed, and
- provides an optional validity check of an XML document against a schema or Document Type Definition (DTD).

Technical Report ISO/IEC 24716 extends the COBOL specification defined in ISO/IEC 1989:2002, *Information technology — Programming languages — COBOL*. It provides new syntax to read, write, and update XML documents in COBOL.

Annex A forms a normative part of this Technical Report. Annex B and Annex C and the Bibliography are for information only.

Information technology — Programming languages, their environment and system software interfaces — Native COBOL Syntax for XML Support

1 Scope

This Technical Report specifies the syntax and semantics for XML support in COBOL. The purpose of this Technical Report is to promote a high degree of portability in implementations, even though some elements are subject to trial before completion of a final design suitable for standardization.

This specification builds on the syntax and semantics defined in ISO/IEC 1989:2002.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 1989:2002, *Information technology — Programming languages — COBOL*

Extensible Markup Language (XML) 1.0 (Fourth Edition), W3C Recommendation, 16 August 2006

Extensible Markup Language (XML) 1.1 (Second Edition), W3C Recommendation, 16 August 2006

Namespaces in XML 1.1, W3C Recommendation, 4 February 2004

XML Schema Part 1: Structures, W3C Recommendation, 28 October 2004

XML Schema Part 2: Datatypes, W3C Recommendation, 28 October 2004

3 Conformance to this Technical Report

This Technical Report is based on ISO/IEC 1989:2002. Conformance to this Technical Report does not require a full implementation of ISO/IEC 1989:2002. The interaction of the features of this technical report with features that are not provided by an implementation of ISO/IEC 1989:2002 is processor dependent.

4 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

4.1
document type definition
DTD
specification of the markup language that defines the elements, attributes, comments, and entities that a document may contain and specifies the relationships among them within the document

4.2
external document type definition
external DTD
DTD that is outside the file containing the XML document

4.3
internal document type definition
internal DTD
DTD that is in the same file as the XML document

4.4
XML document
unit of data that is well-formed as defined either in XML 1.0 or XML 1.1

4.5
XML element
portion of an XML document, the boundaries of which are either delimited by start-tags and end-tags, or, for empty elements, by an empty-element tag

NOTE Each element has a type, identified by name, and may have a set of attribute specifications.

4.6
XML file
file with XML organization

4.7
XML Schema
XML language for constructing schemas that describe the syntax of XML documents. Each schema defines a class of XML documents by constraining the structures and data types of instance documents that conform to the schema

5 Description techniques

Description techniques and language fundamentals are the same as those described in ISO/IEC 1989:2002.

6 Changes to ISO/IEC 1989:2002

These changes refer to clause and rule numbers in ISO/IEC 1989:2002.

6.1 Changes to 8, Language fundamentals

[a] Add the following reserved words to the list in 8.9, Reserved Words

DOCUMENT
END-OPEN
IDENTIFIED
VERSION-XML

[b] In 8.10, Context-sensitive words, change the context for ATTRIBUTE to read:

"IDENTIFIED clause, DELETE statement, READ statement, REWRITE statement, SET statement, START statement, and WRITE statement"

[c] In 8.10, Context-sensitive words, change the context for ONLY to read:

"Object-view, SHARING clause of the file control entry, SHARING phrase of the OPEN statement, USAGE clause, READ statement, REWRITE statement, and WRITE statement"

[d] Add the following context-sensitive words to the list in 8.10, Context-sensitive words

CHECK	TYPE clause of the file control entry
DISCARD	CLOSE statement
DTD	TYPE clause of the file control entry
ELEMENT	DELETE statement, READ statement, REWRITE statement, START statement, and WRITE statement

NAMESPACE	IDENTIFIED clause
RAW	IDENTIFIED clause
SCHEMA	TYPE clause of the file control entry
STACK	OPEN statement
VALIDITY	TYPE clause of the file control entry
XML	ACCESS clause and ORGANIZATION clause

6.2 Changes to 9, I-O, objects, and user-defined functions

[a] 9.1.3, File connector, second paragraph, second sentence, add "XML" to the end of the list of types of file organization in the parenthetical and to the end of the list of access modes in the parenthetical.

[b] 9.1.6, Fixed file attributes, change the third sentence to read:

"The file organizations are sequential, relative, indexed, and XML."

[c] 9.1.7, Organization,

[1] Change the first sentence to read:

"The file organizations are sequential, relative, indexed, and XML."

[2] add 9.1.7.4, XML, as follows:

"9.1.7.4 XML

XML is a file organization used for data that is in the format of an XML document or a sequence of XML documents. An XML document can be on a traditional file medium or it can be in memory. An XML document is a string of text that is given a structure by the presence of tags, which separate the document into elements. XML documents are organized in a hierarchical manner, similar to a COBOL record structure, where an element may contain other elements. Within the context of its superordinate elements, each subordinate element may be stored, retrieved, or deleted based on its tag name. The element position vector for an XML file associates an element position or an attribute position with each data item for which an IDENTIFIED clause is specified.

COBOL allows multiple documents in a file that is open in the input, output, or extend mode. COBOL allows only one document in a file that is open in the i-o mode. The XML 1.0 and 1.1 recommendations do not consider the processing of anything beyond a single document. Therefore, many XML processors might not consider multiple-document files to be well-formed.

The file format OPEN statement for an XML file establishes a connection to the physical file. The file position indicator maintains the document number of the current document in the XML file. An XML-document format OPEN statement creates an internal representation of the current document. The READ statement transfers data into associated data items described in the file section. The START statement positions the element position vector to attributes or elements within the internal representation. The DELETE statement, REWRITE statement, and WRITE statement change the internal representation, but do not change the physical file. The XML-document format CLOSE statement optionally writes an XML document based on the internal representation and then deletes the internal representation; the file remains in the open state. A file format CLOSE statement closes the physical file.

COBOL processes documents that conform to XML 1.0 or XML 1.1.

On input, COBOL resolves character references, predefined entity references, general entity references, and parameter entity references consistent with the TYPE clause of the file control entry, and makes the content of CDATA sections available to the program. COBOL does end-of-line normalization and attribute-value normalization as specified in the applicable XML recommendation. All required parts of the XML document are made available to the COBOL program with the exception of Processing Instructions.

COBOL, by default writes XML documents that are well-formed with respect to XML 1.0 or optionally to XML 1.1.

All XML documents written by COBOL start with an XML declaration that includes the version information and an encoding declaration. If a DTD is included, the XML declaration also includes a standalone document declaration with a value of “yes.”

XML documents written by COBOL contain predefined references and character references as required by the appropriate XML Recommendation.

XML documents written by COBOL do not contain general entity references, parameter entity references, CDATA sections, processing instructions, or white space. The implementor may include comments.

NOTE The term 'white space' is defined in XML 1.1."

[d] 9.1.8, Access modes,

[1] In the first paragraph, last sentence, add "XML" to the list of access modes.

[2] In the second paragraph, add the following as a new last sentence:

"A file with organization XML may be accessed only in XML access mode."

[3] Add the following:

"9.1.8.4 XML access mode

When a file is accessed in XML access mode, the order of access of the documents in that file is the sequential order of the documents in that file.

Within each document, START statements may be used to set the element position vector to a specific occurrence of an attribute or element."

[e] 9.1.11, File position indicator, second paragraph, insert the following text after "for a relative file,":

"the document number of the current document for an XML file,"

[f] Add the following after 9.1.11, File position indicator:

"9.1.11a Element position vector

The element position vector is a conceptual entity that exists for each XML file to maintain the positions of attributes and elements within each level in the hierarchy of an XML document. Position is maintained for each attribute or element in the COBOL record description that is described with an IDENTIFIED clause. Attributes and elements may be accessed by their attribute names or element names, respectively, as long as all containing elements are indicated by the element position vector.

The element position vector is set only by CLOSE, OPEN, READ, and START statements. The XML-document format OPEN statement for a file open in input mode or i-o mode sets the element position vector to the first XML element associated with each COBOL data item that has an IDENTIFIED clause and to the first attribute, if any, of each of those elements. The XML-document format OPEN statement for a file open in extend mode or output mode sets the element position vector at the start of a new document. The START statement positions the element position vector to a specific occurrence of an attribute or element. The READ statement accesses occurrences of elements sequentially and repositions the element position vector. The READ statement accesses occurrences of attributes sequentially when the associated IDENTIFIED clause contains the USING phrase and repositions the element position vector. The READ statement accesses the specified occurrence of an attribute when the associated IDENTIFIED clause contains the BY phrase, but does not reposition the element position vector. The CLOSE statement causes the element position vector to indicate that no valid position has been established."

[g] 9.1.12.1, Successful completion, add the following new entries:

“6) I-O status = 08. The input-output statement is successfully executed but a READ statement ignored one or more XML attributes or elements, or both, because there is no COBOL data item with which one or more attributes or elements can be associated.

7) I-O status = 09. The input-output statement is successfully executed but the referenced DTD or schema is not available; the requested validity check of the XML document was not done."

[h] 9.1.12.4, Invalid key condition with unsuccessful completion, I-O status 23, rule 3, change the period to "; or" at the end of 3d and insert the following text:

"e) an attempt is made to access a document that does not exist at the current file position in an XML file; or

f) an attempt is made to access an attribute or element that does not exist at the current position in the internal representation of an XML file."

[i] 9.1.12.4, Invalid key condition with unsuccessful completion, I-O status 24, rule 4, change to read:

"4) I-O status = 24. This condition exists because:

"a) an attempt is made to write beyond the externally-defined boundaries of a physical relative or indexed file; the implementor specifies the manner in which these boundaries are defined; or

b) a sequential WRITE statement is attempted for a relative file and the number of significant digits in the relative record number is larger than the size of the relative key data item described for the file; or

c) an attempt is made to close an XML document in an in-memory XML file and the XML document to be created is larger than the associated length of that memory."

[j] 9.1.12.4, Invalid key condition with unsuccessful completion, add the following text:

"5) I-O status = 25. An attempt is made to delete an XML attribute or element at an invalid position."

[k] 9.1.12.5, Permanent error condition with unsuccessful completion, rule 3, I-O status = 34, first sentence, insert the following text after "externally-defined boundaries of":

"an XML file or"

[l] 9.1.12.5, Permanent error condition with unsuccessful completion, add the following text:

"8) I-O status = 3A. An OPEN statement is attempted on an XML document that is not well-formed.

9) I-O status = 3B. A CLOSE statement is attempted on an XML document that is not valid.

10) I-O status = 3C. An OPEN statement with the I-O phrase is attempted on a multiple document file.

11) I-O status = 3D. An OPEN statement is attempted on a document in a file that is open in the input or i-o mode and the coded character set used for encoding the document could not be determined."

[m] 9.1.12.6, Logic error condition with unsuccessful completion, add the following text:

"10) I-O status = 4A. An attempt is made to write XML that is not well-formed by a document-format CLOSE statement, a REWRITE statement, or a WRITE statement.

11) I-O status = 4B. An attempt is made to reference a file connector that is not open by an XML-document format CLOSE statement or an XML-document format OPEN statement.

12) I-O status = 4C. The execution of a READ statement is unsuccessful because duplicate names are specified in IDENTIFIED clauses at the same level in the hierarchy of the associated record description entry.

13) I-O status = 4D. An attempt is made by an XML-document format CLOSE statement to reference a file connector that does not have an open document.

14) I-O status = 4E. An attempt is made by a WRITE or REWRITE statement to create XML text with an attribute name or element name that cannot be represented in the coded character set used for the encoding of the XML document."

6.3 Changes to 12, Environment division

[a] Add a new file control entry format to 12.3.4.1, General format:

"Format 5 (XML):

$$\underline{\text{SELECT}}\ [\ \underline{\text{OPTIONAL}}\ ]\ \text{file-name-1}$$

$$\underline{\text{ASSIGN}} \left\{ \begin{array}{l} \text{TO} \left\{ \begin{array}{l} \left\{ \begin{array}{l} \text{device-name-1} \\ \text{literal-1} \end{array} \right\} \ldots\ [\ \underline{\text{USING}}\ \text{data-name-1}\ ] \\ \underline{\text{DATA}}\ \text{data-name-9} \\ \text{data-name-10}\ \underline{\text{LENGTH}}\ \text{IS data-name-11} \end{array} \right\} \\ \underline{\text{USING}}\ \text{data-name-1} \end{array} \right\}$$

$$[\ \underline{\text{ACCESS}}\ \text{MODE IS}\ \underline{\text{XML}}\ ]$$

$$[\ \text{FILE}\ \underline{\text{STATUS}}\ \text{IS data-name-4}\ ]$$

$$\underline{\text{ORGANIZATION}}\ \text{IS}\ \underline{\text{XML}}$$

$$\left[\underline{\text{TYPE}}\ \text{IS} \left\{ \begin{array}{l} \underline{\text{DTD}} \\ \underline{\text{EXTERNAL}} \left\{ \begin{array}{l} \underline{\text{DTD}} \\ \underline{\text{SCHEMA}} \end{array} \right\} \left\{ \begin{array}{l} \text{literal-3} \\ \text{data-name-12} \end{array} \right\} \end{array} \right\} \left[\underline{\text{CHECK}}\ \text{VALIDITY ON} \left\{ \begin{array}{l} |\ \underline{\text{INPUT}}\ | \\ |\ \underline{\text{OUTPUT}}\ | \end{array} \right\} \right] \right]$$

$$[\ \underline{\text{VERSION-XML}}\ \ \text{IS literal-4}\ ].$$

NOTE ORGANIZATION is a required word in the XML format, rather than optional, so that XML can be a context-sensitive word. "

[b] Add the following new file control entry syntax rules to 12.3.4.2:

"FORMAT 5

"14) The physical file associated with file-name-1 shall have organization XML. The associated file description entry shall not be a sort-merge file description entry.

15) Data-name-9 shall reference a data item of category alphanumeric or national and shall not be subordinate to the file description entry for file-name-1.

16) Data-name-10 shall reference a data item of category data-pointer and shall not be subordinate to the file description entry for file-name-1.

17) Data-name-11 shall reference an unsigned integer data item described without the picture symbol 'P' and shall not be subordinate to the file description entry for file-name-1.

18) Data-name-12 shall reference a data item of category alphanumeric or national and shall not be subordinate to the file description entry for file-name-1.

19) The OPTIONAL phrase shall not be specified when the DATA or LENGTH phrase is specified."

[c] Change 12.3.4.3 file control entry general rule 1:

[1] subrule 1b, add data-name-9, data-name-10, and data-name-11 to both lists.

[2] add the following additional subrules:

"n) The same specification of the TYPE clause, when data-name-12, if specified, references an external data item.

o) The same specification of the VERSION-XML clause."

[d] Change 12.3.4.3 file control entry general rule 3:

[1] add the following text at the end of the first sentence:

"; or to a memory location referenced by either data-name-9 or the content of data-name-10"

[2] subrule 3a, change in part to read:

"When device-name-1 or literal-1 is specified and the USING phrase is omitted …"

[3] add the following new subrules:

"c) If data-name-9 is specified, one or more XML documents are contained in the data item referenced by data-name-9. The XML document is said to be in-memory.

d) If data-name-10 is specified, one or more XML documents begin at the address specified by the content of the data item referenced by data-name-10 and continue for the number of bytes specified by the content of the data item referenced by data-name-11. The XML document is said to be in-memory."

[e] Add the following new file control entry general rule to 12.3.4.3:

"FORMAT 5

“10) The XML format defines a file connector for an XML file. "

[f] In 12.3.4.4.1, general format of the ACCESS MODE clause, add "XML" to the stack in the curly braces.

[g] In 12.3.4.4.2, syntax rules of the ACCESS MODE clause, add the following new syntax rules:

"3) The DYNAMIC, RANDOM, and SEQUENTIAL phrases shall not be specified for an XML file.

4) The XML phrase may be specified only for an XML file."

[h] In 12.3.4.4.3, general rules of the ACCESS MODE clause,

[1] Change general rule 1 to read:

"1) If the ACCESS MODE clause is not specified,

a) if the organization is XML, XML access is assumed,

b) for other organizations, sequential access is assumed."

[2] add the following new general rule:

"5) If the access mode is XML, documents within the file are accessed in sequential order. Elements in the current document can be accessed via their element names only when all containing elements are indicated by the element position vector."

[i] In 12.3.4.9.1, General format [ORGANIZATION clause],

[1] Add the heading "FORMAT 1 (sequential-relative-indexed)" before the existing format.

[2] Add the following after the existing format:

"Format 2 (XML):

<u>ORGANIZATION</u> IS <u>XML</u>"

[j] In 12.3.4.9.2, General rules [ORGANIZATION clause], add the following:

“4a) The XML phrase specifies that the file organization is XML. XML organization is a permanent logical file structure in which each constituent record is an XML document."

[k] Make the following changes to 12.3.6, SAME clause

[1] Syntax rule 7, insert the following text after "a report file":

", an XML file,"

[2] Add the following new syntax rule:

"6a) A given file-name that represents an XML file may be specified in one file-area format SAME clause, and shall not be specified in a record-area format or sort-merge-area format SAME clause."

[l] Add the following after 12.3.4.15, SHARING clause:

"12.3.4.16 TYPE clause

The TYPE clause specifies the SCHEMA or DTD that describes the XML document.

12.3.4.16.1 General format

$$\underline{\text{TYPE}}\ \text{IS} \left\{ \begin{array}{l} \underline{\text{DTD}} \\ \underline{\text{EXTERNAL}} \left\{ \begin{array}{l} \underline{\text{DTD}} \\ \underline{\text{SCHEMA}} \end{array} \right\} \left\{ \begin{array}{l} \text{literal-1} \\ \text{data-name-1} \end{array} \right\} \end{array} \right\} \left[\underline{\text{CHECK}}\ \text{VALIDITY ON} \left\{ \left| \begin{array}{l} \underline{\text{INPUT}} \\ \underline{\text{OUTPUT}} \end{array} \right| \right\} \right]$$

12.3.4.16.2 Syntax rules

1) Literal-1 shall be an alphanumeric or national literal and shall not be a figurative constant.

2) Data-name-1 may be qualified.

12.3.4.16.3 General rules

1) If the TYPE clause is specified without the EXTERNAL phrase, then the document is described by an internal DTD.

2) If the EXTERNAL phrase and the DTD phrase are specified, the document is described by the external DTD specified by literal-1 or data-name-1.

3) If the SCHEMA phrase is specified, the document is described by the schema specified by literal-1 or data-name-1. Schemas are defined in XML Schema Part 1: Structures and XML Schema Part 2: Datatypes.

4) If the INPUT phrase is specified, when an XML-document format OPEN statement is executed, the specified DTD or schema is used to check the document for validity.

5) If the OUTPUT phrase is specified, when an XML-document format CLOSE statement is executed, the specified DTD or schema is used to check the document for validity.

12.3.4.17 VERSION-XML clause

The VERSION-XML clause specifies the version of the XML specification to which created XML files conform.

12.3.4.17.1 General format

$$\underline{\text{VERSION-XML}}\ \text{IS literal-1}$$

12.3.4.17.2 Syntax Rules

1) Literal-1 shall be an alphanumeric or national literal with the value '1.0' or '1.1'.

12.3.4.17.3 General Rules

1) When an XML-document format CLOSE statement is executed for a file connector open in the extend or output mode,

   - if literal-1 is 1.1, it is written as the version number in the XML declaration; otherwise,
   - 1.0 is written as the version number in the XML declaration."

6.4 Changes to 13, Data division

[a] 13.3.4.1, General format of the file description entry, add the following new format:

"Format 4 (XML):

$$\begin{array}{l} \underline{\text{FD}}\ \text{file-name-1} \\ \qquad [\ \text{IS}\ \underline{\text{EXTERNAL}}\ [\ \underline{\text{AS}}\ \text{literal-1}\ ]\ ] \\ \qquad [\ \text{IS}\ \underline{\text{GLOBAL}}\ ] \\ \qquad \left[\underline{\text{CODE-SET}}\ \text{IS} \left\{ \begin{array}{l} \text{alphabet-name-3} \\ \text{data-name-6} \end{array} \right\} \right]\ .\ " \end{array}$$

[b] 13.3.4.2, Syntax rules of the file description entry, add the following new rules:

"FORMAT 4

"9) Format 4 is the file description entry for an XML file.

10) One or more record description entries shall be associated with the XML file description entry.

11) Subordinate entries shall define data items that are category alphabetic, alphanumeric, boolean, national, or numeric."

[c] Add the following new clauses in alphabetical order to the format 1 data description entry, in 13.14.1:

"$\underline{\text{COUNT}}$ IN data-name-7"

$$\text{"}\underline{\text{IDENTIFIED}}\ \left\{\begin{array}{l}\underline{\text{BY}}\left\{\begin{array}{l}\text{data-name-8}\\\text{literal-2}\end{array}\right\}\\\underline{\text{USING}}\ \text{data-name-9}\end{array}\right\}\ \text{IS}\left\{\begin{array}{l}\underline{\text{ATTRIBUTE}}\\\underline{\text{ELEMENT}}\ [\,\underline{\text{RAW}}\,]\end{array}\right\}\left[\underline{\text{NAMESPACE}}\left\{\begin{array}{l}\text{IS}\left\{\begin{array}{l}\text{data-name-10}\\\text{literal-3}\\\underline{\text{NULL}}\end{array}\right\}\\\underline{\text{USING}}\ \text{data-name-11}\end{array}\right\}\right]\text{"}$$

[d] CODE-SET clause, 13.16.13:

[1] In 13.16.13.1, General format, add the heading "FORMAT 1 (sequential):" before the current syntax diagram and add the following after that syntax diagram:

"Format 2 (XML):

$$\underline{\text{CODE-SET}}\ \text{IS}\left\{\begin{array}{l}\text{alphabet-name-3}\\\text{data-name-1}\end{array}\right\}\text{"}$$

[2] In 13.16.13.2, Syntax rules, add the heading "FORMAT 1" before the current syntax rules and the following after those rules:

"FORMAT 2:

4) Alphabet-name-3 shall reference an alphabet that defines a coded character set.

5) Data-name-1 may be qualified.

6) Data-name-1 shall reference either an alphanumeric or a national data item."

[3] In 13.16.13.3, General rules, old general rules 1, 6, and 7 apply to both formats. Old general rules 2 through 5 apply only to format 1. The following new general rules apply only to format 2:

"8) If the file connector associated with the file description in which this clause is specified is open in the extend or output mode, during the execution of an XML-document format OPEN statement the operating environment shall save the encoding referenced by the specified code-set. The execution of an XML-document format CLOSE statement referencing this file connector writes this saved encoding as the encoding declaration in the XML document.

If alphabet-name-3 is specified, the coded character set used to represent data on the storage medium is the one referenced by alphabet-name-3.

If data-name-1 is specified, the coded character set used to represent data is the one whose name is contained in the data item referenced by data-name-1 after the leading and trailing spaces are deleted.

9) If the file connector associated with the file description in which this clause is specified is open in the input, i-o, or extend mode, during the execution of an XML-document format OPEN statement:

a) If alphabet-name-3 is specified and the referenced encoding does not match the encoding declaration specified in the document, the EC-XML-CODESET exception is set to exist. If a RESUME statement with the NEXT STATEMENT phrase is executed in an associated declarative and the implementation can determine the coded character set used for encoding the document, processing continues with the OPEN statement and the OPEN statement is successful.

b) If data-name-1 is specified, the encoding declaration specified in the document is moved into the data item referenced by data-name-1.

10) The specified encoding is used for code-set conversion of all data items in the document. Both UTF-8 and UTF-16 shall be supported by the implementation. If the CODE-SET clause is not specified, UTF-8 is used by default."

[e] Add the following new clause 13.16.15a, COUNT clause: