

1 **IEEE P1484.12.3/D2**  
2 **Draft Standard for—**  
3 **Extensible Markup Language (XML) Schema**  
4 **Definition Language Binding for Learning**  
5 **Object Metadata**

6 Sponsor

7 **Learning Technology Standards Committee**  
8 of the  
9 **IEEE Computer Society**

10

11 **Abstract:** This standard defines a World Wide Web Consortium (W3C) Extensible  
12 Markup Language (XML) Schema definition language binding of the learning object  
13 metadata (LOM) data model defined in IEEE 1484.12.1–2002 *Standard for Learning*  
14 *Object Metadata*. The purpose of this standard is to allow the creation of LOM in-  
15 stances in XML. This allows for interoperability and the exchange of LOM instances  
16 between various systems. This standard uses the W3C XML Schema definition lan-  
17 guage to define the syntax and semantics of the XML encodings.

18 **Keywords:** learning object metadata, LOM, LOM instance, LOM XML Schema  
19 binding, 1484.12.1–2002, Extensible Markup Language, XML, XML Schema defini-  
20 tion, XSD, W3C XML Schema definition language, metadata.

21 \_\_\_\_\_

22 The Institute of Electrical and Electronics Engineers, Inc.  
23 Three Park Avenue, New York, NY 10016-5997, USA

24 Copyright © The Institute of Electrical and Electronics Engineers, Inc.  
25 All rights reserved. Published **date to be supplied**. Printed in the United States of  
26 America

27 This document is an unapproved draft of a proposed IEEE Standard. As such, this  
28 document is subject to change. USE AT YOUR OWN RISK! Because this is an un-  
29 approved draft, this document must not be utilized for any conformance/compliance  
30 purposes. Permission is hereby granted for IEEE Standards Committee participants  
31 to reproduce this document for purposes of IEEE standardization activities only. Prior  
32 to submitting this document to another standards development organization for stan-

33 dardization activities, permission must first be obtained from the Manager, Standards  
 34 Licensing and Contracts, IEEE Standards Activities Department. Other entities seek-  
 35 ing permission to reproduce this document, in whole or in part, must obtain permis-  
 36 sion from the Manager, Standards Licensing and Contracts, IEEE Standards Activi-  
 37 ties Department.

38 IEEE Standards Department  
 39 Standards Licensing and Contracts  
 40 445 Hoes Lane, P.O. Box 1331  
 41 Piscataway, NJ 08855-1331, USA

42

43 *[Note: Information about IEEE LTSC P1484.12.3 can be found at:*

44

45 <http://ltsc.ieee.org/wg12>

46

47 *This note will be removed upon reaching the final draft of this IEEE document.]*

## 48 Introduction

49 (This introduction is not part of P1484.12.3, Draft Standard for Extensible Markup Lan-  
 50 guage (XML) Schema Binding for Learning Object Metadata.)

51 This standard defines a W3C Extensible Markup Language XML Schema definition lan-  
 52 guage binding for the LOM data model defined in IEEE 1484.12.1–2002 *Standard for*  
 53 *Learning Object Metadata*.

## 54 Participants

55 At the time this standard was completed, the working group had the following member-  
 56 ship:

57

Wayne Hodgins, <i>Chair</i> Erik Duval and Scott Lewis, <i>Technical Editors</i>		
To be supplied		

58 The following persons were on the balloting committee: (To be provided by IEEE editor at  
 59 time of publication.)

60 **To be supplied**

61 Also included are the following nonvoting IEEE-SA Standards Board liaisons:

62 **To be supplied**

63	Contents	
64		
65	<b>1. Overview</b> .....	<b>1</b>
66	1.1 Scope .....	1
67	1.2 Purpose .....	1
68	<b>2. References</b> .....	<b>1</b>
69	<b>3. Definitions and acronyms</b> .....	<b>2</b>
70	3.1 Definitions.....	2
71	3.2 Acronyms and abbreviations.....	4
72	<b>4. Conformance</b> .....	<b>4</b>
73	4.1 Strictly conforming LOM instances .....	4
74	4.2 Conforming LOM instances.....	4
75	<b>5. LOM XML Schema binding definition</b> .....	<b>4</b>
76	5.1 General information.....	5
77	5.1.1 Smallest permitted maximums .....	5
78	5.1.2 Post-parsing validation.....	5
79	5.1.3 Extension support.....	5
80	5.2 LOM namespaces .....	6
81	5.3 Tabular format and table organization.....	6
82	5.4 LOM.....	7
83	5.4.1 General.....	9
84	5.4.2 Life Cycle .....	11
85	5.4.3 Meta-Metadata.....	12
86	5.4.4 Technical.....	14
87	5.4.5 Educational .....	16
88	5.4.6 Rights .....	19
89	5.4.7 Relation.....	20
90	5.4.8 Annotation .....	22
91	5.4.9 Classification .....	22
92	5.5 Common data types and elements .....	23
93	5.5.1 CharacterString.....	23
94	5.5.2 DateTime .....	24
95	5.5.3 Duration .....	25
96	5.5.4 LangString .....	26
97	5.5.5 Language.....	27
98	5.5.6 Token .....	27
99	5.5.7 vCard.....	28
100	5.5.8 Vocabulary.....	28
101	<b>Annex A (informative) Bibliography</b> .....	<b>29</b>
102	<b>Annex B (informative) Internet availability of XSD files</b> .....	<b>30</b>
103	<b>Annex C (informative) XSD file descriptions</b> .....	<b>31</b>
104	<b>Annex D (informative) XSD implementation choices</b> .....	<b>37</b>
105		
106		

107 **Draft Standard for—**  
108 **Extensible Markup Language (XML) Schema**  
109 **Definition Language Binding for Learning**  
110 **Object Metadata**

111 **1. Overview**

112 The scope and purpose of this standard are discussed in 1.1 and 1.2.

113 **1.1 Scope**

114 This standard defines a World Wide Web Consortium (W3C) Extensible Markup Language  
115 (XML) Schema definition language (see XML Schema, Parts 1 & 2) binding of the learning  
116 object metadata (LOM) data model defined in IEEE 1484.12.1–2002 *Standard for Learning*  
117 *Object Metadata*. An implementation that conforms to this standard shall conform to IEEE  
118 1484.12.1–2002.

119 **1.2 Purpose**

120 The purpose of this standard is to allow the creation of LOM instances in XML. This standard  
121 uses the W3C XML Schema definition language as the encoding. This allows for interoperability  
122 and the exchange of LOM instances between various systems.

123 **2. References**

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

127 IEEE 1484.12.1–2002, Standard for Learning Object Metadata.

128 IETF RFC 2048:1996, Multipurpose Internet Mail Extensions (MIME) Part Four: Regis-  
129 tration Procedures.

130 IETF RFC 2426:1998, vCard MIME Directory Profile.

131 ISO 639–1, Code for the representation of names of languages – Part 1: Alpha-2 code.

- 132 ISO 639–2, Codes for the representation of names of languages – Part 2: Alpha-3 code.
- 133 ISO 3166–1, Codes for the representation of names of countries and their subdivisions – Part  
134 1: Country codes.
- 135 ISO/IEC 10646-1, Information technology—Universal multiple-octet coded character set –  
136 Part 1: Architecture and basic multilingual plane.
- 137 XML Schema Part 1: Structures, W3C Recommendation, 2 May 2001.
- 138 XML Schema Part 2: Data types, W3C Recommendation, 2 May 2001.

### 139 **3. Definitions and acronyms**

140 Definitions and acronyms are defined in 3.1 and 3.2, respectively

#### 141 **3.1 Definitions**

142 For purposes of this standard, the following terms and definitions apply. IEEE 100, *The Au-*  
143 *thoritative Dictionary of IEEE Standards Terms*, Seventh Edition [A1]<sup>1</sup>, should be referenced  
144 for terms not defined in this Clause.

145 **aggregate element**: A data element that contains other data elements called subelements. *See*  
146 *also*: **subelement**.

147 **component XML Schema definition (component XSD)**: An Extensible Markup Language  
148 Schema definition that represents a constituent or part element of a composite schema. *See*  
149 *also*: **composite XML Schema definition**.

150 **composite XML Schema definition (composite XSD)**: An Extensible Markup Language  
151 Schema definition that is a structure made up of distinct component XML Schema definitions.  
152 *See also*: **component XML Schema definition**.

153 **content model**: A framework that identifies the makeup (i.e., data types, multiplicity con-  
154 straints, ordering) of a specific model.

155 **data type**: A property of distinct values, indicating common features of those values and op-  
156 erations on those values.

157 **Extensible Markup Language infoset (XML infoset)**: An abstract data set that provides a  
158 consistent set of definitions for use in other specifications that need to refer to the information  
159 in a well-formed XML document [A5].

---

<sup>1</sup> The numbers in brackets correspond to those of the bibliography in Annex A.

160 **Extensible Markup Language Schema binding (XML Schema binding):** A textual repre-  
161 sentation of the behaviors, attributes, and value space of a data-model element in W3C Exten-  
162 sible Markup Language Schema definition language to make the data-model element amena-  
163 ble to machine processing.

164 **LangString:** A data type that represents one or more character strings. A LangString value  
165 may include multiple semantically equivalent character strings, such as translations or alterna-  
166 tive descriptions. *See also:* **data type**.

167 **learning object:** In this standard, any entity, digital or non-digital, that may be used for learn-  
168 ing, education, or training.

169 **learning object metadata data element (LOM data element):** A data element for which the  
170 name, explanation, size, ordering, value space, and data type are defined in IEEE 1484.12.1–  
171 2002. *See also:* **LOMv1.0 base schema**.

172 **learning object metadata instance (LOM instance):** In this standard, a collection of meta-  
173 data for a learning object that conforms to IEEE 1484.12.1–2002, is represented in XML, and  
174 adheres to the requirements and constraints of the XML binding defined in this standard. *See*  
175 *also:* **LOMv1.0 base schema**.

176 **LOMv1.0 base schema:** A structured collection of standard data items, including their data  
177 types, multiplicities, and container/component relationships, described in IEEE 1484.12.1–  
178 2002, Clause 6.

179 **Multipurpose Internet Mail Extensions type (MIME type):** A standard way of classifying  
180 content types on the Internet.

181 **subelement:** A data element that is contained within another data element called an aggregate  
182 element. A subelement may contain other subelements, in which case it is also an aggregate  
183 element. *See also:* **aggregate element**.

184 **token:** Tokens are character strings. The [value space](#) of a token is the set of strings that do not  
185 contain the line feed (#xA) nor tab (#x9) characters, that have no leading or trailing spaces  
186 (#x20), and that have no internal sequences of two or more spaces. The [lexical space](#) of a to-  
187 ken is the set of strings that do not contain the line feed (#xA) nor tab (#x9) characters, that  
188 have no leading or trailing spaces (#x20), and that have no internal sequences of two or more  
189 spaces. The [base type](#) of a token is [normalizedString](#) (adapted from XML Schema Part 2: Data  
190 types). For purposes of this standard, tokens are case sensitive.

191 **token set:** A set of tokens in which each token is unique. *See also:* **token**.

192 **value space:** The set of values for a given data type (ISO/IEC 11404:1996 [A4]).

193 NOTE—In IEEE 1484.12.1–2002, a value space is typically enumerated outright or defined by  
194 reference to another standard or specification.

## 195 **3.2 Acronyms and abbreviations**

196	LOM: Learning Object Metadata
197	MIME: Multipurpose Internet Mail Extensions
198	SPM: smallest permitted maximum
199	UTC: coordinated universal time
200	W3C: World Wide Web Consortium
201	XML: Extensible Markup Language
202	XSD: XML Schema definition

## 203 **4. Conformance**

204 This standard provides definitions of W3C XML Schema definition language structures used  
205 to produce both *strictly conforming* and *conforming* LOM instances.

206 In this standard, “shall” is to be interpreted as a requirement on an implementation; “shall not”  
207 is to be interpreted as a prohibition.

### 208 **4.1 Strictly conforming LOM instances**

209 A *strictly conforming* LOM instance

- 210 – Shall conform to the LOMv1.0 base schema requirements of IEEE 1484.12.1–  
211 2002;
- 212 – Shall conform to the requirements of Clause 5; and
- 213 – Shall not contain any extensions to the LOMv1.0 base schema defined in IEEE  
214 1484.12.1–2002.

### 215 **4.2 Conforming LOM instances**

216 A *conforming* LOM instance

- 217 – Shall conform to the LOMv1.0 base schema requirements of IEEE 1484.12.1–  
218 2002; and
- 219 – Shall conform to the requirements of Clause 5.  
220

## 221 **5. LOM XML Schema binding definition**

222 The LOM XML Schema binding is defined in 5.1 – 5.5. Subclause 5.4 lists the data elements  
223 of the LOMv1.0 base schema. For each data element, the corresponding W3C XML Schema  
224 definition language element for XML binding is described.

## 225 **5.1 General information**

226 All LOMv1.0 base schema data elements are optional. An XML Schema definition (XSD) for  
227 the LOMv1.0 base schema shall not require any LOM data to be present in a LOM instance.

228 The LOMv1.0 base schema defines an aggregation relationship between data items. An XSD  
229 for LOM defines an aggregation relationship between subelements that maintains the relation-  
230 ship defined in the LOMv1.0 base schema. In a LOM instance, a subelement shall appear only  
231 within the parent element of the aggregation relationship. For example, in 5.4.3.1, the Identifi-  
232 fier subelement appears by definition as a component of the Meta-Metadata element described  
233 in 5.4.3. The presence of the subelement automatically implies the presence of the aggregate  
234 element to which the component belongs.

235 The LOMv1.0 base schema does not define any sequence of elements, except for their aggre-  
236 gation relationships. Therefore, an XML binding of the LOMv1.0 base schema shall not de-  
237 fine any sequence of the elements in a LOM instance, except for their aggregation relation-  
238 ships.

239 NOTE—W3C XML Schema definition language element names are derived from LOM data  
240 element names and are encoded in *lowerCamelCase*.

### 241 **5.1.1 Smallest permitted maximums**

242 The W3C XML Schema definition language does not support the concept of smallest permit-  
243 ted maximums (SPMs) as defined in IEEE 1484.12.1–2002. The W3C XML Schema defini-  
244 tion language does not provide a mechanism to specify the minimum number of elements in  
245 composite data types. The W3C XML Schema definition language does provide a means for  
246 restricting maximum lengths (i.e., `maxLength` for character strings) and maximum numbers  
247 of occurrences (i.e., `maxOccurs` for composite elements). However, these restrictions are not  
248 compatible with the definition of SPM.

249 NOTE—To encourage interoperability, creators of XSDs intended to validate LOM instances  
250 against IEEE 1484.12.1–2002 should not use `maxOccurs` or `maxLength` restrictions.

### 251 **5.1.2 Post-parsing validation**

252 The post-schema validation XML infoset of a LOM instance is insufficient to determine its  
253 validity with respect to IEEE 1484.12.1–2002. After a LOM instance is validated against the  
254 XSD, it shall be subjected to additional post-parsing validation rules. These additional rules  
255 are defined throughout 5.4 and 5.5.

### 256 **5.1.3 Extension support**

257 A *strictly conforming* XML binding of the LOMv1.0 base schema shall not support extensions  
258 to the LOM data elements.

259 A *conforming* XML binding of the LOMv1.0 base schema may include extensions to the  
260 LOM data elements. Extensions shall not conflict with the W3C XML Schema definition lan-  
261 guage names and namespaces defined in this standard.

## 262 5.2 LOM namespaces

263 The following namespaces are reserved by this standard and shall be used in defining the ele-  
264 ments defined in this standard:

- 265 – The namespace name for this standard shall be:  
266 `http://ltsc.ieee.org/xsd/LOM`. This namespace is reserved by this stan-  
267 dard, and shall not be used to represent any other namespace;
- 268 – The namespace name prefix for the `http://ltsc.ieee.org/xsd/LOM` name-  
269 space shall be `lom`. This value is reserved by this standard, and the prefix shall not be  
270 used to represent any other namespace;
- 271 – The namespace `http://ltsc.ieee.org/xsd/LOM/custom` shall be reserved  
272 for the LOM XML custom composite XSDs (see Annex B) provided by IEEE;
- 273 – The namespace `http://ltsc.ieee.org/xsd/LOM/unique` shall be reserved  
274 for the LOM XML unique composite XSDs (see Annex B) provided by IEEE;
- 275 – The namespace `http://ltsc.ieee.org/xsd/LOM/vocab` shall be reserved  
276 for the LOM XML vocab composite XSDs (see Annex B), provided by IEEE; and
- 277 – The namespace `http://ltsc.ieee.org/xsd/LOM/extend` shall be reserved  
278 for the LOM XML extension composite XSDs (see Annex B) provided by IEEE.

279 NOTE—If an organization provides extension elements to the LOM instance, the organization is  
280 free to define the namespace for these extensions.

## 281 5.3 Tabular format and table organization

282 This standard uses a tabular format to describe the requirements for the LOM instances. The  
283 tables express requirements for each of the LOM data elements in the LOMv1.0 base schema.  
284 These requirements are

- 285 – *LOM name*: The name of the LOM data element in the LOMv1.0 base schema  
286 element.
- 287 – *XML name*: The XML binding of the LOM data element.
- 288 – *Subelements*: A listing of subelements of the LOM data element. An entry of  
289 "None" indicates that the element does not have subelements. If present, the  
290 subelements listed shall be contained by their associated aggregate element. This  
291 ensures that the aggregation relationships of the LOMv1.0 base schema compo-  
292 nents are enforced. The order of appearance of subelements shall not be signifi-  
293 cant.
- 294 – *Min*: The requirements on the minimum number of times the subelement may ap-  
295 pear in a LOM instance, in the context of the subelement's aggregate element.
- 296 – *Max*: The requirements on the maximum number of times the subelement may  
297 appear in a LOM instance, in the context of the subelement's aggregate element.  
298 An "n" indicates that the maximum number of times is unbounded.

- 299 – *Order*: Indicates whether the order of the values is significant. This standard uses  
 300 three designators for the order: "Ordered", "Unordered", and "Unspecified".  
 301 – *Data type*: Indicates whether the values are LangString, DateTime, Duration, Vo-  
 302 cabulary, CharacterString, or Unspecified (Unspecified indicates that the data  
 303 element has no data type). If the data type is Vocabulary, the permissible values  
 304 from the LOMv1.0 base schema are listed.

305 Each table that describes an aggregate element includes the first-generation subelements of  
 306 that aggregate element. For example, Table 1 in 5.4 describes the LOM element and includes  
 307 the element's first-generation subelements, such as General and Life Cycle. However, the table  
 308 does not include the subelements of the General element, such as Identifier and Title. If a  
 309 subelement is also an aggregate element, a separate table describes the subelement with its  
 310 first-generation subelements.

311 With the exception of the order for Intended End User Role, in the case of any discrepancy in  
 312 the tables in 5.4 and 5.5 and IEEE 1484.12.1–2002, the values from IEEE 1484.12.1–2002  
 313 shall be used.

## 314 5.4 LOM

315 Table 1 describes the LOM element and its first-generation subelements.

316 **Table 1—The LOM element**

LOM name	XML name	Subelements	Min	Max	Order	Data type
LOM	<lom>	General Life Cycle Meta-Metadata Technical Educational Rights Relation Annotation Classification	1	1	Unspecified	Unspecified
General	<general>	Identifier Title Language Description Keyword Coverage Structure Aggregation Level	0	1	Unspecified	Unspecified
Life Cycle	<lifecycle>	Version Status Contribute	0	1	Unspecified	Unspecified

LOM name	XML name	Subelements	Min	Max	Order	Data type
Meta-Metadata	<metaMetadata>	Identifier Contribute Metadata Schema Language	0	1	Unspecified	Unspecified
Technical	<technical>	Format Size Location Requirement Installation Remarks Other Platform Requirements Duration	0	1	Unspecified	Unspecified
Educational	<educational>	Interactivity Type Learning Resource Type Interactivity Level Semantic Density Intended End User Role Context Typical Age Range Difficulty Typical Learning Time Description Language	0	n	Unspecified	Unspecified
Rights	<rights>	Cost Copyright and Other Restrictions Description	0	1	Unspecified	Unspecified
Relation	<relation>	Kind Resource	0	n	Unordered	Unspecified
Annotation	<annotation>	Entity Date Description	0	n	Unordered	Unspecified
Classification	<classification>	Purpose Taxon Path Description Keyword	0	n	Unordered	Unspecified

### 317 Namespace declaration

318 The LOM instance shall include a namespace declaration that declares the LOM namespace  
319 for the LOM element and its components. The LOM namespace shall be  
320 "http://ltsc.ieee.org/xsd/LOM" as defined in 5.2. The namespace declaration used  
321 in an XML instance can be defined in the root element of that XML instance, which may be  
322 an element other than the LOM element, or by declaring the LOMv1.0 base schema name-  
323 space within the LOM element each time that element appears in an XML instance.

324 The namespace declaration shall take the form of a default namespace declaration or a prefix-  
 325 specific namespace declaration in conformance with W3C XML Recommendations. Exam-  
 326 ples are shown in Figures 1 and 2, respectively:

```
327
328 <lom xmlns="http://ltsc.ieee.org/xsd/LOM">
329 . . .
330 </lom>
331
```

332 **Figure 1—An example default namespace declaration**

```
333
334 <lom:lom xmlns:lom="http://ltsc.ieee.org/xsd/LOM">
335 . . .
336 </lom:lom>
337
```

338 **Figure 2—An example prefix-specific namespace declaration**

339 NOTE—A namespace declaration is required so that applications will recognize LOM data as a  
 340 LOM instance that conforms to this standard.

### 341 5.4.1 General

342 Table 2 describes the General element and its first generation subelements.

343 **Table 2—The General element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
General	<general>	Identifier Title Language Description Keyword Coverage Structure Aggregation Level	0	1	Unspecified	Unspecified
Identifier	<identifier>	Catalog Entry	0	n	Unspecified	Unspecified
Title	<title>	String	0	1	Unspecified	LangString
Language	<language>	Unspecified	0	n	Unordered	CharacterString
Description	<description>	String	0	n	Unordered	LangString
Keyword	<keyword>	String	0	n	Unordered	LangString
Coverage	<coverage>	String	0	n	Unordered	LangString
Structure	<structure>	Source Value	0	1	Unspecified	Vocabulary

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Aggregation Level	<aggregationLevel>	Source Value	0	1	Unspecified	Vocabulary

344 \*Common data types and data elements are defined in 5.5.

### 345 5.4.1.1 Identifier

346 Table 3 describes the Identifier element and its first-generation subelements.

347 **Table 3—The Identifier element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Identifier	<identifier>	Catalog Entry	0	n	Unspecified	Unspecified
Catalog	<catalog>>	None	0	1	Unspecified	CharacterString
Entry	<entry>	None	0	1	Unspecified	CharacterString

348 \*Common data types and data elements are defined in 5.5.

### 349 5.4.1.2 Language

350 The Language element shall be a character string where the value of the character string shall  
351 be one the following:

- 352 – the format and values described in 5.5.4.1; or
- 353 – the token none.

### 354 5.4.1.3 Structure

355 If the source of the value space for the Structure element is the LOMv1.0 base schema (i.e.,  
356 <source>LOMv1.0<\source>), then the valid value for the element shall be one of the fol-  
357 lowing tokens:

- 358 – atomic
- 359 – collection
- 360 – networked
- 361 – hierarchical
- 362 – linear

### 363 5.4.1.4 Aggregation Level

364 If the source of the value space for the Aggregation Level element is the LOMv1.0 base  
365 schema (i.e., <source>LOMv1.0<\source>), then the valid value for the element shall be  
366 one of the following tokens:

- 367 – 1

- 368 – 2  
 369 – 3  
 370 – 4

## 371 5.4.2 Life Cycle

372 Table 4 describes the Life Cycle element and its first-generation subelements.

373 **Table 4—The Life Cycle element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Life Cycle	<lifecycle>	Version Status Contribute	0	1	Unspecified	Unspecified
Version	<version>	String	0	1	Unspecified	LangString
Status	<status>	Source Value	0	1	Unspecified	Vocabulary
Contribute	<contribute>	Role Entity Date	0	n	Ordered	Unspecified

374 \*Common data types and data elements are defined in 5.5.

### 375 5.4.2.1 Status

376 If the source of the value space for the Status element is the LOMv1.0 base schema (i.e.,  
 377 <source>LOMv1.0<\source>), then the valid value for the element shall be one of the fol-  
 378 lowing tokens:

- 379 – draft  
 380 – final  
 381 – revised  
 382 – unavailable

### 383 5.4.2.2 Contribute

384 Table 5 describes the Contribute element and its first-generation subelements.

385 **Table 5—The Contribute element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Contribute	<contribute>	Role Entity Date	0	n	Ordered	Unspecified
Role	<role>	Source Value	0	1	Unspecified	Vocabulary
Entity	<entity>	None	0	n	Ordered	vCard

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Date	<date>	DateTime Description	0	1	Unspecified	DateTime

386 \*Common data types and data elements are defined in 5.5.

#### 387 5.4.2.2.1 Role

388 If the source of the value space for the Role element is the LOMv1.0 base schema (i.e.,  
389 <source>LOMv1.0<\source>), then the valid value for the element shall be one of the fol-  
390 lowing tokens:

- 391 - author
- 392 - publisher
- 393 - unknown
- 394 - initiator
- 395 - terminator
- 396 - validator
- 397 - editor
- 398 - graphical designer
- 399 - technical implementer
- 400 - content provider
- 401 - technical validator
- 402 - educational validator
- 403 - script writer
- 404 - instructional designer
- 405 - subject matter expert

#### 406 5.4.3 Meta-Metadata

407 Table 6 describes the Meta-Metadata element and its first-generation subelements.

408 **Table 6—The Meta-Metadata element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Meta-Metadata	<metaMetadata>	Identifier Contribute Metadata Schema Language	0	1	Unspecified	Unspecified
Identifier	<identifier>	Catalog Entry	0	n	Unspecified	Unspecified
Contribute	<contribute>	Role Entity Date	0	n	Ordered	Unspecified
Metadata Schema	<metadataSchema>	None	0	n	Unordered	CharacterString
Language	<language>	None	0	1	Unspecified	CharacterString

409 \*Common data types and data elements are defined in 5.5.

#### 410 5.4.3.1 Identifier

411 Table 7 describes the Identifier element and its first-generation subelements.

412 **Table 7—The Identifier element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Identifier	<identifier>	Catalog Entry	0	n	Unspecified	Unspecified
Catalog	<catalog>>	None	0	1	Unspecified	CharacterString
Entry	<entry>	None	0	1	Unspecified	vCard

413 \*Common data types and data elements are defined in 5.5.

#### 414 5.4.3.2 Contribute

415 Table 8 describes the Contribute element and its first-generation subelements.

416 **Table 8—The Contribute element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Contribute	<contribute>	Role Entity Date	0	n	Ordered	Unspecified
Role	<role>	Source Value	0	1	Unspecified	Vocabulary
Entity	<entity>	None	0	N	Ordered	vCard
Date	<date>	DateTime Description	0	1	Unspecified	DateTime

417 \*Common data types and data elements are defined in 5.5.

##### 418 5.4.3.2.1 Role

419 If the source of the value space for the Role element is the LOMv1.0 base schema (i.e.,  
420 <source>LOMv1.0<\source>), then the valid value for the element shall be one of the fol-  
421 lowing tokens:

- 422 - creator
- 423 - validator

424 **5.4.3.3 Metadata Schema**

425 For those LOM instances that are *strictly conforming* to this standard (see 4), if the Metadata  
 426 Schema element (<metadataSchema>) is present in a LOM instance, the element shall con-  
 427 tain the value LOMv1.0.

428 For those LOM XML Instances that are *conforming* (i.e., contain extended data model ele-  
 429 ments, see 4), if the Metadata Schema element (<metadataSchema>) is present in a LOM  
 430 instance, the element shall contain the value LOMv1.0, and additional occurrences of  
 431 <metadataSchema> shall list all other metadata schemas or authoritative specifications used  
 432 to create the LOM instance.

433 **5.4.3.4 Language**

434 The Language element shall be a character string where the value of the character string shall  
 435 be have the format and values described in 5.5.4.1.

436 **5.4.4 Technical**

437 Table 9 describes the Technical element and its first-generation subelements.

438 **Table 9—The Technical element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Technical	<technical>	Format Size Location Requirement Installation Remarks Other Platform Requirements Duration	0	1	Unspecified	Unspecified
Format	<format>	None	0	n	Unordered	CharacterString
Size	<size>	None	0	1	Unspecified	CharacterString
Location	<location>	None	0	n	Ordered	CharacterString
Requirement	<requirement>	OrComposite	0	n	Unordered	Unspecified
Installation Remarks	<installationRemarks>	None	0	1	Unspecified	LangString
Other Platform Requirements	<otherPlatformRequirements>	None	0	1	Unspecified	LangString
Duration	<duration>	None	0	1	Unspecified	Duration

439 \*Common data types and data elements are defined in 5.5.

440 **5.4.4.1 Format**

441 The canonical lexical representation of the Multipurpose Internet Mail Extension (MIME)  
442 type values from RFC 2048 shall be a character string literal or the token `non-digital`.

443 **5.4.4.2 Size**

444 The value held by the Size element shall be a combination of the digits "0" . . "9".

445 **5.4.4.3 Requirement**

446 Table 10 describes the Requirement element and its first-generation subelement.

447 **Table 10—The Requirement element**

LOM name	XML name	Subelements	Min	Max	Order	Data type
Requirement	<requirement>	OrComposte	0	n	Unordered	Unspecified
OrComposite	<orComposite>	Type Name Minimum Version Maximum Version	0	n	Unordered	Unspecified

448 **5.4.4.3.1 OrComposite**

449 Table 11 describes the OrComposite element and its first-generation subelements.

450 **Table 11—The OrComposite element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
OrComposite	<orComposite>	Type Name Minimum Version Maximum Version	0	n	Unordered	Unspecified
Type	<type>	Source Value	0	1	Unspecified	Vocabulary
Name	<name>	Source Value	0	1	Unspecified	Vocabulary
Minimum Version	<minimumVersion>	None	0	1	Unspecified	CharacterString
Maximum Version	<maximumVersion>	None	0	1	Unspecified	CharacterString

451 \*Common data types and data elements are defined in 5.5.

**452 Type**

453 The Type and Name elements shall be used as a pair. If one element is used in a LOM in-  
454 stance, the other shall be used also.

455 If the source of the value space for the Type element is the LOMv1.0 base schema (i.e.,  
456 <source>LOMv1.0<\source>), then the valid value for the element shall be one of the fol-  
457 lowing tokens:

- 458 - operating system
- 459 - browser

**460 Name**

461 The Type and Name elements shall be used as a pair. If one element is used in a LOM in-  
462 stance, the other shall be used also.

463 If the source of the value space for the Name element is the LOMv1.0 base schema (i.e.,  
464 <source>LOMv1.0<\source>), and the Type element's value is the token operating  
465 system, then the valid value for the Name element shall be one of the following tokens:

- 466 - pc-dos
- 467 - ms-windows
- 468 - macos
- 469 - unix
- 470 - multi-os
- 471 - none

472 If the source of the value space for the Name element is the LOMv1.0 base schema (i.e.,  
473 <source>LOMv1.0<\source>), and the Type element's value is the token browser, then  
474 the valid value for the Name element shall be one of the following tokens:

- 475 - any
- 476 - netscape communicator
- 477 - ms-internet explorer
- 478 - opera
- 479 - amaya

**480 5.4.5 Educational**

481 Table 12 describes the Educational element and its first-generation subelements.

482

**Table 12—The Educational element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Educational	<educational>	Interactivity Type Learning Resource Type Interactivity Level Semantic Density Intended End User Role Context Typical Age Range Difficulty Typical Learning Time Description Language	0	n	Unspecified	Unspecified
Interactivity Type	<interactivityType>	Source Value	0	1	Unspecified	Vocabulary
Learning Resource Type	<learningResourceType>	Source Value	0	n	Ordered	Vocabulary
Interactivity Level	<interactivityLevel>	Source Value	0	1	Unspecified	Vocabulary
Semantic Density	<semanticDensity>	Source Value	0	1	Unspecified	Vocabulary
Intended End User Role	<intendedEndUserRole>	Source Value	0	n	Ordered	Vocabulary
Context	<context>	Source Value	0	n	Unordered	Vocabulary
Typical Age Range	<typicalAgeRange>	String	0	n	Unspecified	LangString
Difficulty	<difficulty>	Source Value	0	1	Unspecified	Vocabulary
Typical Learning Time	<typicalLearningTime>	Duration Description	0	1	Unspecified	Duration
Description	<description>	String	0	n	Unspecified	LangString
Language	<language>	None	0	n	Unordered	CharacterString

483 \* Common data types and data elements are defined in 5.5.

#### 484 5.4.5.1 Interactivity Type

485 If the source of the value space for the Interactivity Type element is the LOMv1.0 base  
 486 schema (i.e., <source>LOMv1.0<\source>), then the valid value for the element shall be  
 487 one of the following tokens:

- 488 - active
- 489 - expositive

490       – mixed

#### 491   **5.4.5.2 Learning Resource Type**

492   If the source of the value space for the Learning Resource Type element is the LOMv1.0 base  
493   schema (i.e., <source>LOMv1.0<\source>), then the value for the element shall be one of  
494   the following tokens:

- 495       – exercise
- 496       – simulation
- 497       – questionnaire
- 498       – diagram
- 499       – figure
- 500       – graph
- 501       – index
- 502       – slide
- 503       – table
- 504       – narrative text
- 505       – exam
- 506       – experiment
- 507       – problem statement
- 508       – self assessment
- 509       – lecture

#### 510   **5.4.5.3 Interactivity Level**

511   If the source of the value space for the Interactivity Level element is the LOMv1.0 base  
512   schema (i.e., <source>LOMv1.0<\source>), then the valid value for the element shall be  
513   one of the following tokens:

- 514       – very low
- 515       – low
- 516       – medium
- 517       – high
- 518       – very high

#### 519   **5.4.5.4 Semantic Density**

520   If the source of the value space for the Semantic Density element is the LOMv1.0 base  
521   schema (i.e., <source>LOMv1.0<\source>), then the valid value for the element shall be  
522   one of the following tokens:

- 523       – very low
- 524       – low
- 525       – medium
- 526       – high
- 527       – very high

#### 528 **5.4.5.5 Intended End User Role**

529 If the source of the value space for the Intended End User Role element is the LOMv1.0 base  
530 schema (i.e., `<source>LOMv1.0<\source>`), then the value for the element shall be one of  
531 the following tokens:

- 532 - teacher
- 533 - author
- 534 - learner
- 535 - manager

#### 536 **5.4.5.6 Context**

537 If the source of the value space for the Context element is the LOMv1.0 base schema (i.e.,  
538 `<source>LOMv1.0<\source>`), then the value for the element shall be one of the follow-  
539 ing tokens:

- 540 - school
- 541 - higher education
- 542 - training
- 543 - other

#### 544 **5.4.5.7 Difficulty**

545 If the source of the value space for the Difficulty element is the LOMv1.0 base schema (i.e.,  
546 `<source>LOMv1.0<\source>`), then the valid value for the element shall be one of the fol-  
547 lowing tokens:

- 548 - very easy
- 549 - easy
- 550 - medium
- 551 - difficult
- 552 - very difficult

#### 553 **5.4.5.8 Language**

554 The Language element shall be a character string where the value of the character string shall  
555 be have the format and values described in 5.5.4.1.

#### 556 **5.4.6 Rights**

557 Table 13 describes the Rights element and its first-generation subelements.

558

**Table 13—The Rights element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Rights	<rights>	Cost Copyright and Other Restrictions Description	0	1	Unspecified	Unspecified
Cost	<cost>	Source Value	0	1	Unspecified	Vocabulary
Copyright And Other Restrictions	<copyrightAndOtherRestrictions>	Source Value	0	1	Unspecified	Vocabulary
Description	<description>	String	0	1	Unspecified	LangString

559 \*Common data types and data elements are defined in 5.5.

#### 560 5.4.6.1 Cost

561 If the source of the value space for the Cost element is the LOMv1.0 base schema (i.e.,  
562 <source>LOMv1.0<\source>), then the valid value for the element shall be one of the fol-  
563 lowing tokens:

- 564     – yes  
565     – no

#### 566 5.4.6.2 Copyright And Other Restrictions

567 If the source of the value space for the Copy Right And Other Restrictions element is the  
568 LOMv1.0 base schema (i.e., <source>LOMv1.0<\source>), then the valid value for the  
569 element shall be one of the following tokens:

- 570     – yes  
571     – no

#### 572 5.4.7 Relation

573 Table 14 describes the Relation element and its first-generation subelements.

574

**Table 14—The Relation element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Relation	<relation>	Kind Resource	0	n	Unordered	Unspecified
Kind	<kind>	Source Value	0	1	Unspecified	Vocabulary
Resource	<resource>	Identifier Description	0	1	Unspecified	Unspecified

575 \*Common data types and data elements are defined in 5.5.

576 **5.4.7.1 Kind**

577 If the source of the value space for the Kind element is the LOMv1.0 base schema (i.e.,  
 578 <source>LOMv1.0<\source>), then the valid value for the element shall be one of the fol-  
 579 lowing tokens:

- 580 - ispartof
- 581 - haspart
- 582 - isversionof
- 583 - hasversion
- 584 - isformatof
- 585 - hasformat
- 586 - references
- 587 - isreferencedby
- 588 - isbasedon
- 589 - isbasisfor
- 590 - requires
- 591 - isrequiredby

592 **5.4.7.2 Resource**

593 Table 15 describes the Resource element and its first-generation subelements.

594 **Table 15—The Resource element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Resource	<resource>	Identifier Description	0	1	Unspecified	Unspecified
Identifier	<identifier>	Catalog Entry	0	n	Unspecified	Unspecified
Description	<description>	String	0	n	Unspecified	LangString

595 \*Common data types and data elements are defined in 5.5.

596 **5.4.7.2.1 Identifier**

597 Table 16 describes the Identifier element and its first-generation subelements.

598 **Table 16—The Identifier element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Identifier	<identifier>	Catalog Entry	0	n	Unspecified	Unspecified
Catalog	<catalog>	None	0	1	Unspecified	CharacterString
Entry	<entry>	None	0	1	Unspecified	CharacterString

599 \*Common data types and data elements are defined in 5.5.

## 600 5.4.8 Annotation

601 Table 17 describes the Annotation element and its first-generation subelements.

602 **Table 17—The Annotation element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Annotation	<annotation>	Entity Date Description	0	n	Unordered	Unspecified
Entity	<entity>	None	0	1	Unspecified	vCard
Date	<date>	DateTime Description	0	1	Unspecified	DateTime
Description	<description>	String	0	1	Unspecified	LangString

603 \*Common data types and data elements are defined in 5.5.

## 604 5.4.9 Classification

605 Table 18 describes the Classification element and its first-generation subelements.

606 **Table 18—The Classification element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Classification	<classification>	Purpose Taxon Path Description Keyword	0	n	Unordered	Unspecified
Purpose	<purpose>	None	0	1	Unspecified	Vocabulary
Taxon Path	<taxonPath>	Source Taxon	0	n	Unordered	Undefined
Description	<description>	String	0	1	Unspecified	LangString
Keyword	<keyword>	String	0	n	Ordered	LangString

607 \*Common data types and data elements are defined in 5.5.

### 608 5.4.9.1 Purpose

609 If the source of the value space for the Purpose element is the LOMv1.0 base schema (i.e.,  
610 <source>LOMv1.0<\source>), then the valid value for the element shall be one of the fol-  
611 lowing tokens:

- 612 - discipline
- 613 - idea
- 614 - prerequisite
- 615 - educational objective
- 616 - accessibility restrictions

- 617     – educational level
- 618     – skill level
- 619     – security level
- 620     – competency

## 621     **5.4.9.2 Taxon Path**

622     Table 19 describes the Taxon Path element and its first-generation subelements.

623                     **Table 19—The Taxon Path element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Taxon Path	<taxonPath>	Source Taxon	0	n	Ordered	Unspecified
Source	<source>	String	0	1	Unspecified	LangString
Taxon	<taxon>	Id Entry	0	n	Ordered	Unspecified

624     \*Common data types and data elements are defined in 5.5.

## 625     **5.4.9.2.1 Taxon**

626     Table 20 describes the Taxon element and its first-generation subelements.

627                     **Table 20—The Taxon element**

LOM name	XML name	Subelements	Min	Max	Order	Data type*
Taxon	<taxon>	Id Entry	0	n	Ordered	Unspecified
Id	<id>	None	0	1	Unspecified	CharacterString
Entry	<entry>	String	0	1	Unspecified	LangString

628     \*Common data types and data elements are defined in 5.5.

## 629     **5.5 Common data types and elements**

630     Data types that are used in the LOM XML binding are defined in 5.5.1 – 5.5.7.

### 631     **5.5.1 CharacterString**

632     For those elements defined in this standard that have a data type of `CharacterString`, the  
 633     W3C XML Schema definition language binding shall be an `xs:string`. The `xs:string`  
 634     data type supports the repertoire of ISO/IEC 10646-1. The `xs:string` data type ensures the  
 635     support of multiple languages including multibyte languages.

636     NOTE—The `<xs:string>` construct is defined in "XML Schema Part 2."

## 637 5.5.2 DateTime

638 The DateTime data type is represented as an XML element. The DateTime data type may  
639 contain the following subelements:

- 640 – Date Time
- 641 – Description

642 If present, the subelements listed above shall be contained by the element for which the  
643 DateTime type is defined. The order of appearance of subelements shall not be significant.

### 644 5.5.2.1 Date Time

645 Table 21 describes the Date Time element and its first-generation subelements.

646 **Table 21—The DateTime element**

LOM name	XML name	Subelements	Min	Max	Order	Data type
Date	<date>	DateTime Description	0	1	Unspecified	DateTime
DateTime	<dateTime>	None	0	1	Unspecified	CharacterString
Description	<description>	String	0	1	Unspecified	LangString

#### 647 5.5.2.1.1 DateTime

648 The value space for the Date Time element is a pattern which shall be defined according to the  
649 following rules.

650 YYYY[-MM[-DD[Thh[:mm[:ss[.s[TZD]]]]]]] where:

- 651 – YYYY is the four-digit year (>=0001)
- 652 – MM is the two-digit month (01 through 12 where 01=January, etc.)
- 653 – DD is the two-digit day of month (01 through 31, depending on the values of  
654 month and year)
- 655 – hh is two digits of hour (00 through 23) (am/pm NOT allowed)
- 656 – mm is two digits of minute (00 through 59)
- 657 – ss is two digits of second (00 through 59)
- 658 – s is one or more digits representing a decimal fraction of a second
- 659 – TZD is the time zone designator ("Z" for coordinated universal time [UTC] or  
660 +hh:mm or -hh:mm)

661 At least the four-digit year shall be present. If additional parts of the date and time are in-  
662 cluded, the character literals "-", "T", ":", and "." are part of the character lexical representation  
663 for the datetime.

664 If the time portion is present, but the time zone designator is not present, the time zone is in-  
665 terpreted as being UTC.

666 NOTES:

667 1—Because of restrictions placed on the value space, the W3C XML Schema definition language  
668 type `xs:dateTime` could not be used in the XSD files provided with this standard (see Annex  
669 B). Therefore, a regular expression was created to capture all valid date/times supported by IEEE  
670 1484.12.1–2002.

671 2—The date portion represents dates in the Common Era (CE) only. The date portion follows the  
672 Gregorian calendar for dates after October 15, 1582, and the Julian calendar for dates prior to Oc-  
673 tober 15, 1582, independent of locale. Dates Before Common Era (BCE) and other cases should  
674 be represented using the Description element (see 5.5.2.1.2).

675 3—The square bracket meta characters ("[" and "]") indicate optional elements that may appear  
676 zero or one time in the character lexical representation of the DateTime. These meta characters do  
677 not appear in the result; only the associated values described appear (e.g., "DD" is replaced by the  
678 corresponding 2 digit value for day of month).

679 4—This value space is based on ISO 8601:2000 [A3].

### 680 5.5.2.1.2 Description

681 Subclause 5.5.4 defines the LangString data type.

## 682 5.5.3 Duration

683 The Duration data type is represented as an XML element. Table 22 describes the Duration  
684 element and its first-generation subelements.

685 **Table 22—The Duration element**

LOM name	XML name	Subelements	Min	Max	Order	Data type
Duration	unspecified	Duration Description	0	1	Unspecified	Unspecified
Duration	<duration>	None	0	1	Unspecified	CharacterString
Description	<description>	String	0	1	Unspecified	LangString

### 686 5.5.3.1 Duration

687 The value space for the Duration element is a pattern which shall be defined according to the  
688 following rules:

689  $P[yY][mM][dD][T[hH][nM][s[.s]S]]$  where:

- 690 – y is the number of years (integer, > 0, not restricted)
- 691 – m is the number of months (integer, > 0, not restricted, e.g., > 12 is acceptable)
- 692 – d is the number of days (integer, > 0, not restricted, e.g., > 31 is acceptable)
- 693 – h is the number of hours (integer, > 0, not restricted, e.g., > 23 is acceptable)
- 694 – n is the number of minutes (integer, > 0, not restricted, e.g., > 59 is acceptable)

695 – s is the number of seconds or fraction of seconds (integer, > 0, not restricted, e.g.,  
696 > 59 is acceptable)

697 The character literal designators "P", "Y", "M", "D", "T", "H", "M", "S" shall appear if the  
698 corresponding nonzero value is present.

699 If the value of years, months, days, hours, minutes or seconds is zero, the value and corre-  
700 sponding designation (e.g., "M") may be omitted, but at least one designator and value must  
701 always be present. The designator "P" is always present. The designator "T" shall be omitted if  
702 all of the time (hours/minutes/seconds) are zero.

703 Negative durations are not supported.

704 NOTES:

705 1—Because of the restrictions placed on the value space, the W3C XML Schema definition lan-  
706 guage type `xs:dateTime` could not be used in the XSD files provided with this standard (see  
707 Annex B). A regular expression was built to capture all valid date/times supported by IEEE  
708 1484.12.1–2002.

709 2—The value is designated in the Gregorian calendar.

710 3—The ordering of durations may be indeterminate (e.g., 1 month may be 28, 29, 30, or 31 days).

711 4—For durations that apply only while the learning object is in use, but not when its use is sus-  
712 pended, it is recommended that only hours and smaller units of duration be used. *Examples:*  
713 PT43H, PT5M35S. For durations that express a time span, regardless of whether the learning ob-  
714 ject is actually used continuously during that time, days and larger units of duration may be used.  
715 *Examples:* P1Y6M, P20D.

716 5—The square bracket meta characters ("[" and "]") indicate optional elements that may appear  
717 zero or one time in the character lexical representation of the Duration. These meta characters do  
718 not appear in the result; only the associated values described appear (e.g., "dD" is replaced by the  
719 corresponding value for the number of days in the duration and is followed by the character literal  
720 designator "D").

721 6—This value space is based on ISO 8601:2000 [A3].

### 722 **5.5.3.2 Description**

723 The `Description` data type is represented as a `LangString` data type (see 5.5.4).

### 724 **5.5.4 LangString**

725 The `LangString` data type is represented as an XML element. Table 23 describes the  
726 `LangString` element and its first-generation subelements.

727

**Table 23—The LangString element**

LOM name	XML name	Subelements	Min	Max	Order	Data type
LangString	Unspecified	String	0	n	Unordered	Unspecified
String	<string>	None	0	1	Unspecified	CharacterString
Language	language*	None	0	1	Unspecified	CharacterString

728 \*language is an XML attribute, not an XML element.

729 The Language element shall be a character string consisting of a required language code fol-  
 730 lowed by multiple, optional, hyphen-prefixed subcodes (see examples below). The Language  
 731 element is defined in 5.5.4.1.

### 732 5.5.4.1 Language

733 The Language element shall be a valid LanguageID where

734 LanguageID = Langcode ("-"Subcode)\*

735 The following rules apply to the Langcode part of the character string:

- 736 – Two-letter codes are defined in ISO 639–1.
- 737 – Three-letter codes are defined in ISO 639–2.
- 738 – The value "i" is reserved for registrations defined by the Internet Assigned Num-  
 739 bers Authority (IANA).
- 740 – The value "x" is reserved for private use.

741 The following rules apply to the Subcode part of the character string:

- 742 – Two-letter subcodes are ISO 31661 alpha-2 country codes.
- 743 – Subcodes of from 3 to 8 letters are registered with IANA.

744 Rules for additional subcodes are unspecified.

745 NOTE—The Langcode is normally given in lower case and the subcodes (if any) in upper case.  
 746 However, the values are case insensitive.

### 747 Examples

748 "en-GB"  
 749 "de"  
 750 "fr-CA"  
 751 "it"  
 752 "en-US-philadelphia"

### 753 5.5.5 Token

754 For those elements defined in this standard that have a data type of Token, the W3C XML  
 755 Schema definition language binding shall be an xs:token. The xs:token data type sup-

756 ports the repertoire of ISO/IEC 10646-1. The `xs:token` data type ensures the support of mul-  
757 tiple languages including multibyte languages.

758 NOTE—The `<xs:token>` construct is defined in "XML Schema Part 2."

### 759 5.5.6 vCard

760 The `vCard` data type is represented as a `CharacterString` data type. The canonical lexical  
761 representation of the `vCard` value shall be a valid `vCard[]` as defined in IETF RFC 2426:1998.

762 NOTE—IETF RFC 2426:1998 does not rely on XML syntax to express the internal structure of a  
763 valid `vCard[]`. At the time of publication of this standard, no *de facto* standard W3C XML  
764 Schema definition language binding for the `vCard` specification existed.

### 765 5.5.7 Vocabulary

766 The `vocabulary` data type is represented as an XML element. Table 24 describes the Vo-  
767 cabulary element and its first-generation subelements.

768

**Table 23—The Vocabulary element**

LOM name	XML name	Subelements	Min	Max	Order	Data type
Vocabulary	Unspecified	Source Value	Unspecified	Unspecified	Unspecified	Unspecified
Source	<code>&lt;source&gt;</code>	None	0	1	Unspecified	<code>CharacterString</code>
Value	<code>&lt;value&gt;</code>	None	0	1	Unspecified	<code>Token</code>

769 **Annex A**

770 (informative)

771 **Bibliography**

772 [A1] IEEE 100™, *The Authoritative Dictionary of IEEE Standards Terms*, Seventh Edi-  
773 tion.

774 [A2] IETF RFC 2425:1998, MIME Content-Type for Directory Information.

775 [A3] ISO 8601:2000, Data elements and interchange formats – Information interchange –  
776 Representation of dates and times.

777 [A4] ISO/IEC 11404:1996, Information Technology—Programming Languages, their Envi-  
778 ronments and System Software Interfaces—Language-Independent Datatypes.

779 [A5] W3C Recommendation 24 October 2001, XML Information Set.

780 **Annex B**

781 (informative)

782 **Internet availability of XSD files**

783 This standard includes XSD files that are available on the World Wide Web but are not in-  
784 cluded in the printed version of this standard. These XSD files are available for direct inclu-  
785 sion in applications and as examples for those who prefer to develop their own XSD files.

786 The XSD files are available at

787 – <http://ltsc.ieee.org/xsd/lomv1.0/>

## 788 **Annex C**

789 **(informative)**

### 790 **XSD file descriptions**

791 The LOMv1.0 base schema describes a structured collection of standard data items, including  
792 their data types, multiplicities, and container/component relationships, but does not provide a  
793 concrete syntax for encoding LOM data that conforms to the LOMv1.0 base schema.

794 The LOM XML binding described in this standard defines the XML syntax for encoding  
795 LOM data. The binding is a collection of rules describing how to express LOM data in XML  
796 syntax. These binding rules are defined in detail in Clause 5 of this standard.

797 When building XSDs, implementers can decide where to enforce LOMv1.0 base schema con-  
798 straints. These constraints can be defined in an XSD with the intent that schema processors  
799 will perform validation of the constraints defined in the XSD, or outside the XSD validation in  
800 some implementation-defined, post-schema processing. Clause 5 defines the criteria to create  
801 valid LOM instances; XSD implementers must decide how to develop XSD schema files ac-  
802 cording to these constraints. Regardless of the implementation, a LOM instance must conform  
803 to the rules defined in Clause 5.

804 This standard includes informative W3C XML XSDs for implementers to satisfy many of the  
805 requirements of IEEE 1484.12.1–2002 and the practices of its existing adopters (see Annex  
806 B). The XSDs are intended to be used by XML processing tools that support the W3C XML  
807 Schema definition language (see XML Schema, Parts 1 and 2). The LOMv1.0 base schema  
808 places requirements on a LOM instance some of which cannot be expressed in W3C XML  
809 Schema definition language. These requirements are defined in Clause 5.

810 A predefined, composite XSD (see C.1.1) is provided that can be tailored in multiple ways for  
811 particular preferences about LOM element ordering, checking vocabulary values, and using  
812 extensions. Three additional predefined composite XSDs are provided that enforce different  
813 validation rules (see C.1.2 – C.1.4). The four composite XSDs draw from several sets of com-  
814 ponent XSDs (see C.2).

815 The goal is to provide alternative XSDs, each of which supports the LOM XML binding by  
816 making certain choices about which binding rules to describe in the XSDs. Rules defined in an  
817 XSD should be enforced by tools that support XML Schema validation. Rules not expressed  
818 in an XSD are enforced by other means, such as checking of the post-schema validation XML  
819 info set. No matter where the rule enforcement is done, a *conforming* LOM instance must con-  
820 form to all rules defined in Clause 5.

821 The need for alternative XSDs was motivated by the variety of ways in which application de-  
822 velopers need to use the XSDs. The approach of providing alternative XSDs addresses the

823 needs of the various adopting communities to produce *conforming* LOM instances consistent  
824 with their processing and usage requirements.

825 The presence or absence of the enforcement of any part of the LOM XML binding by a par-  
826 ticular XSD in no way relieves a particular LOM instance from satisfying all of the LOM  
827 XML binding rules in order to be considered a *conforming* LOM instance. The presence of  
828 alternative XSDs merely provides developers choices in determining how to enforce the LOM  
829 XML binding rules described in this standard.

## 830 **C.1 Composite XSDs**

831 Composite XSDs are defined in C.1.1 – C.1.4. Each of these XSDs is structured from compo-  
832 nent XSDs defined in C.2.

833 NOTE—The specific names of XSDs in C.1.1 – C.1.4, `lom.xsd`, `lomLoose.xsd`,  
834 `lomStrict.xsd`, and `lomCustom.xsd`, refer to files that are accessed through URLs (see An-  
835 nex B).

### 836 **C.1.1.1 Generic approach**

837 The `lom.xsd` composite XSD encodes a particular set of binding rules in the W3C XML  
838 Schema definition language. The `lom.xsd` composite XSD

- 839 – Defines a mechanism to describe custom vocabularies (source/value pairs) in ad-  
840 dition to the vocabularies defined in the LOMv1.0 base schema. A validating  
841 parser will check that vocabulary values in a LOM instance conform to either the  
842 vocabularies expressed in the LOMv1.0 base schema or those expressed in the  
843 custom vocabulary;
- 844 – Expresses the uniqueness constraints defined by the LOMv1.0 base schema; and
- 845 – Defines a mechanism to describe extensions to the LOMv1.0 base schema. Exten-  
846 sions must be defined in a set of supporting XSDs. A validating parser will check  
847 that the extensions used in a LOM instance conform to rules defined in the sup-  
848 porting XSDs.

### 849 **C.1.1.2 Loose composite XSD**

850 The `lomLoose.xsd` composite XSD encodes a particular set of binding rules in the W3C  
851 XML Schema definition language. The `lomLoose.xsd` composite XSD

- 852 – Does not provide validation to any defined vocabulary schema (i.e., the LOMv1.0  
853 base schema or a custom schema). A validating parser will verify that the vocabu-  
854 laries are valid tokens, only;
- 855 – Does not express the uniqueness constraints defined by the LOMv1.0 base  
856 schema; and
- 857 – Defines a mechanism to describe extensions to the LOMv1.0 base schema. Exten-  
858 sions must be defined in a set of supporting XSDs. A validating parser will check

859 that the extensions used in a LOM instance conform to rules defined in the sup-  
860 porting XSDs that include extensions.

861 NOTE—The absence of the enforcement of vocabulary values does not relieve a particular LOM  
862 instance from satisfying vocabulary requirements defined in the LOMv1.0 base schema. Applica-  
863 tions that require the use of `vocab/loose.xsd` component XSD should enforce those vocabu-  
864 lary requirements by other means.

### 865 **C.1.1.3 Strict composite XSD**

866 The `lomStrict.xsd` composite XSD encodes a particular set of binding rules in the W3C  
867 XML Schema definition language such that a LOM instance will *strictly conform* to IEEE  
868 1484–12.1–2002. The `lomStrict.xsd` composite XSD

- 869 – Limits vocabularies (source/value pairs) to those defined in the LOMv1.0 base  
870 schema, only;
- 871 – Expresses the uniqueness constraints defined by LOMv1.0 base schema; and
- 872 – does not permit any extensions to the LOMv1.0 base schema.

### 873 **C.1.1.4 Custom composite XSD**

874 The `lomCustom.xsd` composite XSD encodes a particular set of binding rules in the W3C  
875 XML Schema definition language such that a LOM instance will *conform* to IEEE 1484–  
876 12.1–2002. The `lomCustom.xsd` composite XSD

- 877 – Defines a mechanism to describe custom vocabularies (source/value pairs) in ad-  
878 dition to the vocabularies defined in the LOMv1.0 base schema. A validating  
879 parser will check that vocabulary values in a LOM instance conform to either the  
880 vocabularies expressed in the LOMv1.0 base schema or those expressed in the  
881 custom vocabulary;
- 882 – Expresses the uniqueness constraints defined by the LOMv1.0 base schema; and
- 883 – Defines a mechanism to describe extensions to the LOMv1.0 base schema. Exten-  
884 sions must be defined in a set of supporting XSDs. A validating parser will check  
885 that the extensions used in a LOM instance conform to rules defined in the sup-  
886 porting XSDs that include extensions.

## 887 **C.2 Component XSDs**

888 A component XSD represents a constituent or part element of a composite XSD. The follow-  
889 ing component XSDs are used by the composite XSDs defined in C.1.

- 890 – The `common/dataTypes.xsd` component XSD defines global schema type dec-  
891 larations for data types defined in IEEE 1484.12.1–2002.
- 892 – The `common/elementNames.xsd` component XSD defines global element dec-  
893 larations for each of the data elements defined in IEEE 1484.12.1–2002. This  
894 component XSD is used to check for the uniqueness of elements declared to be  
895 unique within their parent by the presence of the `uniqueElementName` attribute.

- 896 The XML `unique` element is used to enforce multiplicity constraints of no more  
897 than one item within a LOM instance.
- 898 – The `common/elementTypes.xsd` component XSD defines global schema data  
899 type declarations for data elements defined in IEEE 1484.12.1–2002. This com-  
900 ponent XSD defines the aggregation relationship among the LOMv1.0 base  
901 schema elements. These aggregation relationships enforce the IEEE 1484.12.1–  
902 2002 requirement that elements can only be present in a LOM instance as ele-  
903 ments of the aggregate element to which they belong.
  - 904 – The `common/vocabTypes.xsd` component XSD defines global type declara-  
905 tions for those data elements that have values taken from a vocabulary data type  
906 defined in IEEE 1484.12.1–2002.
  - 907 – The `common/vocabValues.xsd` component XSD defines the standard vocabu-  
908 lary value declarations as defined in IEEE 1484.12.1–2002. These vocabulary  
909 value declarations are used in conjunction with both `vocab/custom.xsd` and  
910 `vocab/strict.xsd`.
  - 911 – The `common/rootElement.xsd` component XSD defines the element name  
912 declaration for the element that contains all other elements for a LOM instance.
  - 913 – The `common/anyElement.xsd` component XSD defines a global declaration  
914 that is re-used for extension data elements.
  - 915 – The `unique/loose.xsd` component XSD defines attribute group declarations  
916 for data elements to support the schema-based validation of uniqueness con-  
917 straints within a LOM instance where the exact set of attributes associated with  
918 each element has to be precisely as specified by the LOM XML binding (i.e.,  
919 where extra, artificial attributes must be avoided). This component XSD is used to  
920 relax the enforcement of the uniqueness constraints to avoid the introduction of  
921 the `uniqueElementName` artificial attribute (see D.3.3).
  - 922 – The `unique/strict.xsd` component XSD defines attribute group declarations  
923 for data elements defined in IEEE 1484.12.1–2002 to support schema-based vali-  
924 dation of the uniqueness constraints with a LOM instance by introducing the arti-  
925 ficial attribute `uniqueElementName` for each data element that appears with a  
926 multiplicity of at most one (see D.3.3).
  - 927 – The `extend/custom.xsd` component XSD defines the content model group  
928 `customElements` to support validation of custom data elements. This compo-  
929 nent XSD should be used if extensions are to be supported in LOM instances.
  - 930 – The `extend/strict.xsd` component XSD defines the content model group  
931 `customElements` to support strict validation of standard data elements. This  
932 component XSD should be used if extensions are not to be supported in LOM in-  
933 stances.
  - 934 – The `vocab/custom.xsd` component schema enforces strict conformance with  
935 the standard vocabulary values specified in the LOMv1.0 base schema and any  
936 custom-defined vocabulary extensions. The `vocab/custom.xsd` enforces strict  
937 adherence to the combined use of standard plus custom vocabulary values by  
938 checking that both sources and values are taken from either a token set defined in  
939 IEEE 1484.12.1–2002 or from a custom token set.

- 940 – The `vocab/strict.xsd` component schema enforces strict conformance with  
941 the standard vocabulary values specified in the LOMv1.0 base schema, but it does  
942 not allow custom vocabulary values from other sources. This component schema  
943 supports strict validation of standard vocabulary values by checking that both  
944 sources and values are from a token set defined in IEEE 1484.12.1–2002.
- 945 – The `vocab/loose.xsd` component schema relaxes the enforcement of the vo-  
946 cabulary source/value relationship constraints, which simplifies the schema vali-  
947 dation process for those applications that perform vocabulary source/value valida-  
948 tion by other means. This component schema relaxes the strict validation con-  
949 straints of the `vocab/strict.xsd` component schema by allowing both sources  
950 and values to be arbitrary strings.

951 NOTES:

952 1—The absence of the enforcement of uniqueness constraints does not relieve a particular LOM  
953 instance from satisfying the uniqueness constraints described in the LOMv1.0 base schema. Ap-  
954 plications that require the use of the `unique/loose.xsd` component XSD have to enforce those  
955 uniqueness constraints by other means.

956 2—For most applications, enforcing the uniqueness constraints using the `unique/strict.xsd`  
957 component XSD is desirable. Although adding such an artificial attribute is undesirable, it is  
958 unlikely to cause problems.

959 3—LOM instances that use the `extend/custom.xsd` component XSD and extensions will be  
960 *conforming* but not *strictly conforming* to this standard.

961 4—By using the `extend/strict.xsd` component XSD and, therefore, not supporting exten-  
962 sions, *conforming* LOM instances will *strictly conform* to this standard.

963 5—Strict adherence to standard plus custom vocabularies using the  
964 `vocab/custom.xsd` component schema is desirable under some circumstances, but may com-  
965 plicate the schema validation process.

966 6—Strict adherence to the standard vocabularies using the `vocab/strict.xsd` component  
967 schema is desirable under some circumstances, but disallowing custom vocabularies is undesir-  
968 able in other circumstances.

969 7—The absence of the enforcement of the vocabulary source/value relationship constraints by the  
970 `vocab/loose.xsd` component schema does not relieve a particular LOM instance from satisfy-  
971 ing the vocabulary constraints described in the LOMv1.0 base schema. Applications that use the  
972 `vocab/loose.xsd` component schema have to enforce those vocabulary constraints by other  
973 means.

## 974 C.3 Enabling extensions

975 With certain constraints, the LOMv1.0 base schema anticipates the need to introduce data ele-  
976 ments not included in the standard collection of elements. These constraints are

- 977       – Extensions retain the original value spaces and data types of LOMv1.0 base  
978       schema data elements; and  
979       – Extensions provide additional elements of an aggregate data element, only, and do  
980       not define data types or value spaces for aggregate data elements.

981       These requirements imply that any aggregate data element should be allowed to contain addi-  
982       tional custom data elements. This requirement can be met by including an appropriate  
983       <xs:any> element in the content model for aggregate elements (<xs:any namespace=  
984       "##other" processContents="lax"/>).

985       The `processContents = "lax"` declaration instructs an XML processor to attempt to  
986       validate the element's content. This declaration allows elements from namespaces other than  
987       the LOM namespace to be included and their contents to be validated, if schema information  
988       for the elements can be found. If such information is not available, the XML processor will  
989       allow any well-formed LOM instance.

990       NOTES:

991       1—To ensure that valid LOM instances are created, an organization that develops extensions  
992       should also provide an XSD that defines the validation rules for the extensions.

993       2—Organizations providing extensions are free to define namespaces for these extensions. If an  
994       organization provides a namespace, then the XSDs provided by this standard may have to be al-  
995       tered to support the extended element namespace.

## 996 **Annex D**

997 **(informative)**

### 998 **XSD implementation choices**

999 Major implementation choices contained in the example XSDs are discussed in D.1 – D.6.

#### 1000 **D.1 Data types**

1001 The `common/dataTypes.xsd` component XSD defines a collection of global W3C XML  
1002 Schema definition language type declarations for the data types used to constrain values for  
1003 the data elements defined in IEEE 1484.12.1–2002. The type declarations are provided so that  
1004 the logical data types in the LOMv1.0 base schema are defined in terms of their underlying  
1005 XML Schema data type only once. This modularization of the schema definition provides a  
1006 means to change the underlying schema data type, if necessary, without making any other  
1007 changes to the schema definition.

1008 The following global W3C XML Schema definition language types are defined :

- 1009 – `CharacterString`: an alias for `xs:string`.
- 1010 – `LanguageId`: an alias for `xs:language`.
- 1011 – `LanguageIdNone`: an alias for `xs:token`.
- 1012 – `LanguageIdOrNone`: a union of the `LanguageID` and `LanguageIdNone`.
- 1013 – `VCard`: an alias for `CharacterString`.
- 1014 – `MimeType`: an alias for `CharacterString`.
- 1015 – `Size`: an alias for `xs:nonNegativeInteger`.
- 1016 – `LangString`: a sequence of zero or more `<string>` elements with an optionally  
1017 defined language attribute. The language attribute describes the language of  
1018 the value held by the `<string>` element.
- 1019 – `DateTime`: optional subelements `<dateTime>` and `<description>`.
- 1020 – `Duration`: optional subelements `<duration>` and `<description>`.

1021 `LanguageId` is used to constrain the values of the various `<language>` elements. Note that  
1022 the General category's language elements use the related type `LanguageIdOrNone` (see  
1023 5.4.1).

1024 `VCard` and `MimeType` are data types that act as placeholders. These data types may be further  
1025 defined in future editions of this standard.

1026 The `LangString` type allows each `<string>` element to contain the optional attribute  
1027 language of type `LanguageId`.

1028 The `dateTime` type has the optional subelements `<dateTime>` of type `CharacterString`  
 1029 and `<description>` of type `LangString`. The `CharacterString` type for the  
 1030 `<dateTime>` subelement contains a restricted pattern of characters. The pattern, defined us-  
 1031 ing a restricted expression in W3C XML Schema definition language, is

```
1032 ([0-9]{3}[1-9] | [0-9]{2}[1-9][0-9] | [0-9][1-9][0-9]{2} | [1-9][0-9]{3})(\-(0[1-9] | 1[0-
1033 2])\-(0[1-9] | [1-2][0-9] | 3[0-1])(T([0-1][0-9] | 2[0-3])(:[0-5][0-9](:[0-5][0-9](\.[0-
1034 9){1,})Z | ((\+|\-)([0-1][0-9] | 2[0-3]):[0-5][0-9]))?)?)?)?)?
```

1035 The restricted character string pattern is used instead of the `xs:dateTime` XML Schema data  
 1036 type because of the restrictions placed on the value defined in IEEE 1484.11.1–2002. This re-  
 1037 stricted pattern cannot be enforced by the `xs:dateTime` XML Schema data type.

1038 The `Duration` type has the optional subelements `<duration>`, of type `Character-`  
 1039 `String`, and `<description>`, of the same type as the `<description>` element elsewhere.  
 1040 The `CharacterString` type for the `<duration>` subelement contains a restricted pattern  
 1041 of characters. The pattern is

```
1042 P([0-9]{1,}Y){0,1}([0-9]{1,}M){0,1}([0-9]{1,}D){0,1}(T([0-9]{1,}H){0,1}([0-
1043 9]{1,}M){0,1}([0-9]{1,}(\.[0-9]{1,}){0,1}S){0,1}){0,1}
```

1044 The restricted character string pattern is used instead of the `xs:duration` XML Schema data  
 1045 type because of the restrictions placed on the value defined in IEEE 1484.11.1–2002. This re-  
 1046 stricted pattern cannot be enforced by the `xs:duration` XML Schema data type.

## 1047 D.2 Elements

1048 The `common/elementNames.xsd` component XSD defines a collection of global W3C  
 1049 XML Schema definition language type declarations for the LOM data elements. The compo-  
 1050 nent XSD contains 78 type declarations, one for each LOM data element. Duplicate type dec-  
 1051 larations for elements that share element names are included within XML comments for com-  
 1052 pleteness. Those elements included within XML comments should be left in XML comments.  
 1053 The global schema type declarations are provided so that the composite schema is defined in  
 1054 terms of a collection of underlying types, rather than being defined directly in terms of XML  
 1055 Schema constructs. This modularization of the schema definition provides a means to change  
 1056 the definition of the elements without making any other changes to the schema definition.

1057 For each LOM data element, an XML element name is assigned by convention using camel-  
 1058 case spelling of the full name of the data element with an initial lower-case letter  
 1059 (*lowerCamelCase*).

1060 Each element is provided with a global W3C XML Schema definition language element dec-  
 1061 laration, a global W3C XML Schema definition language type declaration, and a global W3C  
 1062 XML Schema definition language attribute group declaration. The XML Schema element  
 1063 name is used for the name of each of these declarations.

1064 Typically, the W3C XML Schema definition language type declaration for an element will  
 1065 provide the content model for the element, referring to the global element declarations for  
 1066 subelements as needed and including the attributes from the W3C XML Schema definition  
 1067 language attribute group. LOMv1.0 base schema aggregate elements will have declarations  
 1068 that are represented by complex types. LOMv1.0 base schema simple elements will have dec-  
 1069 larations that are represented by simple types.

## 1070 D.3 Aggregates

1071 The selection of the W3C XML Schema definition language structure for aggregate data ele-  
 1072 ments involves satisfying three competing requirements that arise either directly or indirectly  
 1073 from the LOMv1.0 base schema. These are

- 1074 a) An ordering restriction is not defined in IEEE 1484.12.1–2002. Each element  
 1075 should allow arbitrary ordering of its subelements.
- 1076 b) Each element should enforce multiplicity constraints defined in IEEE 1484.12.1–  
 1077 2002 on its subelements.
- 1078 c) The aggregation structure should be preserved without the introduction of artifi-  
 1079 cial container elements (i.e., elements that do not map directly to those defined in  
 1080 IEEE 1484.12.1–2002, but are introduced to meet the first two requirements).

1081 Requirements (a) and (b) introduce complexity in expressing the content model for aggregate  
 1082 elements. Requirement (c) arises from the implied constraint that the content model should  
 1083 match the LOMv1.0 base schema as closely as possible. The W3C XML Schema definition  
 1084 language does not provide a way to satisfy requirements (a) and (b) without the introduction of  
 1085 artificial container elements.

1086 Three alternatives for the selection of the content model for aggregate elements could be im-  
 1087 plemented.

### 1088 D.3.1 Using <xs:sequence>

1089 The use of the XML <xs:sequence> construct defines a way to meet the requirements of  
 1090 multiplicity (b) and avoiding artificial containers (c) but does not provide arbitrary ordering  
 1091 (a). If subelements are required to appear in a specific order, the XSD can easily enforce the  
 1092 multiplicity constraints without introducing artificial container elements. Figure D.1 shows  
 1093 how the XML sequence construct <xs:sequence> can be used to define the LOMv1.0 base  
 1094 schema.

```

1095
1096 <xs:complexType name="lom">
1097   <xs:sequence>
1098     <xs:element minOccurs="0" ref="general"/>
1099     <xs:element minOccurs="0" ref="lifeCycle"/>
1100     <xs:element minOccurs="0" ref="metaMetadata"/>
1101     <xs:element minOccurs="0" ref="technical"/>
1102     <xs:element minOccurs="0" maxOccurs="unbounded"
1103 ref="educational"/>
  
```

```

1104     <xs:element minOccurs="0" ref="rights"/>
1105     <xs:element minOccurs="0" maxOccurs="unbounded" ref="relation"/>
1106     <xs:element minOccurs="0" maxOccurs="unbounded"
1107 ref="annotation"/>
1108     <xs:element minOccurs="0" maxOccurs="unbounded"
1109 ref="classification"/>
1110   </xs:sequence>
1111 </xs:complexType>
1112

```

1113

1114

### Figure D.1—An example using <xs:sequence>

1115 According to Figure 1 the general element can appear zero or one time (satisfies [b]). The ex-  
 1116 ample does not introduce artificial container elements (satisfies [c]). However, the general ele-  
 1117 ment, if present, must precede the lifecycle element (does not satisfy [a]).

### 1118 D.3.2 Using <xs:all>

1119 The use of the XML <xs:all> construct defines a way to meet the requirements of arbitrary  
 1120 ordering (a) and multiplicity (b). However the <xs:all> construct introduces the use of arti-  
 1121 ficial container elements (c). The XML <xs:all> construct requires the introduction of arti-  
 1122 ficial container elements to collect multiple instances of a given element. The LOMv1.0 base  
 1123 schema contains elements that can have multiplicities of more than one (e.g., Educational).  
 1124 Figure D.2 illustrates how the XML sequence construct <xs:all> can be used to define the  
 1125 LOMv1.0 base schema.

```

1126 <xs:complexType name="lom">
1127   <xs:all>
1128     <xs:element minOccurs="0" ref="general"/>
1129     <xs:element minOccurs="0" ref="lifeCycle"/>
1130     <xs:element minOccurs="0" ref="metaMetadata"/>
1131     <xs:element minOccurs="0" ref="technical"/>
1132     <xs:element minOccurs="0" ref="educational"/> <!-- container -->
1133     <xs:element minOccurs="0" ref="rights"/>
1134     <xs:element minOccurs="0" ref="relations"/> <!-- container -->
1135     <xs:element minOccurs="0" ref="annotations"/> <!-- container -->
1136     <xs:element minOccurs="0" ref="classifications"/> <!-- container
1137 -->
1138   </xs:all>
1139 </xs:complexType>
1140

```

1141

1142

1143

### Figure D.2—An example using <xs:all>

1144 According to Figure 2 the general element can appear zero or one time (b) anywhere in the  
 1145 sequence of elements (a). However, the example introduces artificial container elements, such  
 1146 as educational (c), where the plural educational indicates zero or more educational elements.

### 1147 **D.3.3 Using <xs:choice>**

1148 The use of the XML choice construct defines a way to meet the requirements of arbitrary or-  
 1149 dering (a) and avoiding artificial containers (c) but does enforce multiplicity constraints (b). A  
 1150 more flexible solution can be found by temporarily giving up on the enforcement of the multi-  
 1151 plicity constraints. By allowing any subelement to appear any number of times, the W3C  
 1152 XML Schema definition language types can easily be constructed to allow arbitrary ordering  
 1153 of the subelements without having to introduce artificial container elements. Figure D.3 shows  
 1154 how the XML sequence construct <xs:choice> can be used to define the LOMv1.0 base  
 1155 schema.

```

1156 <xs:complexType name="lom">
1157   <xs:choice minOccurs="0" maxOccurs="unbounded">
1158     <xs:element ref="general"/>
1159     <xs:element ref="lifeCycle"/>
1160     <xs:element ref="metaMetadata"/>
1161     <xs:element ref="technical"/>
1162     <xs:element ref="educational"/>
1163     <xs:element ref="rights"/>
1164     <xs:element ref="relation"/>
1165     <xs:element ref="annotation"/>
1166     <xs:element ref="classification"/>
1167   </xs:choice>
1168 </xs:complexType>
  
```

1170

1171

**Figure D.3—An example using <xs:choice>**

1172

1173 The example in Figure 3 does not introduce artificial container elements (satisfies [c]) and al-  
 1174 lows arbitrary ordering (satisfies [a]). However, all of the elements may appear any number of  
 1175 times (does not satisfy [b]).

1176 This solution accepts a superset of what the LOMv1.0 base schema allows without introduc-  
 1177 ing additional complexity in the LOM instances or the LOM XSD. For applications willing to  
 1178 enforce LOM element uniqueness constraints on the post-schema validation XML infoset, this  
 1179 may represent a sufficient solution.

### 1180 **D.3.4 Using <xs:choice> with <xs:unique>**

1181 The example in Figure D.3 does not enforce multiplicity constraints (b) for those elements that  
 1182 can appear zero or one time only. The multiplicity constraints can be enforced using the iden-  
 1183 tity constraint <xs:unique>. For each element with a multiplicity of at most one in its par-  
 1184 ent, define a hidden attribute called "uniqueElementName" with a fixed default value equal  
 1185 to the name of the element, and use this attribute as the element key for <xs:unique>. Figure  
 1186 D.4 shows how the XML sequence construct <xs:unique> can be used to enforce multiplic-  
 1187 ity constraints for element uniqueness when the LOMv1.0 base schema defines that an ele-  
 1188 ment cannot appear more than one time.

```

1189 <xs:element name="lom" type="lom">
1190   <xs:unique name="lomUnique">
1191     <xs:selector xpath="*" />
1192     <xs:field xpath="@uniqueElementName" />
1193   </xs:unique>
1194 </xs:element>
1195
1196
1197

```

1198 **Figure D.4—An example using <xs:unique>**

1199 One such constraint using the construct <xs:unique> is needed for each parent element that  
1200 contains subelements with multiplicities of at most one.

1201 The disadvantages of this approach are the introduction of the "uniqueElementName" at-  
1202 tribute and the introduction of a limited amount of additional complexity in the XSD.

1203 This solution maximizes the amount of validation that can be done by a parser. For applica-  
1204 tions that are not influenced by the addition of the "uniqueElementName" attribute during  
1205 parsing, this may represent a sufficient solution.

### 1206 **D.3.5 Summary**

1207 Subclauses D.3.1 – D.3.4 defined alternative approaches to aggregate content models. A sum-  
1208 mary of those alternatives is given below

- 1209
- 1210 – **<xs:sequence>**: Does not satisfy requirement (a), which states that the subele-  
1211 ments appear in an arbitrary order. This alternative does not conform to IEEE  
1212 1484.12.1-2002.
- 1213 – **<xs:all>**: Does not satisfy requirement (c), which preserves the aggregation  
1214 structure defined in IEEE 1484.12.14--2002. The use of <xs:all> requires the  
1215 introduction of arbitrary container elements (e.g., <educational>). This alterna-  
1216 tive requires all subelements of the container to be collected under the parent con-  
1217 tainer element.
- 1218 – **<xs:choice>**: Does not enforce uniqueness constraints. All of the elements may  
1219 appear any number of times. This alternative does not satisfy requirement (b) and  
1220 does not conform to IEEE 1484.12.1-2002. This alternative is defined in the  
1221 unique/loose.xsd component XSD (see C.2).
- 1222 – **<xs:choice> with <xs:unique> attribute**: Overcomes the problems with  
1223 <xs:choice>. A hidden attribute is applied to each element with a multiplicity  
1224 of at most one. The hidden attribute is enforced by applying an identity constraint.  
1225 This alternative is defined in the unique/strict.xsd component XSD (see  
1226 C.2).

## 1227 D.4 Vocabularies

1228 The selection of the content model for vocabulary data elements involves satisfying three re-  
 1229 quirements that arise either directly or indirectly from the LOMv1.0 base schema and from  
 1230 how vocabularies are to be used by applications. These are

- 1231 – Vocabulary elements should allow values from arbitrary sources.
- 1232 – If the source of a vocabulary value is LOMv1.0 (i.e.,  
 1233 <source>LOMv1.0<\source>), then the value space is described by the  
 1234 LOMv1.0 base schema.
- 1235 – If the source of a vocabulary value is not LOMv1.0 (i.e., not  
 1236 <source>LOMv1.0<\source>), then the value space should not conflict with  
 1237 the LOMv1.0 base schema.

1238 Unfortunately, the W3C XML Schema definition language cannot easily accommodate the  
 1239 goal of strict type checking for vocabulary values in a one-pass model that will meet the needs  
 1240 of the wide range of applications anticipated for the LOM XML Schema binding.

1241 The W3C XML Schema definition language can provide validation of a fixed set of vocabu-  
 1242 laries, only. In LOM instances, the fixed vocabularies are defined based on the source of the  
 1243 vocabulary being made available. If the source of the vocabulary is defined, W3C XML  
 1244 Schema definition language constructs can be defined to aid in the validation of the vocabular-  
 1245 ies. Three alternatives for expressing LOM vocabularies in W3C XML Schema definition  
 1246 language are presented in D.4.1 – D.4.3. These three alternatives provide options on how the  
 1247 validation approaches are taken by XML Schema processors.

### 1248 D.4.1 vocab/loose.xsd

1249 If requirement exists for validation of vocabulary values from the LOMv1.0 base schema vo-  
 1250 cabularies, then the vocab/loose.xsd component schema may be used. The  
 1251 vocab/loose.xsd component schema provides a way to meet the goal of allowing values  
 1252 from arbitrary sources (a) but does not provide validation. This approach is the simplest alter-  
 1253 native in terms of the complexity of the XSD. Each element that takes its values from a vo-  
 1254 cabulary is defined as a `CharacterString` as shown in Figure D.5.

```
1255
1256 <xs:simpleType name="difficulty">
1257   <xs:restriction base="lom:CharacterString"/>
1258 </xs:simpleType>
1259
```

1260

1261 **Figure D.5—An example using vocab/loose.xsd**

1262 Both the <source> and the <value> are optional and may appear in any order. There are no  
 1263 constraints on vocabulary data elements.

1264 This approach accepts a superset of what the LOMv1.0 base schema allows without introduc-  
 1265 ing additional complexity in the LOM instances or the LOMv1.0 base schema. The disadvan-  
 1266 tage of this approach is that it does not allow validation of vocabulary values. Using the vo-  
 1267 cab/loose.xsd component schema requires other validation steps to validate the vocabu-  
 1268 lary values used in both *strictly conforming* and *conforming* LOM instances.

#### 1269 **D.4.2 vocab/strict.xsd**

1270 If vocabularies are limited to only those defined with the LOMv1.0 base schema, then a strict  
 1271 vocabulary validation approach may be desirable. The vocab/strict.xsd component  
 1272 schema defines rules for a strict validation of LOMv1.0 base schema vocabularies. The use of  
 1273 the vocab/strict.xsd component schema requires the source to be LOMv1.0 (i.e.,  
 1274 <source>LOMv1.0<\source>) and the value to be a valid LOM-defined vocabulary. This  
 1275 approach is a straightforward implementation of the standard values allowed by the vocabu-  
 1276 lary elements. As shown in Figure D.6, each element that takes its values from a vocabu-  
 1277 lary is defined where the <source> element is given a fixed value of LOMv1.0, and the value of the  
 1278 <value> element is derived from an enumerated list of the appropriate vocabulary values.

```

1279
1280 <xs:simpleType name="difficulty">
1281   <xs:union memberTypes="lom:difficultyValues" />
1282 </xs:simpleType>
1283
1284 <xs:complexType name="difficultyVocab">
1285   <xs:choice minOccurs="0" maxOccurs="unbounded">
1286     <xs:element name="source" type="sourceValue" />
1287     <xs:element name="value" type="difficultyValue" />
1288     <xs:group ref="ex:customElements" />
1289   </xs:choice>
1290 </xs:complexType>
1291
1292 <xs:simpleType name="difficultyValues">
1293   <xs:restriction base="xs:token">
1294     <xs:enumeration value="very easy" />
1295     <xs:enumeration value="easy" />
1296     <xs:enumeration value="medium" />
1297     <xs:enumeration value="difficult" />
1298     <xs:enumeration value="very difficult" />
1299   </xs:restriction>
1300 </xs:simpleType>
1301

```

1302 **Figure D.6—An example using vocab/strict.xsd**

1303 Both <source> and <value> are optional, and may appear in any order. Values are vali-  
 1304 dated against the vocabulary tokens defined in this standard for the LOMv1.0 base schema.

1305 This approach accepts a subset of what the LOMv1.0 base schema would allow by excluding  
 1306 vocabulary values with sources other than the LOMv1.0 base schema. The advantage of this  
 1307 approach is that it can validate standard vocabulary values in the XSD.

### 1308 D.4.3 vocab/custom.xsd

1309 If vocabularies are limited to both those defined with the LOMv1.0 base schema and a custom  
 1310 defined set of vocabularies, then a custom vocabulary validation approach may be desirable.  
 1311 The vocab/custom.xsd component schema defines rules for validation of both LOMv1.0  
 1312 base schema vocabularies and vocabularies that are defined by other schemas. The use of the  
 1313 vocab/custom.xsd component schema provides a way to meet the goal of allowing values  
 1314 from arbitrary sources (a) but does not meet the requirements of arbitrary ordering (b) and  
 1315 avoiding artificial containers (c). This approach supplies additional flexibility in the partial  
 1316 validation of both standard and custom vocabulary values. As shown in Figure D.7 each ele-  
 1317 ment that takes its values from a vocabulary is defined where the <source> element is al-  
 1318 lowed to have an arbitrary value, and the value of the <value> element is derived from a cus-  
 1319 tom list of the appropriate vocabulary values.

```

1320
1321 <xs:simpleType name="difficulty">
1322   <xs:union memberTypes="lom:difficultyValues lx:difficultyValues"/>
1323 </xs:simpleType>
1324
1325 <xs:complexType name="difficulty">
1326   <xs:complexContent>
1327     <xs:extension base="difficultyVocab">
1328       <xs:attributeGroup ref="ag:difficulty"/>
1329     </xs:extension>
1330   </xs:complexContent>
1331 </xs:complexType>
1332
1333 <xs:simpleType name="difficultyValues">
1334   <xs:restriction base="xs:token">
1335     <xs:enumeration value="very easy"/>
1336     <xs:enumeration value="easy"/>
1337     <xs:enumeration value="medium"/>
1338     <xs:enumeration value="difficult"/>
1339     <xs:enumeration value="very difficult"/>
1340   </xs:restriction>
1341 </xs:simpleType>
1342

```

1343 **Figure D.7—An example using vocab/custom.xsd**

1344 Both <source> and <value> are optional and may appear in any order. Values are validated  
 1345 against the custom vocabulary enumerations.

1346 This approach accepts a superset of what the LOMv1.0 base schema allows and allows the use  
 1347 of both standard and custom the vocabulary values. The advantage of this approach is that it  
 1348 retains the ability for the validation of standard vocabulary values while accommodating the  
 1349 presence of custom vocabulary values. However, the example in Figure 7 does not restrict vo-  
 1350 cabulary values to the LOMv1.0 base schema vocabulary values when the source is LOMv1 . 0  
 1351 (i.e., <source>LOMv1 . 0<\source>). Nor does the example prevent the use of LOMv1.0  
 1352 base schema vocabulary values if the source is not LOMv1 . 0.

1353 NOTE—If an organization plans to extend the vocabulary values and is following an XSD design  
1354 similar to the one defined in this standard, the organization may define the namespace that defines  
1355 the extended vocabularies. The `vocab/custom.xsd` component schema provides a placeholder  
1356 for this extension (`xmlns:lx="http://ltsc.ieee.org/XSD/LOM/custom"`). The organiza-  
1357 tion may change this declaration.

## 1358 D.5 Additional notes

1359 Some additional decisions that were made to enforce conformance to the LOMv1.0 base  
1360 schema. The following additional notes apply to implementation choices:

- 1361 – Certain elements are defined in IEEE 1484.12.1–2002 to be represented as valid  
1362 vCard syntax. At the time of publication of this standard, there was no *de facto*  
1363 standard W3C XML Schema definition language binding for the vCard specifica-  
1364 tion. XSD implementations may want to define a type to encapsulate and define a  
1365 type for those elements that have a value of a vCard. The vCard data type does  
1366 not validate that the vCard value represents a well-formed Internet Mail Consor-  
1367 tium 3.0 vCard (see IETF RFC 2426:1998).
- 1368 – The `MimeType` global W3C XML Schema definition language type is defined to  
1369 encapsulate and define a type for those elements that are defined to hold a MIME  
1370 type. The `MimeType` type does not validate that the `MimeType` value represents a  
1371 valid MIME type (see IETF RFC 2425 [A2]).
- 1372 – A global element type of `LanguageIdOrNone` has been created to encapsulate  
1373 the declaration for the `Language` element (see the `General` element, 5.4.1). This  
1374 type allows the `Language` element to have two values: a `Language` identifier (as  
1375 defined in ISO 639 and ISO 3166) or the token `none` (indicating no language).
- 1376 – An element type of `LanguageId` has been created to encapsulate the declaration  
1377 for the `Language` element (see the `Educational` element, 5.4.5). This type allows  
1378 the `Language` element to have a `Language` identifier (as defined in ISO 639 and  
1379 ISO 3166) and multiplicity greater than one.
- 1380 – The global `<description>` element has content model `LangString` and multi-  
1381 plicity one.
- 1382 – Local element declarations for the `Description` element (see the `General` element,  
1383 5.4.1) and for the `Description` element (see the `Educational` element, 5.4.5) allow  
1384 these elements to have multiplicity greater than one.
- 1385 – The global `<entity>` element has content model `vCard` and multiplicity one.
- 1386 – The enumerated values for the `Name` element are listed as dependent on the value  
1387 of the corresponding `Type` element (see the `OrComposite` element, 5.4.4.3.1). The  
1388 values of the `<name>` element in the XSD are given as the union of the listed val-  
1389 ues.
- 1390 – A local element declaration for the `Source` element (see the `Taxon path` element,  
1391 5.4.9.2) allows the validation of vocabulary elements to provide a global element  
1392 declaration for the `<source>` element that is appropriate for the source of a vo-  
1393 cabulary value.