

1 **IEEE 1484.11.1, Draft 2**
2 **Draft Standard for Learning Technology—Data**
3 **Model for Content Object Communication**

4
5 Sponsored by the Learning Technology Standards Committee
6 of the IEEE Computer Society

7

8 Copyright © 2003 by the Institute of Electrical and Electronics Engineers, Inc.
9 Three Park Avenue
10 New York, NY 10016-5997, USA

11 All rights reserved. This document is an unapproved draft of a proposed IEEE Standard. As
12 such, this document is subject to change. USE AT YOUR OWN RISK! Because this is an
13 unapproved draft, this document must not be utilized for any conformance/compliance pur-
14 poses. Permission is hereby granted for IEEE Standards Committee participants to reproduce
15 this document for purposes of IEEE standardization activities only. Prior to submitting this
16 document to another standards development organization for standardization activities, per-
17 mission must first be obtained from the Manager, Standards Licensing and Contracts, IEEE
18 Standards Activities Department. Other entities seeking permission to reproduce this docu-
19 ment, in whole or in part, must obtain permission from the Manager, Standards Licensing and
20 Contracts, IEEE Standards Activities Department.

21

22 IEEE Standards Activities Department
23 Standards Licensing and Contracts
24 445 Hoes Lane, P.O. Box 1331
25 Piscataway, NJ 08855-1331, USA

26

27 **Abstract:**

28 This Standard describes a data model to support the interchange of data elements and their
29 values between a content object and a runtime service (RTS). It is based on a current industry
30 practice called "CMI—Computer Managed Instruction." The work upon which this Standard
31 is based was developed to support a client/server environment in which a learning system,
32 generically called a learning management system (LMS), delivers digital content, called con-
33 tent objects, to learners. The data model supports learner data and preferences, interactions,
34 objectives, content-object entry, exit, and status information, time parameters, and scores.

35 **Keywords:**

36 content object, computer managed instruction, CMI, data model, interoperability, learning
37 content, learning management system, LMS, runtime service, RTS.

38 *[Note: Information about IEEE LTSC P1484.11 can be found at:*

39 <http://ieee.ltsc.org/wg11>

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

41 Introduction

42 (This introduction is not a part of P1484.11.1, Draft Standard for Learning Technology—Data
43 Model for Content Object Communication.)

44 This document describes a data model for communication between content objects and a run-
45 time service. This data model may be extended as other data communication needs are identi-
46 fied.

47

48 Participants

49 At the time this Standard was completed, the working group had the following membership:

Tyde Richards, <i>Chair</i> Jack Hyde, <i>Chair (December, 1998 – March, 2001)</i> Scott Lewis, <i>Technical Editor</i>		
Mitchell Bonnett	Tom King	Claude Ostyn
Frank Farance	Rolf Lindner	Daniel Rehak
Mike Fore	Kiyoshi Nakabayashi	Robby Robson
Leonard Greenberg	Boyd Nielsen	Schawn Thropp

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

52

53	Contents	
54	1. Overview.....	6
55	1.1 Scope	6
56	1.2 Purpose.....	6
57	2. Normative references.....	7
58	3. Definitions.....	8
59	3.1 Abbreviations and acronyms.....	9
60	4. Conformance	10
61	4.1 Data instances	10
62	4.2 Sending implementations.....	10
63	4.3 Receiving Implementations.....	10
64	4.4 Repository implementations.....	10
65	4.5 Smallest permitted maximum values.....	10
66	5. Conceptual model.....	11
67	6. Data model	12
68	6.1 content_object_communication.....	12
69	6.1.1 comments_from_learner	14
70	6.1.2 comments_from_lms.....	14
71	6.1.3 completion_status.....	14
72	6.1.4 credit.....	15
73	6.1.5 data_model_version	15
74	6.1.6 entry.....	16
75	6.1.7 exit.....	16
76	6.1.8 interaction	17
77	6.1.9 launch_data.....	27
78	6.1.10 learner_id.....	28
79	6.1.11 learner_name	28
80	6.1.12 learner_preference_data.....	28
81	6.1.13 lesson_status	31
82	6.1.14 location.....	31
83	6.1.15 max_time_allowed.....	32
84	6.1.16 mode.....	32
85	6.1.17 objective.....	32
86	6.1.18 raw_passing_score	35
87	6.1.19 scaled_passing_score	36
88	6.1.20 score	36
89	6.1.21 session_time.....	36
90	6.1.22 success_status.....	37
91	6.1.23 suspend_data.....	37
92	6.1.24 time_limit_action.....	38
93	6.1.25 total_time	38
94	6.2 Auxiliary data types.....	38
95	6.2.1 con_ob_con_comment	39
96	6.2.2 con_ob_con_localized_string.....	40
97	6.2.3 con_ob_con_long_identifier.....	41

98 6.2.4 con_ob_con_score 41
99 6.2.5 con_ob_com_short_identifier 43
100 **Annex A** **44**
101 **Annex B** **45**
102 B.1 Real data type 45
103 B.2 Time interval data type 45
104 B.3 Time and date data type (time stamp) 46

105 **Draft Standard for Learning Technology—Data** 106 **Model for Content Object Communication**

107 **1. Overview**

108 The scope and purpose of this Standard are discussed in subclauses 1.1 and 1.2.

109 **1.1 Scope**

110 This Standard describes a data model to support the interchange of agreed upon data elements
111 and their values between a learning-related content object and a runtime service (RTS) used to
112 support learning management. This Standard does not specify the means of communication
113 between a content object and an RTS nor how any component of a learning environment shall
114 behave in response to receiving data in the form specified. This Standard is based on a related
115 data model defined in the "Computer Managed Instruction (CMI) Guidelines For Interopera-
116 bility," version 3.4 [A1]¹, defined by the Aviation Industry CBT Committee (AICC). To bal-
117 ance the need to support existing implementations with the need to make technical corrections
118 and support emerging practice, this Standard selectively includes those data elements from the
119 CMI specification that are commonly implemented; renames some data elements taken from
120 the CMI specification to clarify their intended meaning; modifies the data types of data ele-
121 ments taken from the CMI specification to reflect ISO standard data types and internationali-
122 zation requirements; removes some organizational structures used in the CMI specification to
123 group data elements that are specific to the AICC community of practice and not generally
124 applicable; and introduces some data elements not present in the CMI specification to correct
125 known technical defects in data elements taken from that specification.

126 **1.2 Purpose**

127 There is widespread acknowledgement that the data model for content object communication
128 defined in the AICC "Computer Managed Instruction (CMI) Guidelines for Interoperability",
129 version 3.4, has broad applicability to systems used for learning management. The purpose of
130 this Standard is to build consensus around, resolve ambiguities, and correct defects in this data
131 model for the data exchanged between learning-related content and an RTS used to support
132 learning management.

¹ The numbers in brackets correspond to those of the bibliography in Annex A.

133 **2. Normative references**

- 134 This Standard shall be used in conjunction with the following publications.
- 135 ISO 639:1988, Code for the representation of names of languages.
- 136 ISO 639-2:1998, Codes for the representation of names of languages – Part 2: Alpha-3 code.
- 137 ISO/IEC 646:1991, Information technology – ISO 7-bit coded character set for information
138 interchange.
- 139 ISO 3166-1:1997, Codes for the representation of names of countries and their subdivisions –
140 Part 1: Country codes.
- 141 ISO/IEC 10646-1:2000, Information technology – Universal Multiple-Octet Coded Character
142 Set (UCS)—Part 1: Architecture and Basic Multilingual Plane
- 143 ISO/IEC 11404:1996, Information technology – Programming languages, their environments
144 and system software interfaces – Language-independent datatypes.
- 145 RFC 2396, "Uniform Resource Identifiers (URI): Generic Syntax," Network Working Group,
146 August 1998.

147 3. Definitions

148 For purposes of this Standard, the following terms and definitions apply. IEEE 100, *The Au-*
149 *thoritative Dictionary of IEEE Standards Terms*, Seventh Edition [A3], should be referenced
150 for terms not defined in this Clause.

151 **content object:** A collection of digital content that is intended for presentation to a learner by
152 a learning system. A content object may include learning material and processing code. Ex-
153 ample: A content object might be an HTML page with an embedded video clip and an
154 ECMAScript written in accordance with *Draft Standard for ECMAScript API for Content to*
155 *Runtime Services Communication* [A2].

Comment: Update when standard is final.

156 **interaction:** An information exchange between a learner and a system executing a content
157 object.

158 **launch (v.):** To cause a content object to be delivered to a learner.

159 **learner:** An individual engaged in acquiring knowledge or skills with a learning technology
160 system.

161 **learner attempt:** A tracked effort by a learner to satisfy the requirements of a learning activity
162 that uses a content object. An attempt may span one or more learner sessions. The attempt be-
163 gins with the beginning of the first learner session and continues until the activity terminates.
164 The state of the attempt may be suspended between learner sessions. *See also:* **learner ses-**
165 **sion.**

166 **learner session:** An uninterrupted period of time during which a learner is accessing a content
167 object. *See also:* **learner attempt.**

168 **learning management system (LMS):** A computer system that may include the capabilities
169 to register learners, schedule learning resources, control and guide the learning process, ana-
170 lyze and report learner performance, and schedule and track learners. *See also:* **runtime ser-**
171 **vice.**

172 NOTE—Some implementations of learning management systems also have the ability to launch
173 and deliver content. For this Standard, these capabilities are known as a runtime service.

174 **runtime service (RTS):** Software that controls the execution and delivery of learning content
175 and that may provide services such as resource allocation, scheduling, input-output control,
176 and data management. *See also:* **learning management system.**

177 **score:** A result of assessing a learner, expressed as a numerical value or a point on a descrip-
178 tive scale.

179 **3.1 Abbreviations and acronyms**

180 CMI computer managed instruction

181 IANA Internet Assigned Numbers Authority

182 LMS learning management system

183 RTS runtime service

184 SPM smallest permitted maximum

185 URI Uniform Resource Identifier

186 URN Uniform Resource Name

187 **4. Conformance**

188 Conformance to this Standard is discussed in subclauses 4.1 through 4.5.

189 **4.1 Data instances**

190 A conforming data instance an instance of the data model as defined in Clause 6.1.

191 **4.2 Sending implementations**

192 A conforming sending implementation shall send data instances that conform to this Standard.

193 **4.3 Receiving Implementations**

194 A conforming receiving implementation shall accept data instances that conform to this Stan-
195 dard.

196 **4.4 Repository implementations**

197 A conforming repository implementation of this Standard shall receive, store, and send con-
198 forming data upon subsequent request.

199 **4.5 Smallest permitted maximum values**

200 This Standard defines smallest permitted maximum (SPM) values for data elements with
201 datatypes that include bag, array, set, and characterstring. For these data elements, a receiving
202 implementation or a repository implementation that conforms to this Standard shall accept and
203 process at least that number of entries or characters specified by its SPM and may accept and
204 process a larger number.

205 NOTE 1—The intent is for the SPM values to cover most cases.

206 NOTE 2—What "processing" means in the above depends on the nature of the application.

207 NOTE 3—This Standard does not define any provision for how and whether a sending system
208 can determine whether a receiving system can process more than the SPM for a particular data
209 element.

210 5. Conceptual model

211 (This Clause is informative and not normative.)

212 In one conceptual model for the use of this Standard, shown in Figure 1, the learner interacts
 213 with a content object in the learning environment. The content object may require information
 214 about the learner. It acquires this information through an RTS, which, in turn, gets the infor-
 215 mation from an LMS.

216

217

218

219

220

221

222

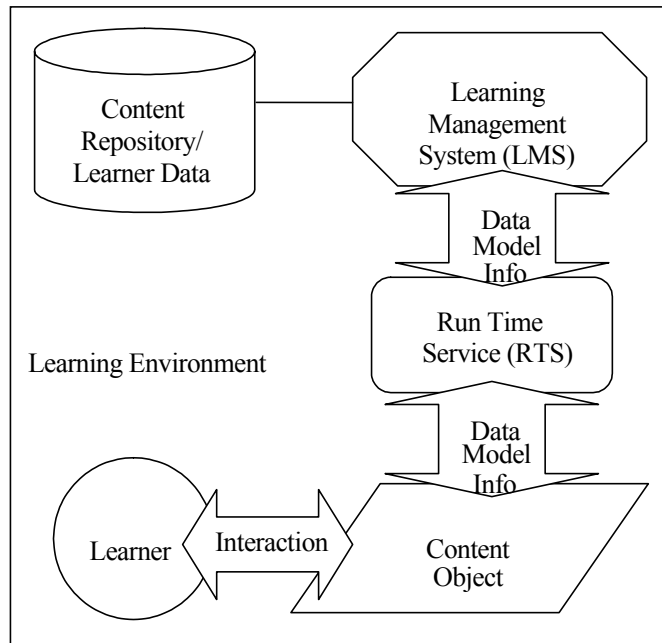
223

224

225

226

227



228

Figure 1—Conceptual model diagram

229 As the learner interacts with the content object, the content object may gather information on
 230 the learner's performance. This information is passed to the RTS, which passes it on to the
 231 LMS.

232 Other conceptual models exist that may use the data model. Although this conceptual model
 233 includes an RTS and an LMS, they are not required for the use of the data model. This con-
 234 ceptual model is only designed to describe one possible use of the data model.

235 6. Data model

236 This Clause defines a data model that a content object can use for obtaining information from
 237 an RTS to enable the content object to perform its expected functions. An RTS can use the
 238 data model for obtaining information from a content object to enable the RTS to manage the
 239 content object properly.

240 The data model in this Standard provides a description of the information that may pass to and
 241 from the content object. This Standard does not specify how, when, or in which direction the
 242 information may flow. In addition, this Standard does not specify permanence of the data, how
 243 often it may be written or rewritten, and by whom it may be created or destroyed.

244 Unless noted otherwise, all components of "records" are optional in a data instance.

245 NOTE 1—The ISO/IEC 11404 notation describes the semantics of the abstract datatypes across
 246 all bindings (e.g., implementation of a datatype as itself, its subtypes, its subclasses, and its spe-
 247 cializations). As long as the requirements of the ISO/IEC 11404 datatype are maintained (e.g.,
 248 properties, characterizing operations, value space, etc.), the implementation will have compatible
 249 semantics with other implementations. For example, an ISO/IEC 11404 "array" may be imple-
 250 mented as an SQL table (because SQL tables support indexing, a requirement for arrays); an
 251 ISO/IEC 11404 "state" may be implemented as a C programming language bit field; an ISO/IEC
 252 11404 "characterstring" may be implemented in an encoding (ISO 646, ASCII, ISO 8859-1, UTF-
 253 8, UTF-16, UTF-32, etc.) that supports the repertoire specified in the parameter to characterstring
 254 datatype. Bindings provide standard implementations of the abstract datatypes.

255 6.1 content_object_communication

256 Synopsis

```

257     type content_object_communication :
258     record
259     (
260         comments_from_learner :
261             array(0..99) of con_ob_con_comment,
262             // array dimension is as an SPM
263         comments_from_lms :
264             array(0..99) of con_ob_con_comment,
265             // array dimension is an SPM
266         completion_status :
267             state( completed, incomplete, not_attempted, unknown ),
268         credit :
269             state( credit, no_credit ),
270         data_model_version :
271             characterstring(iso-10646-1), // SPM: 255
272         entry :
273             state( ab_initio, resume, _nil_ ),
274         exit :
```

```

275     state( timeout, suspend, logout, _nil_ ),
276     interactions :
277     bag of interaction, // SMP: 100
278     launch_data :
279     characterstring(iso-10646-1), // SPM: 4096
280     learner_id :
281     con_ob_con_long_identifier,
282     learner_name :
283     con_ob_con_localized_string(255),
284     //parameter value is an SPM
285     learner_preference :
286     learner_preference_data,
287     lesson_status :
288     state( passed, completed, failed, incomplete, browsed,
289           not_attempted ),
290     location :
291     characterstring(iso-10646-1), // SPM: 255
292     max_time_allowed :
293     timeinterval(second,10,2),
294     mode :
295     state( browse, normal, review ),
296     objectives :
297     set of objective, // SPM: 100
298     raw_passing_score :
299     real(10,7),
300     scaled_passing_score :
301     real(10,7) range(-1..1),
302     score :
303     con_ob_con_score,
304     session_time :
305     timeinterval(second,10,2),
306     success_status :
307     state( passed, failed, unknown ),
308     suspend_data :
309     characterstring(iso-10646-1), // SPM: 4096
310     time_limit_action :
311     state( exit_message, continue_message, exit_no_message,
312           continue_no_message ),
313     total_time :
314     timeinterval(second,10,2),
315     ),

```

316 **Description**

317 The components of content_object_communication are defined in subclauses 6.1.1
318 through 6.1.25. Depending on the direction, destination, and purpose of the communication,
319 an instance of content_object_communication includes at least one of but typically not
320 all of the defined components.

321 **6.1.1 comments_from_learner**

322 **Synopsis**

```
323     comments_from_learner :  
324         array(0..999) of con_ob_con_comment,  
325         // array dimension is an SPM
```

326 **Description**

327 The `comments_from_learner` data element contains text from the learner.

328 Subclause 6.2.1 defines `con_ob_con_comment`.

329 NOTE 1—The value of this data element is intended to provide feedback about the content object
330 or the learning experience with the content object from a specific user. Using this data element
331 for other purposes may adversely affect interoperability.

332 NOTE 2—The structure of the text is not specified.

333 NOTE 3—This Standard does not specify the mechanism for collecting comments.

334 **6.1.2 comments_from_lms**

335 **Synopsis**

```
336     comments_from_lms :  
337         array(0..99) of con_ob_con_comment,  
338         // array dimension is an SPM
```

339 **Description**

340 The `comments_from_lms` data element contains comments and annotations intended to be
341 made available to the learner.

342 Subclause 6.2.1 defines `con_ob_con_comment`.

343 NOTE—This Standard does not specify the mechanism for collecting or providing comments.

344 **6.1.3 completion_status**

345 **Synopsis**

```
346     completion_status :  
347         state( completed, incomplete, not_attempted, unknown ),
```

348 **Description**

349 The `completion_status` data element indicates whether the learner has completed the
350 content object. The data element shall have one of the following permissible values. The de-
351 fault value shall be `unknown`.

- 352 – `completed`: The learner has experienced enough of the content object to con-
 353 sider it completed.
 354 – `incomplete`: The learner has not experienced enough of the content object to
 355 consider it completed.
 356 – `not_attempted`: The learner is considered not to have used the content ob-
 357 ject in any significant way. Note: The learner has not accessed the content ob-
 358 ject, or the learner previously has accessed the content object but has experi-
 359 enced so little of it that the content object is considered to be not attempted.
 360 – `unknown`: No assertion is made.

361 NOTE—This Standard does not specify how to determine `completion_status`. It may be re-
 362 ported by a content object, determined by an RTS, determined on the basis of objectives set by an
 363 outside agent (e.g. an instructor), or by some other means.

364 **6.1.4 credit**

365 **Synopsis**

```
366        credit :
367        state( credit, no_credit ),
```

368 **Description**

369 The `credit` data element indicates whether the learner will be credited for performance
 370 (pass/fail and score) in this content object. The data element shall have one of the following
 371 permissible values. The default value shall be `credit`.

- 372 – `credit`: The learner is taking the content object for credit.
- 373 – `no_credit`: The learner is taking the content object for no credit.

374 NOTE—This Standard does not specify how to determine the value of `credit`.

375 **6.1.5 data_model_version**

376 **Synopsis**

```
377        data_model_version :
378        characterstring(iso-10646-1), // SPM: 255
```

379 **Description**

380 The `data_model_version` data element is the version of the data model for an instance.
 381 The value shall consist of a period-delimited string containing major and minor release values
 382 as whole numbers; for example, "1.0". Any characters appearing after the minor release
 383 value shall be separated from the minor release value by a period ("."). The syntax and se-
 384 mantics of any characters following the minor release value is not specified by this Standard.

385 The major version number shall be "1". The minor version number shall be "0". Therefore,
 386 the first three characters of the value of the `version` attribute shall be "1.0".

387 An implementation may append additional characters to the value of the data element,
388 in which case the first four characters shall be "1.0.0".

389 **6.1.6 entry**

390 **Synopsis**

```
391     entry :
392         state( ab_initio, resume, _nil_ ),
```

393 **Description**

394 The `entry` data element contains information from the RTS that asserts whether the learner
395 has previously accessed the content object. The data element shall have one of the following
396 permissible values. The default value shall be `_nil_`.

- 397 – `ab_initio`: Indicates that the RTS asserts that the learner has never accessed
398 the content object.
- 399 – `resume`: Indicates that (1) the learner has previously accessed the content ob-
400 ject, and (2) upon exiting, the `exit` data element had the value `suspend` (see
401 subclause 6.1.7).
- 402 – `_nil_`: Indicates all other conditions. The RTS has no knowledge of previous
403 access or does not wish to indicate a specific entry condition.

404 NOTE 1—This Standard does not specify the scope of the assertion (e.g., course, enrollment,
405 global).

406 NOTE 2—If the value for `entry` is `resume`, it indicates that either `location` or `suspend_data`
407 may contain data stored in a previous learner session that is relevant to resuming the learner ses-
408 sion. (See subclauses 6.1.14 and 6.1.23, respectively.)

409 **6.1.7 exit**

410 **Synopsis**

```
411     exit :
412         state( timeout, suspend, logout, _nil_ ),
```

413 **Description**

414 The `exit` data element indicates how or why the learner left the content object. The data ele-
415 ment shall have one of the following permissible values. The default value shall be `_nil_`.

- 416 – `time_out`: The content object terminated because the time limit specified by
417 `max_time_allowed` had been exceeded. (See subclause 6.1.15.)
- 418 – `suspend`: The learner exited the content object with the intent of returning to
419 it at the point of exit.
- 420 – `logout`: The content object signaled a desire to terminate the entire learning
421 activity of which the content object is a part.
- 422 – `_nil_`: The content object exited normally.

423 **6.1.8 interaction**424 **Synopsis**

```

425     type interaction =
426         record
427         (
428             id :
429                 con_ob_con_long_identifier,
430             type :
431                 state( true_false, multiple_choice, fill_in,
432                     long_fill_in, matching, sequencing,
433                     performance, likert, numeric, other ),
434             objectives_id :
435                 array(0..9) of con_ob_con_long_identifier),
436                 // array dimension is an SPM
437             timestamp :
438                 time(second,10,2),
439             correct_responses :
440                 choice
441                 (
442                     state( true_false, multiple_choice, fill_in,
443                         long_fill_in, likert, matching,
444                         performance, sequencing, numeric, other )
445                 )
446             weighting :
447                 real(10,7),
448             learner_response :
449                 choice
450                 (
451                     state( true_false, multiple_choice, fill_in,
452                         long_fill_in, likert, matching,
453                         performance, sequencing, numeric, other )
454                 )
455             result :
456                 state( correct, wrong, unanticipated, neutral,
457                     real(10,7) ),
458             latency :
459                 timeinterval(second,10,2),
460             description :
461                 con_ob_com_loacalized_string(255),
462                 // parameter value is an SPM
463         ),

```

464 **Description**

465 The interaction type is a set of learner responses that may be passed from the content ob-
466 ject to the RTS. The data element defines information that is recorded during an interaction for
467 the purpose of measurement or assessment. Instances of this type shall include interac-
468 tion.id; all other components are optional.

469 The components of *interaction* are defined in subclauses 6.1.8.1 through 6.1.8.10.

470 NOTE 1—Interactions are intended to be responses to individual questions or tasks that the con-
471 tent developer wants to record. This Standard does not specify how interaction data is to be re-
472 corded, used, or interpreted.

473 NOTE 2—The interactions data model includes data elements that correspond to a limited set of
474 interaction types, but it does not support logging of discrete learner events.

475 NOTE 3—This Standard does not specify how interactions are presented or rendered.

476 NOTE 4—This Standard does not specify how interactions are grouped in a question (i.e., one
477 interaction or multiple interactions per question).

478 NOTE 5—The primary intent of interaction data is to communicate data about the status of an
479 interaction object such as a test item, a simulation, or other interactive feature of the content ob-
480 ject. Interaction data may also be used to communicate interaction events as they occur, but in
481 that case only the data elements that carry information specific to the event should be communi-
482 cated.

483 **6.1.8.1 id**

484 **Synopsis**

```
485     id :
486         con_ob_con_long_identifier,
```

487 **Description**

488 The *id* data element is a label for the interaction. This label shall be unique at least within the
489 scope of the content object.

490 Subclause 6.2.3 defines *con_ob_con_long_identifier*.

491 NOTE—This Standard does not specify how *ids* are created, assigned, and resolved.

492 **6.1.8.2 type**

493 **Synopsis**

```
494     type :
495         state( true_false, multiple_choice, fill_in, long_fill_in,
496             matching, performance, sequencing, likert, numeric,
497             other),
```

498 **Description**

499 The *type* data element indicates which category of interaction is recorded and how the inter-
500 action response should be interpreted. The data element shall have one of the following per-
501 missible values. The content developer may create extended types using "other".

- 502 – *true_false*: The interaction has two possible responses. Examples: "True or
503 False", "Yes or No", "Agree or Disagree".
- 504 – *multiple_choice*: The interaction has a set of two or more predefined re-
505 sponses from which the learner may select.

- 506 – `fill_in`: The interaction requires the learner to supply a short response in
507 the form of a string of characters. Notes: Typically, the correct response con-
508 sists of part of a word, one word, or a few words.
- 509 – `long_fill_in`: The interaction requires the learner to supply a response in
510 the form of a long string of characters. Notes: (1) Typically, the correct re-
511 sponse is a sentence, paragraph, or short composition, but long composition
512 forms are also possible. (2) Typically, the interaction is presented as an ex-
513 amination statement the learner must analyze and respond to by creating a
514 written answer of a specified length, such as a short or long essay.
- 515 – `matching`: The interaction consists of two sets of items. Members of the first
516 set are related to zero or more members of the second set. Responding to the
517 interaction requires the learner to indicate matches between members of the
518 first set and members of the second set.
- 519 – `performance`: The interaction requires the learner to perform a task that re-
520 quires multiple steps. Example: The task is a simulation for the changing of a
521 sparkplug on an automobile engine involving six steps (1) pull off the rubber
522 boot from the plug, (2) unscrew the spark plug, (3) gap the replacement plug
523 to a specific dimension, (4) screw in the replacement, (5) torque the plug us-
524 ing a torque wrench set to 12 foot pounds, (6) push the boot back on.
- 525 – `sequencing`: The interaction requires the learner to identify a logical order
526 for the members of a list. Example: The learner may be asked to place a series
527 of events in chronological order or to rank a group of items by the order of
528 their importance.
- 529 – `likert`: The interaction asks the learner to select from a discrete set of
530 choices on a scale. Note: This Standard does not specify the number or re-
531 sponses, the scale, nor the meaning of the scale. Example: A typical response
532 scale has five choices ranging from "strongly disagree" to "strongly agree".
- 533 – `numeric`: The interaction requires a numeric response from the learner. The
534 response is a simple number with an optional decimal point.
- 535 – `other`: Any other type of interaction not defined by this Standard. The se-
536 mantics and structure of the `correct_responses` and `learner_response`
537 data element values are not defined by this Standard when the interaction type
538 is `other`. (See subclauses 6.1.8.5 and 6.1.8.7.) Note: When the interaction
539 type is `other`, information identifying this extended type should be embedded
540 in the `correct_responses` and `learner_response` data element values.
541 For example, this may take the form of a prefix in the string used to commu-
542 nicate those values.

543 **6.1.8.3 objectives_id**

544 **Synopsis**

```
545     objectives_id :
546         array(0..9) of con_ob_con_long_identifier,
547         // array dimension is an SPM
```

548 **Description**

549 The `objectives_id` data element is a label for objectives (see subclause 6.1.18) associated
550 with the interaction. The label shall be unique at least within the scope of the content object.

551 Subclause 6.2.3 defines `con_ob_con_long_identifier`.

552 NOTE—This Standard does not specify how ids are created, assigned, and resolved.

553 **6.1.8.4 timestamp**554 **Synopsis**

```
555     timestamp :
556         time(second,10,2),
```

557 **Description**

558 The `timestamp` data element is the time at which the interaction was first available for
559 response to the learner.

560 NOTE 1—This is wall-clock time.

561 NOTE 2—If several interactions are presented at the same time, they have the same `timestamp`
562 value. If an interaction was never available for response, such as an interaction that is not used in
563 an adaptive test, no `timestamp` value is available for that interaction.

564 NOTE 3—If a `timestamp` value is available for an interaction but no learner response data is
565 available, it should be assumed that the interaction has been available to the learner but the
566 learner did not respond.

567 **6.1.8.5 correct_responses**568 **Synopsis**

```
569     correct_responses :
570         choice
571         (
572             state( true_false, multiple_choice, fill_in, long_fill_in,
573                 likert, matching, performance, sequencing, numeric,
574                 other )
575         )
576     of
577     (
578         true_false :
579             state( true, false ),
580         multiple_choice :
581             bag of set of con_ob_com_short_identifier,
582             // bag SPM: 10, set SPM: 36
583         fill_in :
584             bag of record
585             (
586                 case_matters :
```

```

587         boolean,
588     match_text :
589         con_ob_con_localized_string(255),
590         //parameter value is an SPM
591     ), // SPM: 5
592 long_fill_in :
593     bag of record
594     (
595         case_matters :
596             boolean,
597         match_text :
598             con_ob_com_localized_string(4096),
599             //parameter value is an SPM
600     ), // SPM: 5
601 likert :
602     con_ob_com_short_identifier,
603 matching :
604     bag of bag of record
605     (
606         source :
607             con_ob_con_short_identifier,
608         target :
609             con_ob_con_short_identifier,
610     ), // outer bag SPM: 5, inner bag SPM: 36
611 performance :
612     bag of record
613     (
614         order_matters :
615             boolean,
616         answers :
617             array(0..254) of record
618             // array dimension is an SPM
619             (
620                 step_name :
621                     con_ob_con_short_identifier,
622                 step_answer :
623                     choice
624                     (
625                         state( literal, numeric )
626                     )
627                 of
628                 (
629                     literal :
630                         characterstring(iso-10646-1),
631                         // SPM: 255
632                     numeric :
633                         record
634                         (
635                             min :
636                                 real(10,7),
637                             max :

```

```

638                                     real(10,7),
639                                     ),
640                                     ),
641                                     ),
642                                     ), // SPM: 5
643 sequencing :
644     bag of array(0..35) of con_ob_con_short_identifier,
645     // bag SPM: 5, array dimension is an SPM
646 numeric :
647     record
648     (
649         min :
650             real(10,7),
651         max :
652             real(10,7),
653     ),
654 other :
655     characterstring(iso-1046-1), // SPM: 4096
656 ),

```

657 **Description**

658 The `correct_responses` data element indicates the correct response to the interaction. The
659 data element shall have one of ten possible variants that shall match the conditions described
660 below.

661 Several response types support more than one correct response. For these types, a list (bag) of
662 correct response(s) is provided. A correct response may require multiple inputs. For these re-
663 sponses, a list (bag or array depending on whether the order of input matters) of input(s) is
664 provided.

665 The content developer may create extended types using "other".

- 666 – `true_false`: A state that contains the values `true` and `false`. The state
667 `true` means true or a synonym of true (e.g. agree, yes, richtig). The state
668 `false` means false or a synonym of false (e.g. disagree, no, falsch).
- 669 – `multiple_choice`: A bag of sets short identifiers. The bag contains one or
670 more sets, any of which satisfies the requirement for a correct response. Each
671 set contains one or more short identifiers, all of which are required for a cor-
672 rect response. Each of the short identifiers represents an expected choice. Ex-
673 ample: A single choice may be allowed: "a", or the valid combinations of
674 choices may be "a", "b", "c" and "a", "b", "d".
- 675 – `fill_in`: A bag of records. The bag contains one or more records, any of
676 which satisfies the requirement for a correct response. Each record consists of
677 a localized string and a flag. The localized string represents a correct response.
678 The flag indicates whether the case of the character string matters to judge the
679 correctness of the response. If `case matters` is not specified, it is assumed
680 to be false.

- 681 – `long_fill_in`: A bag of records. The bag contains one or more records, any
682 of which satisfies the requirement for a correct response. Each record consists
683 of a localized string and a flag. The localized string represents a correct re-
684 sponse. The flag indicates whether the case of the character string matters to
685 judge the correctness of the response. If `case_matters` is not specified, it is
686 assumed to be false. Note: Although a correct response for `long_fill_in`
687 can be specified, the evaluation of a `long_fill_in` response typically in-
688 volves an interpretative process that is outside of the scope of this Standard.
- 689 – `likert`: A short identifier that matches a choice on a scale. Note: Although a
690 correct response for `likert` can be specified, `likert` interactions typically do not
691 include correct responses.
- 692 – `matching`: A bag of bags of records. The outer bag contains one or more in-
693 ner bags, any of which satisfies the requirement for a correct response. Each
694 inner bag consists of one or more records, all of which are required for a cor-
695 rect response. Each of the records is a pair of short identifiers representing an
696 expected matching input. Each of the inputs consists of a source and a target
697 (or stem and alternative). Each source and each target shall be represented by
698 a different short identifier.
- 699 – `performance`: A bag of records. The bag contains one or more records, any
700 of which satisfies the requirement for a correct response. Each record consists
701 of a flag and an array. The array represents a set of inputs for a correct re-
702 sponse. The flag indicates whether the order of the inputs matters for a correct
703 response. If the order matters, the learner must provide each input in the exact
704 order of the array. If the order does not matter, the learner may provide inputs
705 in any order. If `order_matters` is not specified, it is assumed to be true.
706 Each input consists of a name and either a single literal value or a numeric
707 range. If an input is expressed as a literal value, the interaction implementa-
708 tion determines how to use the value to evaluate the corresponding response.
709 If an input is expressed as a numeric range, the learner must provide an input
710 that falls within that range.
- 711 – `sequencing`: A bag of arrays of short identifiers. The bag contains one or
712 more arrays, any of which satisfies the requirement for a correct response.
713 Each array represents a sequence of short identifiers for a correct response.
714 Each short identifier identifies one element that shall be available to be se-
715 quenced when the interaction is presented to the learner. Each array shall con-
716 tain a different sequence of short identifiers. Different arrays may contain dif-
717 ferent sets of short identifiers.
- 718 – `numeric`: Two numbers with optional decimal points. The numbers may be
719 used to express a range for the correct response. If the numbers are the same,
720 then a single number is specified as the correct response.
- 721 – `other`: A string defined by the specific "other" interaction type. The content
722 of this string is not defined by this Standard.

723 NOTE 1—The `correct_responses` data element is a structured mechanism for identifying the
724 correct learner response or responses relating to each of the types of interactions described in
725 subclause 6.1.8.2. The determination of correctness is an implementation-defined feature of the
726 content object.

727 NOTE 2—The interaction type, as defined in subclause 6.1.8.2, has to be known in order to select
728 the appropriate variant.

729 **6.1.8.6 weighting**

730 **Synopsis**

```
731     weighting :
732         real(10,7),
```

733 **Description**

734 The `weighting` data element is the weight used by the content object to compute a total
735 score.

736 NOTE—Interaction weights typically are used to explain the effect of an interaction on a total
737 score but are not intended to be used by systems other than the content object to compute a score.

738 **6.1.8.7 learner_response**

739 **Synopsis**

```
740     learner_response :
741         choice
742         (
743             state( true_false, multiple_choice, fill_in, long_fill_in,
744                 likert, matching, performance, sequencing, numeric,
745                 other )
746         )
747     of
748     (
749         true_false :
750             state( true, false ),
751         multiple_choice :
752             bag of con_ob_con_short_identifier,
753         fill_in :
754             con_ob_con_localized_string (255),
755             //parameter value is an SPM
756         long_fill_in:
757             con_ob_con_localized_string(4096),
758             //parameter value is an SPM
759         likert :
760             con_ob_con_short_identifier,
761         matching :
762             bag of record
763             (
764                 source :
765                     con_ob_con_short_identifier,
766                 target :
767                     con_ob_con_short_identifier,
768             ), // SPM: 36
```

```

769     performance :
770         array(0..254) of record // array dimension is an SPM
771         (
772             step_name :
773                 con_ob_con_short_identifier,
774             step_answer :
775                 choice
776                 (
777                     state( literal, numeric )
778                 )
779             of
780             (
781                 literal:
782                     characterstring(iso-10646-1), // SPM: 255
783                 numeric :
784                     real(10,7),
785             ),
786         ),
787     sequencing :
788         array(0..35) of con_ob_con_short_identifier,
789         // array dimension is an SPM
790     numeric :
791         real(10,7),
792     other :
793         characterstring(iso-1046-1), // SPM: 4096
794 ),

```

795 **Description**

796 The `learner_response` data element consists of data generated by a learner when responding to an interaction or a description of the learner's action.

798 The `learner_response` data element shall have one of the ten possible variants that shall match the conditions described below. The content developer may create extended types using "other".

- 801 – `true_false`: A state that contains the values `true` and `false`. The state `true` means true or a synonym of true (e.g. agree, yes, richtig). The state `false` means false or a synonym of false (e.g. disagree, no, falsch).
- 804 – `multiple_choice`: A bag of short identifiers. If the interaction allowed only one choice, the bag contains only one short identifier. If the interaction allowed more than one choice or more than one combination of choices, the combination of choices made by the learner is represented by the short identifiers in the bag. If the learner made no choice, the bag is empty. Examples: If single choice was allowed: "a". If a combination of choices was allowed: "a", "b", "d", the order of which is insignificant.
- 811 – `fill_in`: A localized string.
- 812 – `long_fill_in`: A localized string.
- 813 – `likert`: The choice made by the learner is represented by short identifier.

- 814 – `matching`: A bag that contains zero or more pairs short identifiers. Each pair
815 represents a match made by the learner.
- 816 – `performance`: An array of responses in the order they were provided by the
817 learner in response to the interaction. Each response consists of a step name
818 and either a single literal value or a number. The step names and types of the
819 responses shall match those provided in the `correct_responses` for the in-
820 teraction, but the responses in the `learner_response` may be in a different
821 order. Because a learner may perform the same step more than once, the step
822 names of the responses may not be unique, i.e. a response may appear more
823 than once with the same value or with a different value. Example: If the per-
824 formance involves setting several valves to specific positions, the learner may
825 adjust the position of the same valve more than once in the course of the per-
826 formance. The name values pairs for the response might be "valve_1:open,
827 valve 2:closed, valve 1:closed". Notes: 1. The SPM for `learner_response`
828 is twice the size of the SPM for `correct_responses` to allow recording of
829 extra steps, as in the example above. 2. The syntax of the name-value pairs is
830 not specified by this Standard.
- 831 – `sequencing`: An array of short identifiers. The sequence determined by the
832 learner is represented by the order of the array. Each short identifier identifies
833 one element that was available to be sequenced.
- 834 – `numeric`: A number with an optional decimal point.
- 835 – `other`: A string defined by the specific "other" interaction type. The content
836 of this string is not defined by this Standard.

837 NOTE 1—The `learner_response` data element is a structured mechanism for identifying the
838 exact learner response relating to each of the types of interactions described in subclause 6.1.8.2.
839 The determination of correctness is an implementation-defined feature of the content object.

840 NOTE 2—The interaction type, as defined in subclause 6.1.8.2, has to be known in order to select
841 the appropriate variant.

842 **6.1.8.8 result**

843 **Synopsis**

```
844       result :
845           state( correct, incorrect, unanticipated, neutral,
846                 real(10,7) ),
```

847 **Description**

848 The `result` data element is a judgment of the correctness of the learner response. The data
849 element shall have one of the following permissible values:

- 850 – `correct`: The learner response was correct.
- 851 – `incorrect`: The learner response was incorrect.
- 852 – `unanticipated`: The learner response was not expected by the content ob-
853 ject.
- 854 – `neutral`: The learner response was neither correct nor incorrect.

855 – `real(10,7)`: A real number.

856 NOTE 1—This Standard does not specify where or how the value of `result` is determined.

857 NOTE 2—The state `real(10,7)` is included to allow a numeric judgment of the correctness of
858 the learner response.

859 **6.1.8.9 latency**

860 **Synopsis**

```
861       latency :  
862           timeinterval(second,10,2),
```

863 The `latency` data element is the time elapsed between the time the interaction was first avail-
864 able for response to the learner and the time of the first response.

865 NOTE—The `latency` information is not available for an interaction if the learner did not re-
866 spond. The `latency` is, in effect, the time difference between the `timestamp` of the interaction
867 and the time of the first response. If several interactions have the same `timestamp` because they
868 became available for response at the same time, the `latency` recorded for each interaction can be
869 used to determine the order in which the learner responded to these interactions.

870 **6.1.8.10 description**

871 **Synopsis**

```
872       description :  
873           con_ob_com_loacalized_string(255),  
874           // parameter value is an SPM
```

875 **Description**

876 The description data element is a brief informative description of the interaction.

877 **6.1.9 launch_data**

878 **Synopsis**

```
879       launch_data :  
880           characterstring(iso-10646-1), // SPM: 4096
```

881 **Description**

882 The `launch_data` data element lets the RTS provide data specific to a content object that the
883 content object can use for initialization. The value of this data element is not specified.

884 NOTE—The allowable values for this data element are defined by the implementer of the content
885 object. Typically, the documentation for the content object would specify what data can or has to
886 be provided.

887 **6.1.10 learner_id**888 **Synopsis**

```
889     learner_id :
890         con_ob_con_long_identifier,
```

891 **Description**

892 The `learner_id` data element identifies the learner on behalf of whom this content object
893 instance has been launched by the RTS. The label shall be unique at least within the scope of
894 the content object.

895 Subclause 6.2.3 defines `con_ob_con_long_identifier`.

896 NOTE—This Standard does not specify how ids are created, assigned, and resolved.

897 **6.1.11 learner_name**898 **Synopsis**

```
899     learner_name :
900         con_ob_con_localized_string(255),
901         //parameter value is an SPM
```

902 **Description**

903 The `learner_name` data element is the name of the learner.

904 Subclause 6.2.2 defines `con_ob_con_localized_string`.

905 NOTE—This Standard does not specify how learner names are created, assigned, and resolved.

906 **6.1.12 learner_preference_data**907 **Synopsis**

```
908     type learner_preference_data =
909         record
910             (
911                 audio :
912                     real(10,7), range(0..*),
913                 language :
914                     characterstring(iso-646), // SPM: 255
915                 speed :
916                     real(10,7), range(0..*),
917                 text :
918                     state( -1, 0, 1),
919             ),
```

920 **Description**

921 The `learner_preference` type specifies learner preferences associated with the learner's
922 use of the content object.

923 The components of `learner_preference` are defined in subclauses 6.1.12.1 through
924 6.1.12.4.

925 NOTE—This Standard does not specify whether the content object, the RTS, or both have the
926 ability to set or interpret learner preferences.

927 **6.1.12.1 audio**928 **Synopsis**

```
929     audio :
930         real(10,7), range(0..*),
```

931 **Description**

932 The `audio` data element is a multiplier value that specifies an intended change in perceived
933 audio level, with 1 meaning "no change". For example, the value 0 specifies infinite attenua-
934 tion, the value 0.5 specifies an attenuation of 10 decibels, and the value 2 specifies an attenua-
935 tion of 10db.

936 **6.1.12.2 language**937 **Synopsis**

```
938     language :
939         characterstring(iso-646), // SPM: 255
```

940 **Description**

941 The `language` data element is the learner's preferred language for content objects with multi-
942 lingual capability. The format is

```
943     langcode ("-" subcode)*
```

944 where "`langcode`" is required and multiple, optional, hyphen-prefixed subcodes may follow.

945 The following rules apply to "`langcode`" :

- 946 – 2-letter codes are defined by ISO 639.
- 947 – 3-letter codes are defined by ISO 639-2.
- 948 – The value "i" is reserved for registrations defined by the Internet Assigned
949 Numbers Authority (IANA).
- 950 – The value "x" is reserved for private use.

951 The following rules apply to the first "`subcode`" :

- 952 – 2-letter subcodes are ISO 3166–1 alpha-2 country codes.
 953 – Subcodes of from 3 to 8 letters are registered with IANA.
 954 Additional subcodes are unspecified.

955 NOTE—The language code is normally given in lower case and the subcodes (if any) in upper
 956 case. However, the values are case insensitive.

957 **Examples**

958 "en-GB"
 959 "de"
 960 "fr-CA"
 961 "it"
 962 "grc" (ancient greek, until 1453)
 963 "en-US-philadelphia"
 964 "eng-GB-cockney"
 965 "map-PG-buin" (Austronesian - Papua New Guinea buin)
 966 "gem-US-pennsylvania"

967 **6.1.12.3 speed**

968 **Synopsis**

969 speed :
 970 real(10,7), range(0..*),

971 **Description**

972 The `speed` data element is the learner's preferred relative pace of content delivery. The value
 973 is a multiplier of the default pace. For example, 2 is twice as fast as the default delivery pace
 974 and 0.5 is one half the default pace. The default value shall be 1.

975 **6.1.12.4 text**

976 **Synopsis**

977 text :
 978 state(-1, 0, 1),

979 **Description**

980 The `text` data element specifies whether captioning text is displayed with or in the place of
 981 audio. The data element shall have one of the following permissible values. The default value
 982 shall be 0.

- 983 – -1: Captioning is off, and text corresponding to audio is not displayed.
 984 – 0: The current default captioning setting.
 985 – 1: Captioning is on, and text corresponding to audio is displayed.

986 **6.1.13 lesson_status**987 **Synopsis**

```
988     lesson_status :
989         state( passed, failed, completed, incomplete, browsed,
990             not_attempted ),
```

991 **Description**

992 The `lesson_status` data element is an obsolescent feature. New implementations should
993 use `completion_status` and `success_status`. (See subclauses 6.1.3 and 6.1.22.)

994 The `lesson_status` data element indicates whether the learner has attempted, completed,
995 passed, failed, or browsed the associated content object. The data element shall have one of
996 the following permissible values:

- 997 – `passed`: The learner has satisfied the requirements to pass the content object.
- 998 – `failed`: The learner has not satisfied the requirements to pass the content ob-
999 ject.
- 1000 – `completed`: The learner has satisfied the requirements to complete the con-
1001 tent object.
- 1002 – `incomplete`: The learner has not satisfied the requirements to complete the
1003 content object.
- 1004 – `browsed`: The learner has accessed the content object with a mode of browse
1005 or elected to browse while in the content object after a normal launch.
- 1006 – `not_attempted`: The learner has not accessed the content object, or the
1007 learner previously has accessed the content object but has experienced so little
1008 of it that it is considered it to be not attempted.

1009 NOTE—This Standard does not specify how to determine `lesson_status`. It may be reported
1010 by a content object, determined by an RTS by comparing scores to mastery scores, determined on
1011 the basis of objectives set by an outside agent (e.g. an instructor), or by some other means.

1012 **6.1.14 location**1013 **Synopsis**

```
1014     location :
1015         characterstring(iso-10646-1), // SPM: 1000,
```

1016 **Description**

1017 The `location` data element is a location in the content object. Its value and meaning are de-
1018 fined by the content object and are not specified by this Standard. If there is no preferred initial
1019 location, `location` shall equal `_nil_`.

1020 NOTE—The RTS should not interpret or change this data. If a content object communicates a
1021 `location` on exit, this data element provides support for a mechanism that lets the learner return
1022 to the content object at the same place he or she left it. This data element can identify the learner's

1023 exit point with a value that is meaningful to the content object only, and that location information
 1024 can be used by the content object as an entry point the next time the learner enters the content
 1025 object. This data element also can be used by the content object to communicate its location to the
 1026 RTS on an ongoing basis. Example: An RTS may be able to use this information as an opaque
 1027 key to create bookmarks, or to synchronize reference materials or annotations with the `location`
 1028 reported by the content.

1029 **6.1.15 max_time_allowed**

1030 **Synopsis**

```
1031     max_time_allowed :
1032         timeinterval(second,10,2),
```

1033 **Description**

1034 The `max_time_allowed` data element is the amount of accumulated time the learner is al-
 1035 lowed to use a content object. (See `time_limit_action` in subclause 6.1.24 for the content
 1036 object's expected response to exceeding the limit.)

1037 **6.1.16 mode**

1038 **Synopsis**

```
1039     mode :
1040         state( browse, normal, review ),
```

1041 **Description**

1042 The `mode` data element identifies one of three possible modes in which a content object may
 1043 be presented to a learner. The data element shall have one of the following permissible values.
 1044 The default value shall be `normal`.

- 1045 – `browse`: The content object is presented without the intent of recording status
 1046 (i.e., passed, completed, failed).
- 1047 – `normal`: The content object is presented with the intent of recording status
 1048 (i.e., passed, completed, failed).
- 1049 – `review`: The content object has a recorded status (i.e., passed, completed,
 1050 failed) and is presented without the intent of updating its recorded status.

1051 **6.1.17 objective**

1052 **Synopsis**

```
1053     type objective =
1054         record
1055         (
1056             id :
1057                 con_ob_con_long_identifier,
```

```

1058     score :
1059         con_ob_con_score,
1060     status :
1061         state( passed, completed, failed, incomplete, browsed,
1062             not_attempted ),
1063     completion_status :
1064         state( completed, incomplete, not_attempted,
1065             unknown ),
1066     success_status :
1067         state( passed, failed, unknown ),
1068 ),

```

1069 **Description**

1070 The `objective` type specifies learning or performance objectives associated with a content
 1071 object. Instances of this type shall include `objective.id`; all other components are optional.

1072 The components of `objective` are defined in subclauses 6.1.17.1 through 6.1.17.5.

1073 NOTE 1—Information about objectives may come from a content object, from an RTS, or from
 1074 some other source.

1075 NOTE 2—This Standard does not define any relationship between objectives and the content ob-
 1076 ject `completion_status`, `lesson_status`, `score`, or `success_status` (see subclauses 6.1.3, 6.1.13,
 1077 6.1.20, and 6.1.22, respectively).

1078 **6.1.17.1 id**

1079 **Synopsis**

```

1080     id :
1081         con_ob_con_long_identifier,

```

1082 **Description**

1083 The `id` data element is a label for the objective. This label shall be unique at least within the
 1084 scope of the content object.

1085 Subclause 6.2.2 describes `con_ob_con_long_identifier`.

1086 NOTE—This Standard does not specify how ids are created, assigned, and resolved.

1087 **6.1.17.2 score**

1088 **Synopsis**

```

1089     score :
1090         con_ob_con_score,

```

1091 **Description**

1092 The `score` data element is the score achieved by the learner for the objective.

1093 Subclause 6.2.4 defines `con_ob_con_score`.

1094 NOTE—The score for an objective may be determined by the content object or may have been
1095 determined by another content object or any other entity, such as posting of a score by an admin-
1096 istrator.

1097 **6.1.17.3 status**

1098 **Synopsis**

```
1099     status :
1100         state( passed, completed, failed, incomplete, browsed,
1101             not_attempted ),
```

1102 **Description**

1103 The `status` data element is an obsolescent feature. New implementations should use
1104 `completion_status` and `success_status`. (See subclauses 6.1.3 and 6.1.22.)

1105 The `status` data element indicates whether the learner has engaged with that portion of the
1106 content object related to the objective and, if so, whether the learner has demonstrated mastery
1107 of the objective. The data element shall have one of the following permissible values:

- 1108 – `passed`: The objective was passed.
- 1109 – `completed`: All segments of the content object related to the objective were
1110 accessed. The objective may or may not have passed.
- 1111 – `failed`: The objective was failed.
- 1112 – `incomplete`: Not all segments of the content object related to this objective
1113 were accessed.
- 1114 – `not_attempted`: No segment of the content object related to this objective
1115 was accessed.
- 1116 – `browsed`: No specific status information for this objective is available be-
1117 cause the content object related to this objective was launched with a mode of
1118 `browse`.

1119 NOTE—Status may be provided by the content object, by an RTS, or by some other means.

1120 **6.1.17.4 completion_status**

1121 **Synopsis**

```
1122     completion_status :
1123         state( completed, incomplete, not_attempted, unknown ),
```

1124 **Description**

1125 The `completion_status` data element indicates whether the learner has completed the
1126 associated objective. The data element shall have one of the following permissible values. The
1127 default value shall be `unknown`.

- 1128 – `completed`: The learner has experienced enough of the objective for it to be
 1129 considered completed.
 1130 – `incomplete`: The learner has started the objective but did not finish.
 1131 – `not_attempted`: The learner has not accessed the objective, or the learner
 1132 previously has accessed the objective but has experienced so little of it that the
 1133 objective is considered it to be not attempted.
 1134 – `unknown`: No assertion is made.

1135 NOTE—This Standard does not specify how to determine `completion_status`. It may be re-
 1136 ported by a content object, determined by an RTS, determined on the basis of objectives set by an
 1137 outside agent (e.g. an instructor), or by some other means.

1138 **6.1.17.5 success_status**

1139 **Synopsis**

```
1140     success_status :
1141         state( passed, failed, unknown ),
```

1142 **Description**

1143 The `success_status` data element indicates whether the learner has passed the associated
 1144 objective. The data element shall have one of the following permissible values. The default
 1145 value shall be `unknown`.

- 1146 – `passed`: The learner has passed the objective.
 1147 – `failed`: The learner has failed the objective.
 1148 – `unknown`: No assertion is made.

1149 NOTE—This Standard does not specify how to determine `success_status`. It may be reported
 1150 by a content object, determined by an RTS, determined on the basis of objectives set by an out-
 1151 side agent (e.g. an instructor), or by some other means.

1152 **6.1.18 raw_passing_score**

1153 **Synopsis**

```
1154     raw_passing_score :
1155         real(10,7),
```

1156 **Description**

1157 The `raw_passing_score` data element is an obsolescent feature. New implementations
 1158 should use `scaled_passing_score`. (See subclause 6.1.19.)

1159 The `raw_passing_score` data element is the raw (unscaled) passing score for a content ob-
 1160 ject.

1161 **6.1.19 scaled_passing_score**

1162 **Synopsis**

```
1163     scaled_passing_score :  
1164     real(10,7) range(-1..1),
```

1165 **Description**

1166 The `scaled_passing_score` data element is the scaled passing score for a content object.
1167 The value of the data element is scaled to fit the range -1 to 1 inclusive.

1168 NOTE 1—The value of this data element is a real number ≥ 0 and ≤ 1 .

1169 NOTE 2—If a `scaled_passing_score` is defined for the use of a content object, this is a
1170 statement that the requirements associated with the use of that content object are achieved by ob-
1171 taining a score greater than or equal to the `scaled_passing_score`. For example, if the
1172 `scaled_passing_score` for a content object is 0.85 and a learner achieves a normalized score
1173 of 0.90, a `success_status` of "passed" may be assigned to that content object for that
1174 learner. However, this Standard does not specify or require that an RTS, content object, or any
1175 other system component interpret or take action in response to a `scaled_passing_score`.

1176 NOTE 3—A scaled score range of -1 to +1 is used to allow an instructional system designer to
1177 more easily assign a penalty for an incorrect choice, such as in a flight simulation system where
1178 the learner's choice would have resulted in the loss of the aircraft and all aboard.

1179 **6.1.20 score**

1180 **Synopsis**

```
1181     score :  
1182     con_ob_con_score,
```

1183 **Description**

1184 The `score` data element is the learner's score for the content object.

1185 Subclause 6.2.4 defines `con_ob_con_score`.

1186 NOTE—Scores are intended to be performance measurements.

1187 **6.1.21 session_time**

1188 **Synopsis**

```
1189     session_time :  
1190     timeinterval(second,10,2),
```

1191 Description

1192 The `session_time` data element is the amount of time that the learner has spent in the cur-
 1193 rent learner session for this content object. If no learner session is in progress, the session time
 1194 is the time for the last learner session for this content object.

1195 NOTE—The content object determines the value of `session_time` and its accuracy. Examples:
 1196 1. The content object may not start recording session time until after it has initialized a media
 1197 segment. 2. If the learner takes a break in the learner session, the break time may or may not be
 1198 included in the reported `session_time`.

1199 6.1.22 success_status**1200 Synopsis**

```
1201     success_status :
1202         state( passed, failed, unknown ),
```

1203 Description

1204 The `success_status` data element indicates whether the learner has passed the content
 1205 object. The data element shall have one of the following permissible values. The default value
 1206 shall be `unknown`.

- 1207 – `passed`: The learner has passed the content object.
- 1208 – `failed`: The learner has failed the content object.
- 1209 – `unknown`: No assertion is made.

1210 NOTE—This Standard does not specify how to determine `success_status`. It may be reported
 1211 by a content object, determined by an RTS by comparing scores to mastery scores, determined on
 1212 the basis of objectives set by an outside agent (e.g. an instructor), or by some other means.

1213 6.1.23 suspend_data**1214 Synopsis**

```
1215     suspend_data :
1216         characterstring(iso-10646-1), // SPM: 4096
```

1217 Description

1218 The `suspend_data` data element provides information that may be created by a content ob-
 1219 ject as a result of a learner accessing or interacting with that content object. The format of the
 1220 content of this data element is unspecified.

1221 NOTE—The intent is for the content object to store data for later use in the current learner ses-
 1222 sion or a subsequent learner session between the content object and the same learner. The RTS
 1223 should not interpret or change this data.

1224

1225 **6.1.24 time_limit_action**

1226 **Synopsis**

```
1227     time_limit_action :  
1228         state( exit_message, continue_message, exit_no_message,  
1229             continue_no_message ),
```

1230 **Description**

1231 The `time_limit_action` data element indicates what the content object should do when
1232 `max_time_allowed` is exceeded. The data element shall have one of the following permis-
1233 sible values:

- 1234 – `exit_message`: The learner is forced out of the content object. The content
1235 object shall provide a message to the learner indicating that the maximum
1236 time allowed to complete the content object was exceeded.
- 1237 – `continue_message`: The learner is allowed to continue in the content ob-
1238 ject. The content object shall provide a message to the learner indicating that
1239 the maximum time allowed in the content object was exceeded.
- 1240 – `exit_no_message`: The learner is forced out of the content object with no
1241 message.
- 1242 – `continue_no_message`: Although the learner has exceeded the maximum
1243 time allowed, the learner is given no message and should not be forced out of
1244 the content object.

1245 NOTE—When a message is presented to the learner, the content object defines the content and
1246 form of the message.

1247 **6.1.25 total_time**

1248 **Synopsis**

```
1249     total_time :  
1250         timeinterval(second,10,2),
```

1251 **Description**

1252 The value of the `total_time` data element is the sum of all the learner's learner session times
1253 accumulated in the current learner attempt prior to the current learner session.

1254 **6.2 Auxiliary data types**

1255 The following data types are used in conjunction with the data elements described in sub-
1256 clause 6.1.

1257 NOTE—The use of ISO-11404 in the synopses in subclauses 6.2.1 through 6.2.5 is for descrip-
1258 tive purposes only. The use of abstract data types or record structures is not required for confor-
1259 mance.

1260 **6.2.1 con_ob_con_comment**1261 **Synopsis**

```

1262     type con_ob_con_comment =
1263         record
1264         (
1265             comment :
1266                 con_ob_con_localized_string(4096),
1267                 //parameter value is an SPM
1268             location :
1269                 characterstring(iso-10646-1), // SPM: 255
1270             date_time :
1271                 time(second,10,2),
1272         ),

```

1273 **Description**

1274 This data type describes textual input made by the learner. Instances of this data element shall
 1275 include `con_ob_com_comment.comment`.

1276 The components of `con_ob_con_comment` are defined in subclauses 6.2.1.1 through 6.2.1.2.

1277 **6.2.1.1 comment**1278 **Synopsis**

```

1279     comment :
1280         con_ob_con_localized_string(4096),

```

1281 **Description**

1282 The `comment` data element shall describe comments or annotations associated with a content
 1283 object.

1284 Subclause 6.2.2 defines `con_ob_con_localized_string`.

1285 **6.2.1.2 location**1286 **Synopsis**

```

1287     location :
1288         characterstring(iso-10646-1), // SPM: 255

```

1289 **Description**

1290 The `location` data element is the point in the content object to which the comment applies.
 1291 If the value of `location` is `_nil_`, the comment is applicable to the entire content object.

1292 **6.2.1.3 date_time**1293 **Synopsis**

```
1294     date_time :
1295         time(second,10,2),
```

1296 **Description**

1297 The `date_time` data element is the date and time at which the comment was made or
 1298 changed. Implementation shall support, minimally, time periods in the range January 1, 1970
 1299 through January 1, 2038.

1300 NOTE 1—The time is wall-clock time.

1301 NOTE 2—The upper limit is specified as January 1, 2038 to avoid the need to support 64-bit date
 1302 and time formats. Future editions of this Standard may extend the upper limit.

1303 **6.2.2 con_ob_con_localized_string**1304 **Synopsis**

```
1305     type con_ob_con_localized_string(length) =
1306         record
1307         (
1308             language:
1309                 characterstring(iso-646), // SPM: 255
1310             string:
1311                 characterstring(iso-10646-1),
1312                 // SPM: The length parameter
1313         ),
```

1314 **Description**

1315 The following components define this data type.

1316 – `language`: The language of the localized string. The format is

```
1317     langcode ("-" subcode)*
```

1318 where "`langcode`" is required and multiple, optional, hyphen-prefixed subcodes
 1319 may follow.

1320 The following rules apply to "`langcode`" :

- 1321 – 2-letter codes are defined by ISO 639.
- 1322 – 3-letter codes are defined by ISO 639-2.
- 1323 – The value "i" is reserved for registrations defined by IANA.
- 1324 – The value "x" is reserved for private use.

1325 The following rules apply to the first "`subcode`" :

- 1326 – 2-letter subcodes are ISO 3166-1 alpha-2 country codes.

1327 – Subcodes of from 3 to 8 letters are registered with
1328 IANA.

1329 Additional subcodes are unspecified.

1330 – `string`: The localized string itself.

1331 NOTE—The language code is normally given in lower case and the subcodes (if any) in upper
1332 case. However, the values are case insensitive.

1333 **Examples**

1334 The following are three examples of localized strings: "Information Technology" (in
1335 French), "localization" (in British English), and "xxx" (in Japanese hiragana).

```
1336     ( "fr", "Technologies de l'information" ),
1337     ( "en-GB", "localisation" ),
1338     ( "jp-JP-jisx208", "xxx" ),
```

1339 **6.2.3 con_ob_con_long_identifier**

1340 **Synopsis**

```
1341     type con_ob_con_long_identifier =
1342         characterstring(iso-10646-1), // SPM: 4096
```

1343 **Description**

1344 This data type is an identifier (a label) associated with an object that is intended to be unique
1345 within the context of usage of the object. The character string shall conform the syntax for
1346 Uniform Resource Identifiers (URI) as defined by RFC 2396.

1347 NOTE—It is recommended that the URI be a globally unique identifier in the form of a Uni-
1348 form Resource Name (URN) (see RFC 2141 [A5]).

1349 **6.2.4 con_ob_con_score**

1350 **Synopsis**

```
1351     type con_ob_con_score =
1352         record
1353         (
1354             raw :
1355                 real(10,7),
1356             min :
1357                 real(10,7),
1358             max :
1359                 real(10,7),
1360             scaled :
1361                 real(10,7) range(-1..1),
1362         ),
```

1363 Description

1364 This data type describes scoring information.

1365 The components of `con_ob_con_score` are defined in subclauses 6.2.4.1 through 6.2.4.4.

1366 6.2.4.1 raw**1367 Synopsis**

```
1368     raw :  
1369     real(10,7),
```

1370 Description

1371 The `raw` data element is a number that reflects the performance of the learner relative to the
1372 range bounded by the values of `min` and `max`.

1373 NOTE—A `raw` score is not necessarily an unprocessed score. Example: An unprocessed score
1374 might be converted to a percentage score as a decimal value. That is, the learner achieved a score
1375 of 3 out of 4 possible, which is converted to a `raw` score of 0.75 with `min` equal to zero and `max`
1376 equal to one.

1377 6.2.4.2 min**1378 Synopsis**

```
1379     min :  
1380     real(10,7),
```

1381 Description

1382 The `min` data element is the minimum value in the range for the `raw` score.

1383 6.2.4.3 max**1384 Synopsis**

```
1385     max :  
1386     real(10,7),
```

1387 Description

1388 The `max` data element is the maximum value in the range for the `raw` score.

1389 **6.2.4.4 scaled**1390 **Synopsis**

```
1391     scaled :  
1392     real(10,7) range(-1..1),
```

1393 **Description**

1394 The `scaled` data element is a number that reflects the performance of the learner. The value
1395 of the data element is scaled to fit the range -1 to 1 inclusive.

1396 NOTE—A scaled score range of -1 to +1 is used to allow an instructional system designer to
1397 more easily assign a penalty for an incorrect choice, such as in a flight simulation system where
1398 the learner's choice would have resulted in the loss of the aircraft and all aboard.

1399 **6.2.5 con_ob_com_short_identifier**1400 **Synopsis**

```
1401     type con_ob_com_short_identifier :  
1402     characterstring(iso-10646-1), // SPM 255
```

1403 **Description**

1404 This data type is an identifier (a label). The character string shall conform to the syntax de-
1405 fined by RFC 2396.

1406 **Annex A**

1407 (informative)

1408 **Bibliography**

1409 [A1] AICC CMI001, CMI Guidelines For Interoperability, Version 3.4, October 2000.

1410 [A2] IEEE 1484.11.2, Draft Standard for Learning Technology—ECMAScript Application
1411 Programming Interface for Content to Runtime Services Communication.

1412 [A3] IEEE 100, The Authoritative Dictionary of IEEE Standards Terms, Seventh Edition.

1413 [A4] ISO 8601:2000, Data elements and interchange formats—Information interchange—
1414 Representation of dates and times.

1415 [A5] RFC 2141, "URN Syntax," Network Working Group, May 1997.

1416 **Annex B**

1417 (informative)

1418 **Understanding ISO/IEC 11404:1996 data type defini-** 1419 **tions used in this Standard.**

1420 **B.1 Real data type**

1421 `real(10,7)` denotes a real data type with values that have precision to 10^{-7} (i.e., 0.0000001).

1422 .For example, according to this type definition

- 1423 – 5550.000001 and 5550.000002 are different values;
- 1424 – 5550.000000001 and 5550.0 may evaluate to the same value, because the dif-
 1425 ference of 0.000000001 is too small to be accounted for according to the pre-
 1426 cision requirement of the type definition;
- 1427 – 5550.0 and 5550.000000 are the same value; and
- 1428 – 5550.0 and 5550 evaluate to the same value.

1429 **B.2 Time interval data type**

1430 `timeinterval(second,10,2)` denotes that the value for the data element
 1431 `timeinterval` is a number expressed as a real data type with values that are accurate to one
 1432 hundredths of a second.

1433 For example, according to this type definition

- 1434 – a duration of exactly one hour is expressed with the value 3600;
- 1435 – a duration of 2.5 seconds is expressed with the value 2.5; and
- 1436 – this Standard does not require implementations to distinguish between a dura-
 1437 tion of 2.000 seconds and a duration of 2.001 seconds because the difference
 1438 of 0.001 seconds is smaller than the precision requirements for this datatype..

1439 A binding may use a standard string representation for the time interval value. For example,
 1440 "PT1H30M" (1 hour and 30 minutes), where the format is defined by the following pattern:

1441 `P[yY][mM][dD][T[hH][nM][s[.s]S]]` where

- 1442 – `y` = number of years (integer, ≥ 0 , not restricted);
- 1443 – `m` = number of months (integer, ≥ 0 , not restricted, e.g., > 12 is acceptable);
- 1444 – `d` = number of days (integer, ≥ 0 , not restricted, e.g., > 31 is acceptable);
- 1445 – `h` = number of hours (integer, ≥ 0 , not restricted, e.g., > 23 is acceptable);
- 1446 – `n` = number of minutes (integer, ≥ 0 , not restricted, e.g., > 59 is acceptable);
- 1447 and

1448 – *s* = number of seconds or fraction of seconds (real or integer, ≥ 0 , not re-
1449 stricted, e.g., > 59 is acceptable).

1450 The character literal designators "P", "Y", "M", "D", "T", "H", "M", "S" must appear if the
1451 corresponding nonzero value is present.

1452 This Standard does not specify a binding, and other bindings are possible.

1453 **B.3 Time and date data type (time stamp)**

1454 `time(second,10,2)` denotes that the value for time is a number expressed as a real data
1455 type with values that are accurate to one hundredths of a second. The number of seconds in the
1456 time value is the number of seconds since 00:00 on January 1, 1970. Note that this data type
1457 does not represent all possible times and dates, but only the time and date values in the range
1458 January 1, 1970 through January 1, 2038.

1459 A binding may use a standard string representation for the time and date value. For example,
1460 "1997-07-16T19:20:30+01:00" (July 16th, 1997, 30 seconds past 7.20 p.m. with a time
1461 offset of 1 hour with respect to UTC), where the format is defined by the following pattern:

1462 YYYY[-MM[-DD[Thh[:mm[:ss[.s[TZD]]]]]]] where

- 1463 – YYYY = four-digit year (≥ 0001);
- 1464 – MM = two-digit month (01 through 12 where 01=January, etc.);
- 1465 – DD = two-digit day of month (01 through 31, depending on value of month
1466 and year);
- 1467 – hh = two digits of hour (00 through 23) (am/pm NOT allowed);
- 1468 – mm = two digits of minute (00 through 59);
- 1469 – ss = two digits of second (00 through 59);
- 1470 – s = one or more digits representing a decimal fraction of a second; and
- 1471 – TZD = time zone designator ("Z" for UTC or +hh:mm or -hh:mm)

1472 At least the four-digit year must be present. If additional parts of the DateTime are included,
1473 the character literals "-", "T", ":", and "." are part of the character lexical representation
1474 for the DateTime.

1475 If the time portion is present, but the time zone designator is not present, the time zone is in-
1476 terpreted as being UTC.

1477 This binding example is based on ISO8601:2000 [A4]. This Standard does not specify a bind-
1478 ing, and other bindings are possible.