



1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

**UN/CEFACT
XML Naming and Design Rules
Version 3.0**

**Draft for the 2nd Public Review of ODP5
30 December 2008**

19 Abstract

20 This XML Naming and Design Rules specification defines an architecture and set of
21 rules necessary to define, describe and use XML to consistently express business
22 information exchanges. It is based on the World Wide Web consortium suite of XML
23 specifications and the UN/CEFACT Core Components Technical Specification. This
24 specification will be used by UN/CEFACT to define XML Schema and XML Schema
25 documents which will be published and UN/CEFACT standards. It will also be used
26 by other Standards Development Organizations who are interested in maximizing
27 inter- and intra-industry interoperability.

28

29 **Table of Contents**

30

31	Abstract	2
32	Table of Contents	3
33	1 Status of This Document	8
34	2 XML Naming and Design Rules Project Team Participants	9
35	2.1 Acknowledgements	9
36	2.2 Disclaimer	10
37	2.3 Contact Information	10
38	3 Introduction	11
39	3.1 Summary of Contents of Document	11
40	3.1.1 Notation	12
41	3.2 Audience	12
42	4 Objectives	13
43	4.1 Goals of the Technical Specification	13
44	4.2 Requirements	13
45	4.2.1 Conformance	13
46	4.3 Caveats and Assumptions	14
47	4.3.1 Guiding Principles	15
48	5 XML Schema Architecture	16
49	5.1 Overall XML Schema Structure	16
50	5.2 Relationship to CCTS	17
51	5.2.1 CCTS	18
52	5.2.2 The XML Schema Components	18
53	5.2.3 Context Categories	20
54	5.3 Business Message Syntax Binding	20
55	5.4 Naming and Modeling Constraints	22
56	5.5 Reusability Scheme	25
57	5.6 Namespace Scheme	27
58	5.6.1 Namespace Uniform Resource Identifiers	28
59	5.6.2 Namespace Tokens	30
60	5.7 XML Schema Files	30

61	5.7.1	Root XML Schema Files	33
62	5.7.2	Business Information Entity XML Schema Files	33
63	5.7.3	Business Data Type XML Schema Files.....	34
64	5.7.4	Code List XML Schema Files.....	34
65	5.7.5	Identifier Schemes	36
66	5.7.6	Other Standard Bodies BIE XML Schema Files.....	38
67	5.8	Schema Location	38
68	5.9	Versioning Scheme	39
69	5.9.1	Major Versions.....	39
70	5.9.2	Minor Versions.....	40
71	6	Application of Context	42
72	7	General XML Schema Definition Language Conventions	43
73	7.1	Overall XML Schema Structure and Rules.....	43
74	7.1.1	XML Schema Declaration	43
75	7.1.2	XML Schema File Identification and Copyright Information	43
76	7.1.3	Schema Declaration	43
77	7.1.4	CCTS Artifact Metadata.....	44
78	7.1.5	Constraints on Schema Construction	45
79	7.2	Attribute and Element Declarations.....	45
80	7.2.1	Attributes.....	45
81	7.2.2	Elements.....	46
82	7.3	Type Definitions	46
83	7.3.1	Simple Type Definitions	46
84	7.3.2	Complex Type Definitions	47
85	7.4	Use of Extension and Restriction	47
86	7.4.1	Extension	48
87	7.4.2	Restriction.....	48
88	7.5	Annotation.....	50
89	7.5.1	Documentation.....	50
90	7.5.2	Application Information (AppInfo).....	54
91	8	XML Schema Files	58
92	8.1	XML Schema Files, Context and Namespaces	58
93	8.2	Root XML Schema Files	60
94	8.2.1	XML Schema Structure.....	60
95	8.2.2	Includes	61

96	8.2.3	Root Element Declaration	61
97	8.2.4	Type Definitions	62
98	8.2.5	Annotations	62
99	8.3	Business Information Entity XML Schema Files	63
100	8.3.1	Schema Structure	63
101	8.3.2	Includes	64
102	8.3.3	Type Definitions	64
103	8.3.4	Element Declarations and References.....	67
104	8.3.5	Annotation.....	70
105	8.4	Business Data Type XML Schema Files	76
106	8.4.1	Use of Business Data Type XML Schema Files	76
107	8.4.2	XML Schema Structure.....	76
108	8.4.3	Imports and Includes.....	77
109	8.4.4	Type Definitions	77
110	8.4.5	Attribute and Element Declarations	86
111	8.4.6	Annotations.....	86
112	8.5	Code List XML Schema Files	90
113	8.5.1	General Code List XML Schema Components	90
114	8.5.2	Common Code List XML Schema Components	94
115	8.5.3	Business Code List XML Schema Components	100
116	8.6	Identifier Scheme XML Schema Files	101
117	8.6.1	General Identifier Scheme XML Schema Components	102
118	8.6.2	Common Identifier Scheme XML Schema Components.....	105
119	8.6.3	Business Identifier Scheme XML Schema Components.....	108
120	9	XML Instance Documents	111
121	9.1	Character Encoding	111
122	9.2	xsi:schemaLocation.....	111
123	9.3	Empty Content	111
124	9.4	xsi:type	112
125	9.5	Supplementary Components.....	112
126	Appendix A. Related Documents		113
127	Appendix B. Overall Structure		114
128	B.1	XML Declaration	114
129	B.2	Schema Module Identification and Copyright Information.....	114
130	B.3	Schema Start-Tag.....	115

131	B.4 Includes	116
132	B.5 Imports	116
133	B.6 Elements	116
134	B.7 Root element	117
135	B.8 Type Definitions	117
136	Appendix C. ATG Approved Acronyms and Abbreviations	122
137	Appendix D. Core Component XML Schema File	123
138	Appendix E. Business Data Type XML Schema File	124
139	Appendix F. Annotation Templates	125
140	F.1 Annotation Documentation	125
141	F.2 Annotation Application Information	128
142	Appendix G. Mapping of CCTS Representation Terms to CCT and BDT Data Types	
143		131
144	Appendix H. Use Cases for Code Lists	133
145	H.1 Referencing a Common Code List as a Supplementary Component in a	
146	Business Data Types	134
147	H.2 Referencing any code list using BDT CodeType	135
148	H.3 Referencing a Common Code List in a BDT	136
149	H.4 Choosing or Combining Values from Several Code Lists	137
150	H.5 Restricting the Allowed Code Values	138
151	Appendix I. Alternative Business Message Syntax Binding	139
152	I.1 XML Schema Architecture	139
153	I.1.1 Message Assembly Considerations	139
154	I.1.2. Requirements for XML Element Referencing	139
155	I.1.2.1 Implementation of Aggregations – Nesting or Referencing	139
156	I.1.2.2 Other Usages of XML Referencing	140
157	I.1.2.3 Schema Validation Requirements for XML References	140
158	I.2 General XML Schema Language Conventions	141
159	I.2.1 Overall XML Schema Structure and Rules	141
160	I.2.2 Attribute and Element Declarations	142
161	I.3 XML Schema Files	143
162	I.3.1 Root XML Schema Files	143
163	I.3.2 Business Information Entities XML Schema Files	145
164	Appendix J. Naming and Design Rules List	147
165	Naming and Design Rules for the Alternative Business Message Syntax in Appendix	
166	I	173

167 Appendix K. Glossary.....176
168 Disclaimer 182
169 Copyright Statement.....183
170

171 **1 Status of This Document**

172 This UN/CEFACT technical specification is being developed in accordance with the
173 UN/CEFACT/TRADE/R.650/Rev.4/Add.1/Rev.1 Open Development Process (ODP)
174 for technical specifications. The UN/CEFACT Applied Technology Group (ATG) has
175 approved it for broad public review.

176 This technical specification contains information to guide in interpretation or
177 implementation.

178 Specification formatting is based on the Internet Society's Standard RFC format.

179 Distribution of this document is unlimited.

180 This version: UN/CEFACT XML Naming and Design Rules, Version 3.0 ODP 5 Draft
181 2nd Public Review Internal Review 3 of 18 November 2008.

182 Previous version: UN/CEFACT XML Naming and Design Rules, Version 3.0 ODP 5
183 Draft 2nd Public Review Internal Review 2 of 4 November 2008

184 This document may also be available in these non-normative formats: XML, XHTML
185 with visible change markup. See also translations.

186 Copyright © 2008 UN/CEFACT, All Rights Reserved. UN liability, trademark and
187 document use rules apply.

188

189 **2 XML Naming and Design Rules Project Team** 190 **Participants**

191 We would like to recognize the following for their significant participation in the
192 development of *this United Nations Centre For Trade Facilitation and Electronic*
193 *Business (UN/CEFACT) XML Naming and Design Rules* technical specification.

194 **ATG2 Chair**

Jostein Frømyr	EdiSys Consulting AS
----------------	----------------------

195 **Project Team Leader**

Mark Crawford	SAP Labs LLC (U.S.)
---------------	---------------------

196 **Lead Editor**

Michael Rowell	Oracle Corporation / OAGi
----------------	---------------------------

197 **Contributors**

Chuck Allen	HR-XML
-------------	--------

Dipan Anarkat	GS1
---------------	-----

Serge Cayron	ACORD
--------------	-------

Anthony Coates	Independent
----------------	-------------

David Connelly	OAGi
----------------	------

Mavis Cournane	Independent
----------------	-------------

Alain Dechamps	CEN
----------------	-----

Michael Grimley	US Navy
-----------------	---------

Paul Hojka	APACS
------------	-------

Kevin Smith	Independent
-------------	-------------

Gunther Stuhec	SAP AG
----------------	--------

Jim Wilson	KCX / CIDX
------------	------------

198 **2.1 Acknowledgements**

199 This version of UN/CEFACT - *XML Naming and Design Rule* was created to foster
200 convergence among Standards Development Organizations (SDOs) with close
201 coordination with these organizations.

- 202 • ACORD

- 203 • CIDX
- 204 • GS1
- 205 • HR-XML
- 206 • OASIS Universal Business Language (UBL) Technical Committee
- 207 • Open Application Group (OAGi)

208 **2.2 Disclaimer**

209 The views and specification expressed in this technical specification are those of the
210 authors and are not necessarily those of their employers. The authors and their
211 employers specifically disclaim responsibility for any problems arising from correct or
212 incorrect implementation or use of this technical specification.

213 **2.3 Contact Information**

- 214 ATG2 – Jostein Frømyr , EdiSys Consulting AS, Jostein.Fromyr@edisys.no
215 NDR Project Lead – Mark Crawford, SAP Labs LLC (U.S.), mark.crawford@sap.com
216 Lead Editor – Michael Rowell, Oracle Corporation, michael.rowell@oracle.com

217 **3 Introduction**
 218 **3.1 Summary of Contents of Document**

219 This specification consists of the following Sections and Appendices.

Abstract	Informative
Table of Contents	Informative
Section 1: Status of this Document	Informative
Section 2: Project Team	Informative
Section 3: Introduction	Informative
Section 4: Objectives	Normative
Section 5: XML Schema Architecture	Normative
Section 6: Application of Context	Informative
Section 7: General XML Schema Language Conventions	Normative
Section 8: XML Schema Files	Normative
Section 9: XML Instance Documents	Normative
Appendix A: Related Documents	Informative
Appendix B: Overall Structure	Normative
Appendix C: ATG Approved Acronyms and Abbreviations	Normative
Appendix D: Business Data Type XML Schema File	Normative
Appendix E: Annotation AppInfo Templates	Informative
Appendix F: Annotation Documentation Templates	Informative
Appendix G: Mapping of CCTS Representation Terms to CCT and BDT	Informative
Appendix H: Common Use Cases for Code Lists	Informative
Appendix I: Alternate Message Assembly	Informative
Appendix J: Naming and Design Rules List	Normative
Appendix K: Glossary	Normative

220 3.1.1 Notation

221 The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD,
222 SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this
223 specification, are to be interpreted as described in [Internet Engineering Task Force](#)
224 [\(IETF\) Request For Comments \(RFC\) 2119](#).¹

225 Wherever **xsd:** appears in this specification it refers to a construct taken from one of
226 the W3C XML Schema recommendations. Wherever **ccts:** appears it refers to a
227 construct taken from the *UN/CEFACT Core Components Technical Specification*.

228 Example – A representation of a definition or a rule. Examples are informative.

229 [Note] – Explanatory information. Notes are informative.

230 [R n] – Identification of a rule that requires conformance. Rules are normative. In
231 order to ensure continuity across versions of the specification, rule numbers are
232 randomly generated. The number of a rule that is deleted will not be re-issued.
233 Rules that are added will be assigned a previously unused random number.

234 **Courier** – All words appearing in bolded **courier font** are values, objects or
235 keywords.

236 When defining rules, the following annotations are used:

237 [] = optional

238 < > = variable

239 | = choice

240 3.2 Audience

241 The audience for this UN/CEFACT - *XML Naming and Design Rules* Technical
242 Specification is:

- 243 • Members of the UN/CEFACT Applied Technologies Group who are
244 responsible for development and maintenance of UN/CEFACT XML
245 Schema
- 246 • The wider membership of the other UN/CEFACT Groups who participate
247 in the process of creating and maintaining UN/CEFACT XML Schema
248 definitions
- 249 • Designers of tools who need to specify the conversion of user input into
250 XML Schema definitions adhering to the rules defined in this document.
- 251 • Designers of XML Schema definitions outside of the UN/CEFACT Forum
252 community. These include designers from other standards organizations
253 and companies that have found these rules suitable for their own
254 organizations.

Key words for use in RFCs to Indicate Requirement Levels - Internet Engineering Task Force, Request For Comments 2119, March 1997, <http://www.ietf.org/rfc/rfc2119.txt?number=2119>

255 **4 Objectives**

256 **4.1 Goals of the Technical Specification**

257 This technical specification has been developed to provide for XML standards based
258 expressions of semantic data models representing business information exchanges.
259 It can be employed wherever business information is being shared in an open
260 environment using XML Schema to define the structure of business content. It
261 describes and specifies the rules and guidelines UN/CEFACT will use for developing
262 XML schema and schema documents based on Core Component Technical
263 Specification (CCTS) conformant artifacts and information models developed in
264 accordance with the UN/CEFACT CCTS Technical Specification Version 3.0.

265 **4.2 Requirements**

266 Users of this specification should have an understanding of basic data modeling
267 concepts, basic business information exchange concepts and basic XML concepts.

268 **4.2.1 Conformance**

269 Designers of XML Schema in governments, private sector, and other standards
270 organizations external to the UN/CEFACT community have found this specification
271 suitable for adoption. To maximize reuse and interoperability across this wide user
272 community, the rules in this specification have been categorized to allow these other
273 organizations to create conformant XML Schema while allowing for discretion or
274 extensibility in areas that have minimal impact on overall interoperability.

275 Accordingly, applications will be considered to be in full conformance with this
276 technical specification if they comply with the content of normative sections, rules
277 and definitions.

278 Rules in categories 1, 4 and 5 can not be modified. Rules in categories 2, 3, 6, and
279 7 may be tailored within the limits identified in the rule and the related normative text.

[R B998]	Conformance SHALL be determined through adherence to the content of the normative sections and rules. Furthermore each rule is categorized to indicate the intended audience for the rule by the following:		1
	Rule Categorization		
	ID	Description	
	1	Rules which must not be violated by individual organizations else conformance and interoperability is lost – such as named types.	
	2	Rules which may be modified by individual organizations while still conformant to the NDR structure – such as namespace string contents and namespace tokens.	
	3	Rules which may be modified by individual organizations while still conformant to agreed upon data models – such as the use of global or local element declarations. (Changes to the XML Schema Architecture.)	
	4	Rules that if violated lose conformance with the UN/CEFACT data/process model – such as xsd:redefine , xsd:any , and xsd:substitutionGroups .	
	5	Rules that relate to extension that are not used by UN/CEFACT and have specific restrictions on their use by other than UN/CEFACT organizations.	
	6	Rules that relate to extension that are determined by specific organizations.	
7	Rules that can be modified while not changing instance validation capability.		

280 4.3 Caveats and Assumptions

281 Schema created as a result of employing this specification should be made publicly
 282 available as schema documents in a universally freely accessible library.
 283 UN/CEFACT will maintain their XML Schema as published documents in an ebXML
 284 compliant registry and make its contents freely available to any government,
 285 individual or organization who wishes access.

286 Although this specification defines schema components as expressions of core
 287 component artifacts, it can also be used by non-CCTS developers for other class
 288 based expressions of logical data models and information exchanges.

289 This specification does not address transformations via scripts or any other means.
290 It does not address any other representation of Core Component artefacts, for
291 example, OWL, Relax NG, XMI and others are outside the scope of this document.

292 **4.3.1 Guiding Principles**

293 The following guiding principles were used as the basis for all design rules contained
294 in this specification.

295

- 296 • Relationship to UMM – UN/CEFACT XML Schema definitions will be based on
297 UMM metamodel adherent Business Process Models.
- 298 • Relationship to Information Models – UN/CEFACT XML Schema will be based
299 on information models developed in accordance with the UN/CEFACT – *Core*
300 *Components Technical Specification*.
- 301 • XML Schema Creation – UN/CEFACT XML Schema design rules will support
302 XML Schema creation through handcrafting as well as automatic generation.
- 303 • Interchange and Application Use – UN/CEFACT XML Schema and the
304 resulting XML instance documents are intended for a variety of data
305 exchanges.
- 306 • Tool Use and Support - The design of UN/CEFACT XML Schema will not
307 make any assumptions about sophisticated tools for creation, management,
308 storage, or presentation being available.
- 309 • Legibility - UN/CEFACT XML instance documents should be intuitive and
310 reasonably clear in the context for which they are designed.
- 311 • Schema Features - The design of UN/CEFACT XML Schema should use the
312 most commonly supported features of W3C XML Schema Recommendation.
- 313 • Technical Specifications – UN/CEFACT XML Naming and Design Rules will
314 be based on Technical Specifications holding the equivalent of W3C
315 recommended status.
- 316 • XML Schema Specification – UN/CEFACT XML Naming and Design Rules
317 will be fully conformant with W3C XML Schema Recommendation.
- 318 • Interoperability - The number of ways to express the same information in a
319 UN/CEFACT XML Schema and UN/CEFACT XML instance document is to be
320 kept as close to one as possible.
- 321 • Maintenance – The design of UN/CEFACT XML Schema must facilitate
322 maintenance.
- 323 • Context Sensitivity - The design of UN/CEFACT XML Schema must ensure
324 that context-sensitive document types are not precluded.
- 325 • Relationship to Other Namespaces - UN/CEFACT is cautious about making
326 dependencies on other namespaces.
- 327 • Legacy formats - UN/CEFACT XML Naming and Design Rules are not
328 responsible for sustaining legacy formats.

329 **5 XML Schema Architecture**

330 This section defines rules and the corresponding text related to general XML Schema
331 construction including:

- 332 • Overall XML Schema Structure
- 333 • Relationship to CCTS
- 334 • Business Message Syntx Binding
- 335 • Naming and Modeling Constraints
- 336 • Reusability Scheme
- 337 • Namespace Scheme
- 338 • XML Schema Files
- 339 • Schema Location
- 340 • Versioning Scheme

341 **5.1 Overall XML Schema Structure**

342 UN/CEFACT has determined that the World Wide Web Consortium (W3C) XML
343 Schema Recommendation is the schema definition language with the broadest
344 adoption and tool support. Accordingly, all UN/CEFACT XML Schema definitions will
345 be expressed in XML Schema. All references to W3C XML Schema will be as XML
346 Schema. References to XML Schema defined by UN/CEFACT will be as
347 UN/CEFACT XML Schema.

[R 8059]	All XML Schema design rules MUST be based on the W3C XML Schema Recommendations: XML Schema Part 1: Structures Second Edition and XML Schema Part 2: Datatypes Second Edition .	1
----------	--	---

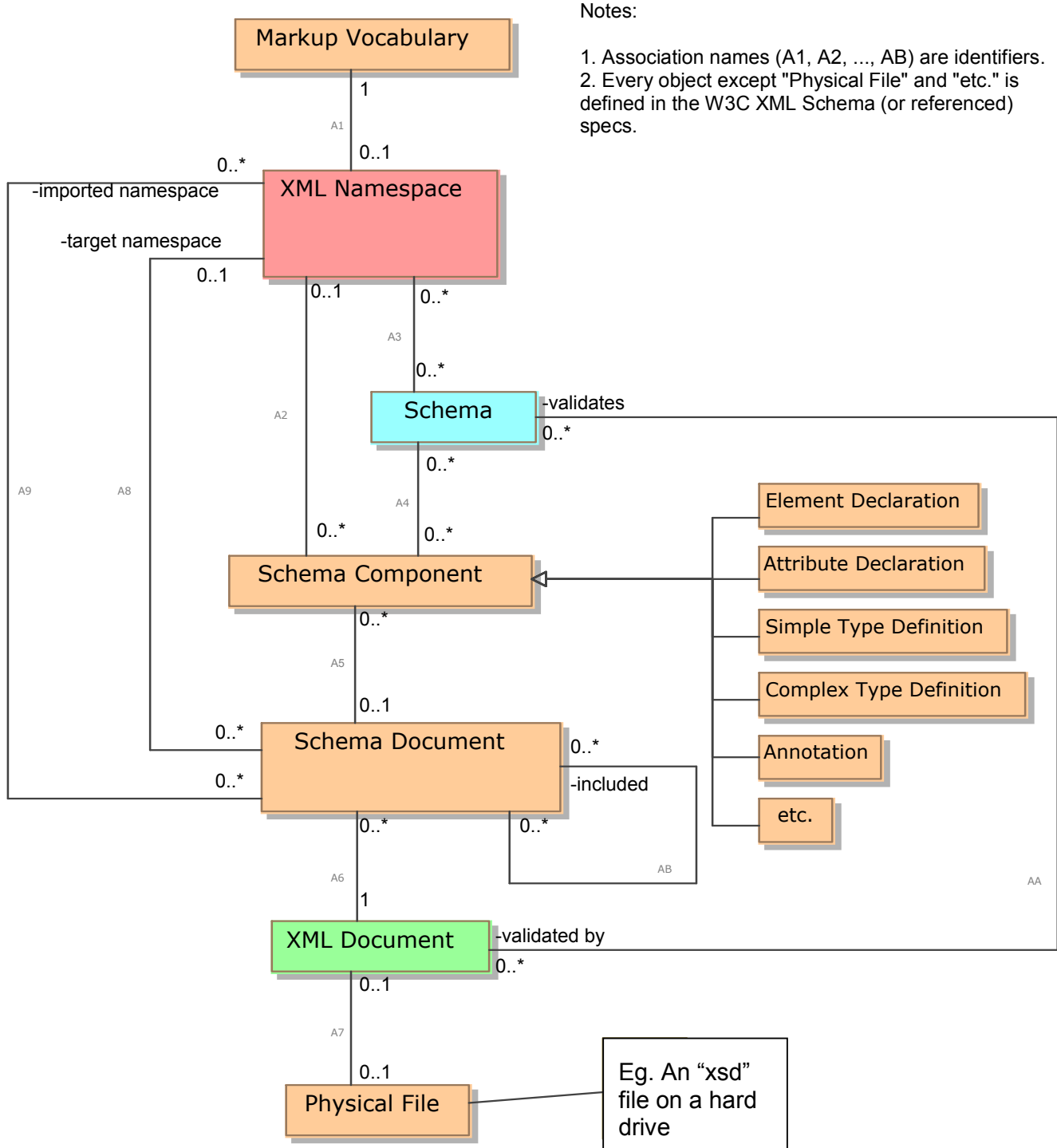
348 The W3C is the recognized source for XML specifications. W3C specifications can
349 hold various statuses. Only those W3C specifications holding recommendation
350 status are considered by the W3C to be stable specifications.

[R 935C]	All conformant XML instance documents MUST be based on the W3C suite of technical specifications holding recommendation status.	1
----------	--	---

351 To maintain consistency in lexical form, all XML Schema need to use a standard
352 structure for all content. This standard structure is contained in Appendix B.

[R 9224]	XML Schema MUST follow the standard structure defined in Appendix B of this document.	1
----------	--	---

353 The W3C XML Schema specification uses specific terms to define the various
354 aspects of a W3C XML Schema. These terms and concepts are used without
355 change in this NDR specification. Figure 5-1, shows these terms and concepts and
356 their relationship as defined by the W3C.



357 **Figure 5-1 W3C XML Schema terms and concepts.**

358 **5.2 Relationship to CCTS**

359 All UN/CEFACT business information modeling and business process modeling
 360 employ the methodology and model described in UN/CEFACT CCTS.

361 **5.2.1 CCTS**

362 CCTS provides a way to identify, capture and maximize the re-use of business
363 information to support and enhance information interoperability.

364 The foundational concepts of CCTS are Core Components (CC) and Business
365 Information Entities (BIE). Core Components are building blocks that can be used for
366 all aspects of data modeling, information modelling and information exchange. Core
367 components are conceptual models that are used to define Business Information
368 Entities (BIEs).

369 BIEs are logical data model artefact expressions. BIEs are used for creating
370 interoperable business process models, business documents, and information
371 exchanges. BIEs are created through the application of context to a CC that may:

- 372 • Be qualified to provide a unique business semantic,
- 373 • Specify a restriction from the underlying CC.

374 Core Components include Aggregate Core Components (ACCs), Basic Core
375 Components (BCCs) and Association Core Components (ASCCs). Business
376 Information Entities (BIE) include Aggregate Business Information Entities (ABIEs),
377 Basic Business Information Entities (BBIEs) and Association Business Information
378 Entities (ASBIEs).

379 The CCTS model for BIEs includes:

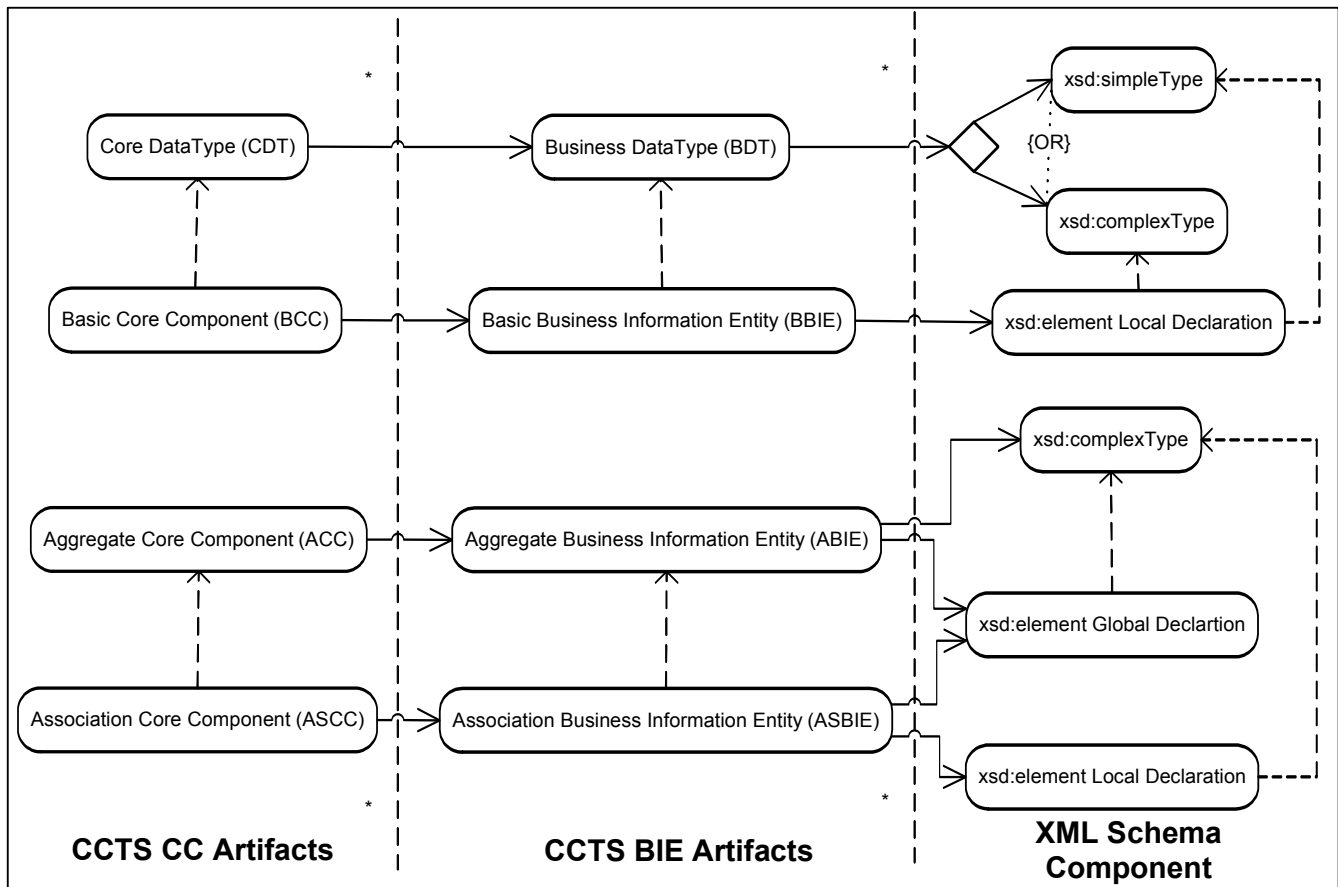
- 380 • Common Information – information that is expressed in the annotation
381 documentation in the XML Schema.
- 382 • Localized Information – information that while expressed in the model is not
383 expressed in the XML Schema.
- 384 • Usage Rules – information that is expressed in the annotation application
385 information in the XML Schema.

386 **5.2.2 The XML Schema Components**

387 UN/CEFACT XML Schema design rules are closely coupled with CCTS. Thus,
388 UN/CEFACT XML Schema will be developed from fully conformant Business
389 Information Entities that are based on fully conformant Core Components. Figure 5-2
390 shows the relationship between relevant CCTS CC artifacts, BIE artifacts and XML
391 Schema Components.

392 [Note:]

393 CCTS specifies DataTypes, CCs and BIEs. The columns in Figure 5-2 indicate the
394 conceptual CC Model view and the logical BIE Model view and how these are
395 translated to XML Schema.



396 **Figure 5-2 Transitions between CCTS Artifacts and XML Schema Components**

397 The solid arrows flowing from the CC to the BIE column show the direct mapping of
 398 the artifacts from CC to BIEs as defined by CCTS.

399 The solid arrow flowing between the BIE column and the XML Schema Component
 400 column show the direct mapping from the BIE to the XML Schema Component used
 401 to represent it. The dotted arrows with the XML Schema Component column indicate
 402 that the given element makes use of the artefact type pointed to by the arrow.

403 **5.2.2.1 Aggregate Business Information Entity**

404 All Aggregate Business Information Entities (ABIEs) are represented as a type
 405 definition (`xsd:complexType`) and global element (`xsd:element`) declaration in
 406 the UN/CEFACT BIE XML Schema File for the namespace in which they are
 407 defined. See section [8.3 Business Information Entities XML Schema Files](#).

408 **5.2.2.2 Association Business Information Entity**

409 Whether an Association Business Information Entity (ASBIE) uses a local or global
 410 element depends upon the type of association (**AggregationKind=shared** or
 411 **AggregationKind=composite**) specified in the model. An ASBIE will be declared
 412 as either a local element or as a global element.

- 413 • If the ASBIE is a “composition” association
- 414 (**AggregationKind=composite**). The ASBIE is declared as a local

415 element (`xsd:element`) within the type (`xsd:complexType`) representing
416 the associating ABIE. This local element (`xsd:element`) makes use of the
417 type (`xsd:complexType`) of associated ABIE.

- 418 • If it is a “shared” association (`AggregationKind=shared`). The ASBIE is
419 referenced as a global element (`xsd:element`) within the type representing
420 the associating ABIE. The global element (`xsd:element`) is declared in the
421 same namespace as the associating ABIE and makes use of the type
422 (`xsd:complexType`) of the associated ABIE.

423 See section [8.3 Business Information Entities XML Schema Files](#).

424 5.2.2.3 Basic Business Information Entity

425 A Basic Business Information Entity (BBIE) is declared as a local element within the
426 `xsd:complexType` representing the parent ABIE. The BBIE is based on a (is of
427 type) BDT. See section [8.3 Business Information Entities XML Schema Files](#).

428 5.2.2.4 Business Data Type

429 A Business Data Type (BDT) is defined as either an `xsd:complexType` or
430 `xsd:simpleType`. If the BDT value domain can be expressed by the facets of an
431 `xsd` built in data type, then the BDT will be defined as an `xsd:simpleType` whose
432 `xsd:base` is the `xsd` built in type.

433 If not, then an `xsd:complexType` will be defined with a content model to support
434 the value domain.

435 See section [8.4 Business Data Type XML Schema Files](#).

436 5.2.3 Context Categories

437 The CCTS identifies a set of context categories – such as business process,
438 geopolitical, system capabilities, business process role – the values of these
439 categories collectively define the context in which context specific BIEs are defined.
440 This NDR specification captures the context through the use of an annotation
441 application information element (`<xsd:annotation> <xsd:appInfo>`)
442 accompanying each element declaration. See section [7.5.2 Application Information](#)
443 [\(AppInfo\)](#) for more information.

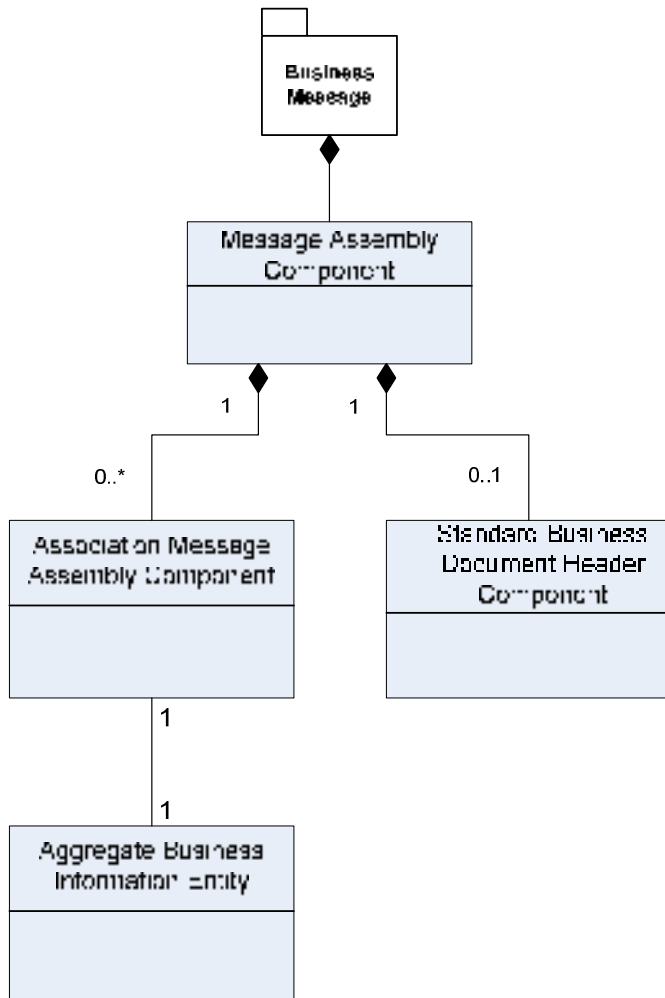
444 UN/CEFACT uses the business process context value to create different
445 namespaces. Each organization adhering to this specification will choose a context
446 category value to incorporate into their namespace. This context category should be
447 the dominant context category for their use. See section [6 Application of Context](#).

448 5.3 Business Message Syntax Binding

449 UN/CEFACT will create the XML syntax binding of its CCTS conformant BIE data
450 models directly from the associations and hierarchies expressed in the Business
451 Message Template for each business message exchange. This transformation
452 approach is based on traditional nesting of all components of the data model.

453 Figure 5-3 shows the UN/CEFACT Business Message structure as defined in the
 454 Business Message Template. The Business Message structure consists of a single
 455 Message Assembly (MA) component representing the Business Message. Each
 456 Association Message Assembly (ASMA) is a proxy for the first level ABIE in a given
 457 Business Message. Additionally, application specific information unique to the
 458 instance can be defined using the UN/CEFACT Standard Business Document
 459 Header specification.

460 The XML Schema Specification also supports an alternative to nesting. This
 461 alternative – using schema identity constraints (`xsd:key/xsd:keyRef` – enables
 462 referencing and reuse of a given XML element in instance documents. UN/CEFACT
 463 is currently evaluating this alternative for future use to include a method for
 464 application at the data model level. In anticipation that the data model issues will be
 465 resolved, UN/CEFACT has already developed a set of rules for its XML
 466 implementation. These rules and the supporting narrative can be found in [Appendix I](#)
 467 [Alternative Business Message Syntax Binding](#). Organizations using this alternative
 468 method will still be considered conformant to this specification, if they adhere to all
 469 other conformance requirements and use the rules defined in the [Appendix I](#)
 470 [Alternative Business Message Syntax Binding](#).



471

472 **Figure 5-3 Business Message Template Metamodel**

473 The business message MA component is defined as a global type and declared as
 474 the sole global element in the Root XML Schema File. The MA content model
 475 consists of a set of ASMA element declarations whose type is the xsd:complexType
 476 definition in the BIE XML Schema File that represent the first level ABIEs used in the
 477 message. It may also contain an optional Standard Business Document Header
 478 component. See section [8.2 Root XML Schema Files](#).

479 5.4 Naming and Modeling Constraints

480

481 UN/CEFACT XML Schemas are derived from components created through the
 482 application of CCTS.UN/CEFACT XML Schema contain XML Schema Components
 483 that follow the naming and design rules in this specification.

484 These naming and design rules take advantage of the features of the XML Schema
 485 specification. In many cases this approach results in the truncation of the CCTS
 486 Dictionary Entry Names (DENs). However, the fully conformant CCTS DENs of the
 487 underlying CCTS artefacts are preserved as part of the annotation documentation
 488 (<xsd:annotation> <xsd:documentation>) element accompanying each
 489 element declaration.

490 The CCTS DEN can be reconstructed by using XPath expressions. The fully
 491 qualified XPath (FQXP) ties the information to its standardized CCTS semantics,
 492 while the XML element or attribute name is a truncation that reflects the hierarchy of
 493 the XML construct.

494 The FQXP anchors the use of a construct to a particular location in a business
 495 information payload. The DEN identifies any semantic dependencies that the FQXP
 496 has on other elements and attributes within the UN/CEFACT library that are not
 497 otherwise enforced or made explicit in its structural definition. The dictionary serves
 498 as a traditional data dictionary, and also provides some of the functions of a
 499 traditional implementation guide.

[R A9E2]	Each element or attribute XML name MUST have one and only one fully qualified XPath (FQXP).	1
----------	---	---

500 Example 5-1 shows a FQXP for Address Coordinate LatitudeMeasure and
 501 Organization Location Name.

502 Example 5-1: Fully Qualified XPath

503
504

Address/Coordinate/LatitudeMeasure Organisation/Location/Name
--

505 The official language for UN/CEFACT is English. All official XML constructs
 506 published by UN/CEFACT will be in English. XML and XML Schema development
 507 work may very well occur in other languages, however official submissions for
 508 inclusion in the UN/CEFACT XML Schema library must be in English. Other language
 509 translations of UN/CEFACT published XML Instances and XML Schema
 510 Components are at the discretion of the users.

[R AA92]	Element, attribute and type names MUST be composed of words in the English language, using the primary English spellings provided in the Oxford English Dictionary.	1
----------	---	---

511 LowerCamelCase (LCC) is used for naming XML Schema attributes and
 512 UpperCamelCase (UCC) is used for naming XML Schema elements and types.
 513 LowerCamelCase capitalizes the first character of each word except the first word
 514 and compounds the name. UpperCamelCase capitalizes the first character of each
 515 word and compounds the name.

[R 9956]	LowerCamelCase (LCC) MUST be used for naming attributes.	1
[R A781]	UpperCamelCase (UCC) MUST be used for naming elements and types.	1
[R 8D9F]	Element, attribute and type names MUST be in singular form unless the concept itself is plural.	1

516 Examples 5-2 through 5-6 show examples of what is allowed and not allowed.

517 **Example 5-2: Attribute**

518 Allowed

519

```
<xsd:attribute name="unitCode" .../>
```

520 **Example 5-3: Element**

521 Allowed

522

```
<xsd:element name="LanguageCode" ...>
```

523 **Example 5-4: Type**

524 Allowed

525

```
<xsd:complexType name="DespatchAdviceCodeType">
```

526 **Example 5-5: Singular and Plural Concept Form**

527 Allowed - Singular:

528

```
<xsd:element name="GoodsQuantity" ...>
```

529 Not Allowed - Plural:

530

```
<xsd:element name="ItemsQuantity" ...>
```

531 **Example 5-6: Non-Letter Characters**

532 Not Allowed

533

```
<xsd:element name="LanguageCode8" ...>
```

534 While CCTS allows for the use of periods, spaces and underscores in the dictionary
 535 entry name. XML best practice is to not include these characters in an XML tag
 536 name. Additionally, XML 1.0 specifically prohibits the use of certain reserved
 537 characters in XML tag names.

[R AB19]	XML element, attribute and type names constructed from dictionary entry names MUST NOT include periods, spaces, or other separators; or characters not allowed by W3C XML 1.0 for XML names.	1
[R 9009]	XML element, attribute and type names MUST NOT use acronyms, abbreviations, or other word truncations, except those included in the defining organizations list of approved acronyms and abbreviations.	1

538 Examples 5-7 and 5-8 show examples of what is allowed and not allowed.

539 **Example 5-7: Spaces in Name**

540 Not Allowed

541

```
<xsd:element name="Customized_ Language. Code:8" ...>
```

542 **Example 5-8: Acronyms and Abbreviations**

543 Allowed – ID is an approved abbreviation

544

```
<xsd:attribute name="currencyID"
```

545 Not Allowed – Cd is not an approved abbreviation, if it was an approved abbreviation
 546 it must appear in all upper case

547

```
<xsd:simpleType name="temperatureMeasureUnitCdType">
```

[R BFA9]	The acronyms and abbreviations listed by the defining organization MUST always be used in place of the word or phrase they represent.	1
[R 9100]	Acronyms MUST appear in all upper case except for when the acronym is the first set of characters of an attribute in which case they will be all lower case.	1

548 **5.5 Reusability Scheme**

549 UN/CEFACT is committed to an object based approach for its process, data, and
550 information models.

551 UN/CEFACT considered adopting an XSD type based approach which uses named
552 types, a type and element based approach, or an element based approach. A type
553 based approach for XML management provides the closest alignment with the
554 process modelling methodology described in UMM. Type information is beginning to
555 be accessible when processing XML instance documents. Post schema-validation
556 infoset (PSVI) capabilities are beginning to emerge that support this approach, such
557 as “data-binding” software that compiles schema into ready-to-use object classes
558 and is capable of manipulating XML data based on their types.

559 The most significant drawback to a type based approach is the risk of developing an
560 inconsistent element vocabulary where elements are declared locally and allowed to
561 be reused without regard to semantic clarity and consistency across types.

562 UN/CEFACT manages this risk by carefully controlling the creation of BBIEs and
563 ASBIEs with fully defined semantic clarity that are only usable within the ABIE in
564 which they appear. This is accomplished through the relationship between BBIEs,
565 ASBIEs and their parent ABIE and the strict controls put in place for harmonization
566 and approval of the semantic constructs prior to their XML Schema instantiation.

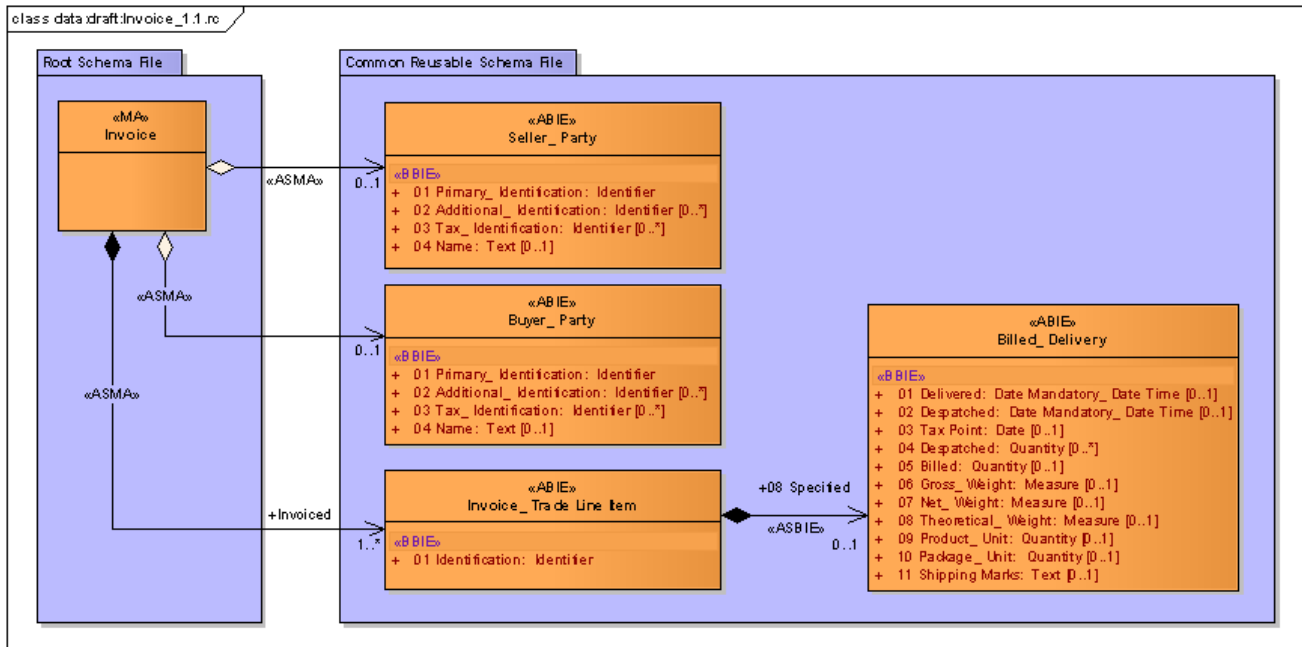
567 A purely type based approach does, however, limit the ability to reuse elements,
568 especially in technologies such as Web Services Description Language (WSDL).

569 For these reasons, UN/CEFACT implements a “hybrid approach” that provides
570 benefits over a pure type based approach. Most significantly it increases reusability
571 of library content both at the modelling and XML Schema level.

572 The key principles of the “hybrid approach” are:

- 573 • All classes (Invoice, Seller_Party, Buyer_Party, Invoice_Trade.Line.Item and
574 Billed_Delivery in Figure 5-4) are declared as a **xsd:complexType**.
- 575 • All attributes of a class are declared as a local **xsd:element** within the
576 corresponding **xsd:complexType**.
- 577 • UML **aggregationKind=composite** associations will result in a locally
578 declared **xsd:element** with a globally declared **xsd:complexType** (e.g.
579 Invoice_Trade.Line.Item and Billed_Delivery in Figure 5-4). A composite
580 aggregation ASBIE represents a relationship wherein if the associating ABIE
581 ceases to exist the associated ABIE ceases to exist.
- 582 • UML **aggregationKind=shared** associations will result in a globally
583 declared **xsd:element** with a globally declared **xsd:complexType** (e.g.
584 Invoice.Buyer. Buyer_Party, Invoice. Seller. SellerParty in Figure 5-4). A
585 shared aggregation ASBIE represents a relationship wherein if the associating
586 ABIE ceases to exist, the associated ABIE continues to exist.

587 The rules pertaining to the 'hybrid approach' are contained in sections [8.3.3 Type](#)
 588 [Definitions](#) and [8.3.4 Element Declarations and References](#).



589 **Figure 5-4 UML Model Example**

590 Figure 5-4 shows an example UML model and Example 5-9 shows the resulting XML
 591 Schema declaration that results from the translation from UML to XML Schema
 592 following the rules defined in this specification.

593 **[Note] - Tokens**

594 The tokens rsm, bie, bdt, bcl, and ccl are used throughout this document to
 595 generically represent root XML Schema Files, BIE XML Schema Files, BDT XML
 596 Schema Files, Business Code List XML Schema Files, and Common Code List XML
 597 Schema Files. The actual tokens are developed using the rules stated elsewhere in
 598 this specification.

599 **Example 5-9: XML Schema declarations representing Figure 5-4.**

600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621

```
<xsd:element name="InvoiceRequest" type="rsm:InvoiceType"/>
<xsd:element name="BuyerParty" type="bie:BuyerPartyType"/>
<xsd:element name="InvoiceTradeLineItem" type="bie:InvoiceTradeLineItemType"/>
<xsd:element name="SellerParty" type="bie:SellerPartyType"/>

<xsd:complexType name="InvoiceType">
  <xsd:sequence>
    <xsd:element name="ID" type="bdt:IDType"/>
    <xsd:element ref="bie:SellerParty"/>
    <xsd:element ref="bie:BuyerParty"/>
    <xsd:element name="InvoiceTradeLineItem"
type="bie:InvoiceTradeLineItemType" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="BuyerPartyType">
  <xsd:sequence>
    <xsd:element name="ID" type="bdt:IDType"/>
    <xsd:element name="Name" type="bdt:NameType"/>
  </xsd:sequence>
</xsd:complexType>
```

622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643

```

<xsd:complexType name="InvoiceTradeLineItemType">
  <xsd:sequence>
    <xsd:element name="ID" type="bdt:IDType"/>
    <xsd:element name="BilledDelivery" type="bie:BilledDeliveryType"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="BilledDeliveryType">
  <xsd:sequence>
    <xsd:element name="ID" type="bdt:IDType"/>
    <xsd:element name="Name" type="bdt:NameType"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="SellerPartyType">
  <xsd:sequence>
    <xsd:element name="ID" type="bdt:IDType"/>
    <xsd:element name="GivenName" type="bdt:NameType"/>
    <xsd:element name="Surname" type="bdt:NameType"/>
  </xsd:sequence>
</xsd:complexType>

```

644 5.6 Namespace Scheme

645 A namespace is an abstract container for a collection of elements, attributes and
646 types that serve to uniquely identify this collection from other collections.

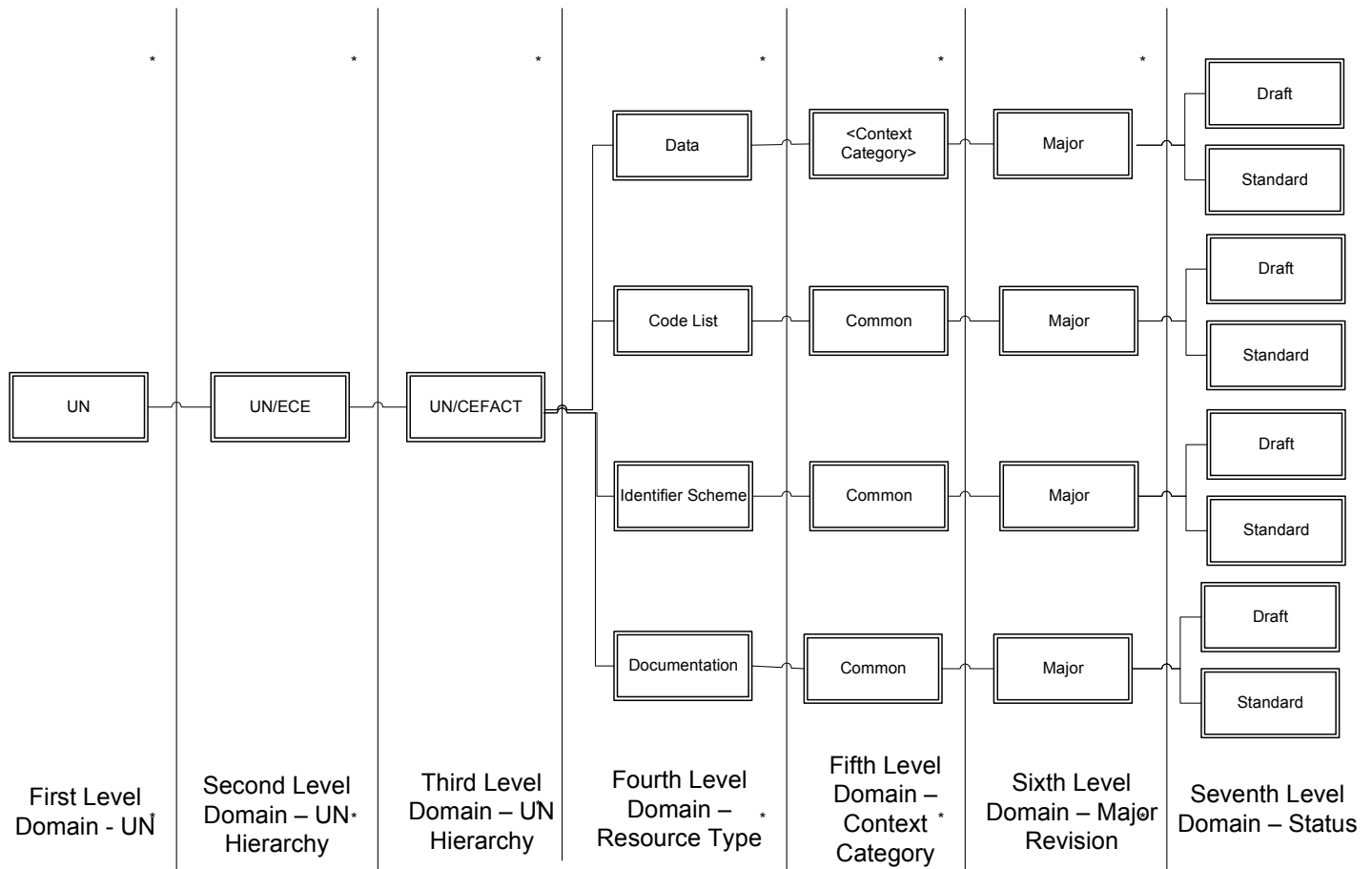
647 “An XML namespace is identified by a URI reference [RFC3986]; element and
648 attribute names may be placed in an XML namespace...”² UNCEFACT assigns XML
649 artifacts to UNCEFACT namespaces following the namespace scheme shown in
650 Figure 5-5.

651 Each organization that intends to adhere to this specification will assign their XML
652 Schema defined content in a namespace that reflects the name of the organization
653 and the primary context category value in which the XML Schema is defined similar
654 to the UN/CEFACT namespace scheme shown in Figure 5-5.

[R 984C]	Each organization’s XML Schema components MUST be assigned to a namespace for that organization.	1
----------	--	---

655 [Note:]
656 The primary context category expressed in the namespace may be chosen by the
657 organization defining or publishing the given set of XML Schema Files.
658 UN/CEFACT has chosen to use the Business Process context category and.
659 UN/CEFACT XML Schema Files will be expressed within a namespace that reflects
660 the Business Process Value that the CCTS artifacts in which the contained XML
661 Schema Components are derived are defined.

² <http://www.w3.org/TR/2006/REC-xml-names-20060816/>



662 **Figure 5-5: UN/CEFACT Namespace Scheme**

663 **5.6.1 Namespace Uniform Resource Identifiers**

664 Namespaces must be persistent. Namespaces should be resolvable. A URI is used
 665 for identifying a namespace. Within the URI space, options include Uniform
 666 Resource Locators (URLs) and Uniform Resource Names (URNs). A URN has an
 667 advantage in that it is persistent. A URL has an advantage in that it implies
 668 resolvability.

669 To ensure consistency, each namespace identifier will have the same general
 670 structure. The URN namespace structure will follow the provisions of *Internet
 671 Engineering Task Force (IETF) Request For Comments (RFC) 2141 – URN Syntax*.

672 The URN format will be:

```
673 urn:<organization>:<org hierarchy>[:<org hierarchy  
674 level>]*:<schematype>:<context category>:<major>:<status>
```

675 The URL namespace structure will follow the provisions of Internet Engineering Task
 676 Force (IETF) Request for Comments (RFC) 1738 – Uniform Resource Locators
 677 (URL)

678 The URL format will be:

```
679 http://<organization>/<org hierarchy>[/<org hierarchy  
680 level>]*/<schematype>/<context category>/<major>/<status>
```

681 Where:

- 682 • organization – An identifier of the organization providing the standard.
- 683 • org hierarchy – The first level of the hierarchy within the organization
- 684 providing the standard.
- 685 • org hierarchy level – Zero to n level hierarchy of the organization providing the
- 686 standard.
- 687 • schematype – A token identifying the type of schema module:
- 688 data|codelist|documentation.
- 689 • context category – The context category [business process] for UN/CEFACT
- 690 from the UN/CEFACT catalogue of common business processes. Other
- 691 values may be used by the other organizations.
- 692 • major – The major version number.
- 693 • status – The status of the schema as: **draft|standard**.

694 UN/CEFACT has determined that URNs are most appropriate as persistence is of a
 695 higher priority for UN/CEFACT. Furthermore, UN/CEFACT recommends that URNs
 696 be used by other organizations that use this specification.

[R 8E2D]	<p>The XML Schema namespaces MUST use the following pattern:</p> <table border="1" data-bbox="440 947 1349 1213"> <tr> <td data-bbox="440 947 548 1079">URN:</td> <td data-bbox="548 947 1349 1079">urn:<organization>:<org hierarchy>[:<org hierarchy level>]*:<schematype>:<context category>:<major>:<status></td> </tr> <tr> <td data-bbox="440 1079 548 1213">URL:</td> <td data-bbox="548 1079 1349 1213">http://<organization>/<org hierarchy>[/<org hierarchy level>]*/<schematype>/context category/<major>/<status></td> </tr> </table>	URN:	urn:<organization>:<org hierarchy>[:<org hierarchy level>]*:<schematype>:<context category>:<major>:<status>	URL:	http://<organization>/<org hierarchy>[/<org hierarchy level>]*/<schematype>/context category/<major>/<status>	3
	URN:	urn:<organization>:<org hierarchy>[:<org hierarchy level>]*:<schematype>:<context category>:<major>:<status>				
URL:	http://<organization>/<org hierarchy>[/<org hierarchy level>]*/<schematype>/context category/<major>/<status>					
<p>Where:</p> <ul style="list-style-type: none"> • organization – An identifier of the organization providing the standard. • org hierarchy – The first level of the hierarchy within the organization providing the standard. • org hierarchy level – Zero to n level hierarchy of the organization providing the standard. • schematype – A token identifying the type of schema module: data codelist documentation. • context category – The context category [business process] for UN/CEFACT from the UN/CEFACT catalogue of common business processes. Other values may be used by the other organizations. • major – The major version number. • status – The status of the schema as: draft standard. 						

697 UN/CEFACT has determined that URNs are most appropriate as persistence is of a
 698 higher priority for UN/CEFACT. Furthermore, UN/CEFACT recommends that URNs
 699 be used by other organizations that use this specification. However, each
 700 organization must decide for themselves if persistence or resolvability is more
 701 important for their namespace solution.

[R 8CED]	UN/CEFACT namespaces MUST be defined as Uniform Resource Names.	3
----------	---	---

702 Example 5-10 and 5-11 show namespace using URNs that follow the valid format for
 703 Draft and Standard specifications.

704 **Example 5-10: Namespace Name at Draft Status**

705 `"urn:un:unece:uncefact:data:ordermanagement:1:draft"`

706 **Example 5-11: Namespace Name at Specification Status**

707 `"urn:un:unece:uncefact:data:odermanagement:1:standard"`

708 UN/CEFACT namespace names include a major version identifier, therefore once a
 709 namespace's content is published; any change that breaks backward compatibility
 710 requires a new namespace. See the section on [5.9.1 Major Versions](#). Only the
 711 publisher of a namespace may change the content defined within the namespace.
 712 The publisher may only make changes that adhere to the rules defined for minor
 713 version changes defined in section [5.9.2 Minor Versions](#).

[R B56B]	Published namespace content MUST only be changed by the publishing organization of the namespace or its successor.	1
----------	--	---

714 **5.6.2 Namespace Tokens**

715 Namespace URIs are typically aliased using tokens rather than citing the entire URI
 716 for the qualifier in a qualified name for XML Schema Components within a given
 717 namespace.

718 Namespace tokens representing the namespace will be created using three
 719 character representations for each unique value within the chosen context category.

720 Additionally, XML Schema Files that are defined for Common Code List will use a
 721 token that is prefixed with 'clm' to indicate that they are Common Code List XML
 722 Schema Files.

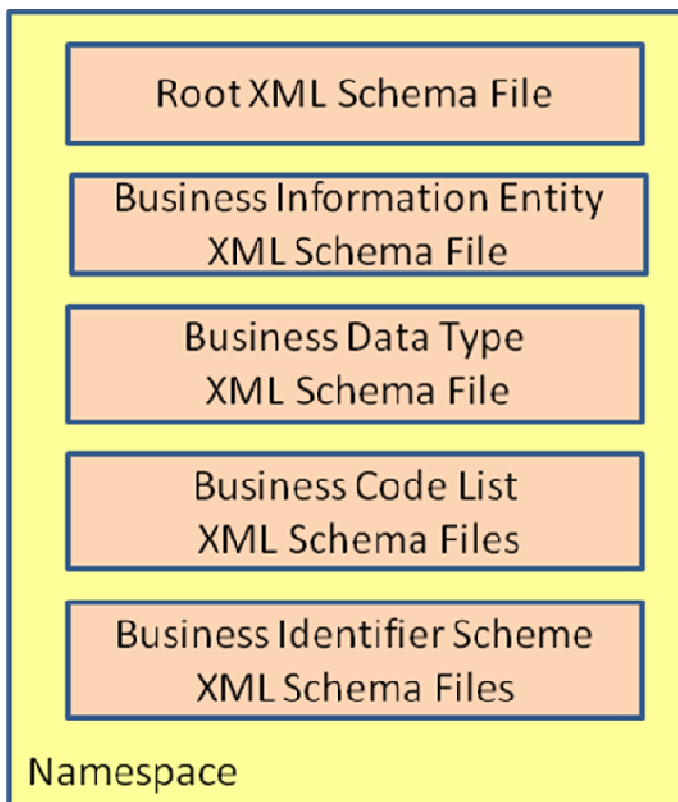
723 **5.7 XML Schema Files**

724 An XML Schema File is a schema document realized as a physical file. As defined
 725 by the W3C, a schema document represents relevant instantiations of the thirteen
 726 defined W3C XML Schema Components that collectively comprise an abstract data
 727 model.

728 For consistency, XML Schema File names will adhere to a specific pattern.

[R 92B8]	The XML Schema File name for files other than code lists and identifier schemes MUST be of the form <code><SchemaModuleName>_<Version>.xsd</code> , with periods, spaces, or other separators and the words 'XML Schema File' removed.	3
[R 8D58]	When representing versioning schemes in file names, the period MUST be represented by a lowercase <code>p</code> .	3

729 XML Schema Files can be either unique in their functionality, or represent splitting of
730 larger XML Schema Files for performance or manageability enhancement. A well
731 thoughtout approach to the layout provides an efficient and effective mechanism for
732 providing components as needed rather than dealing with complex, multi-focused
733 XML Schema Files. XML Schema Files created from this specification represent
734 abstract data models for messages, CCTS conformant ABIEs, BDTs, Business Code
735 Lists (BCL), Business Identifier Schemes (BIS), references to Common Code Lists
736 (CCL) and Common Identifier Schemes (CIS). Figure 5-6 shows how these schema
737 files are collected into relevant namespaces representing business
738 processes/information messages.



739

740 **Figure 5-6: UN/CEFACT XML Schema Files**

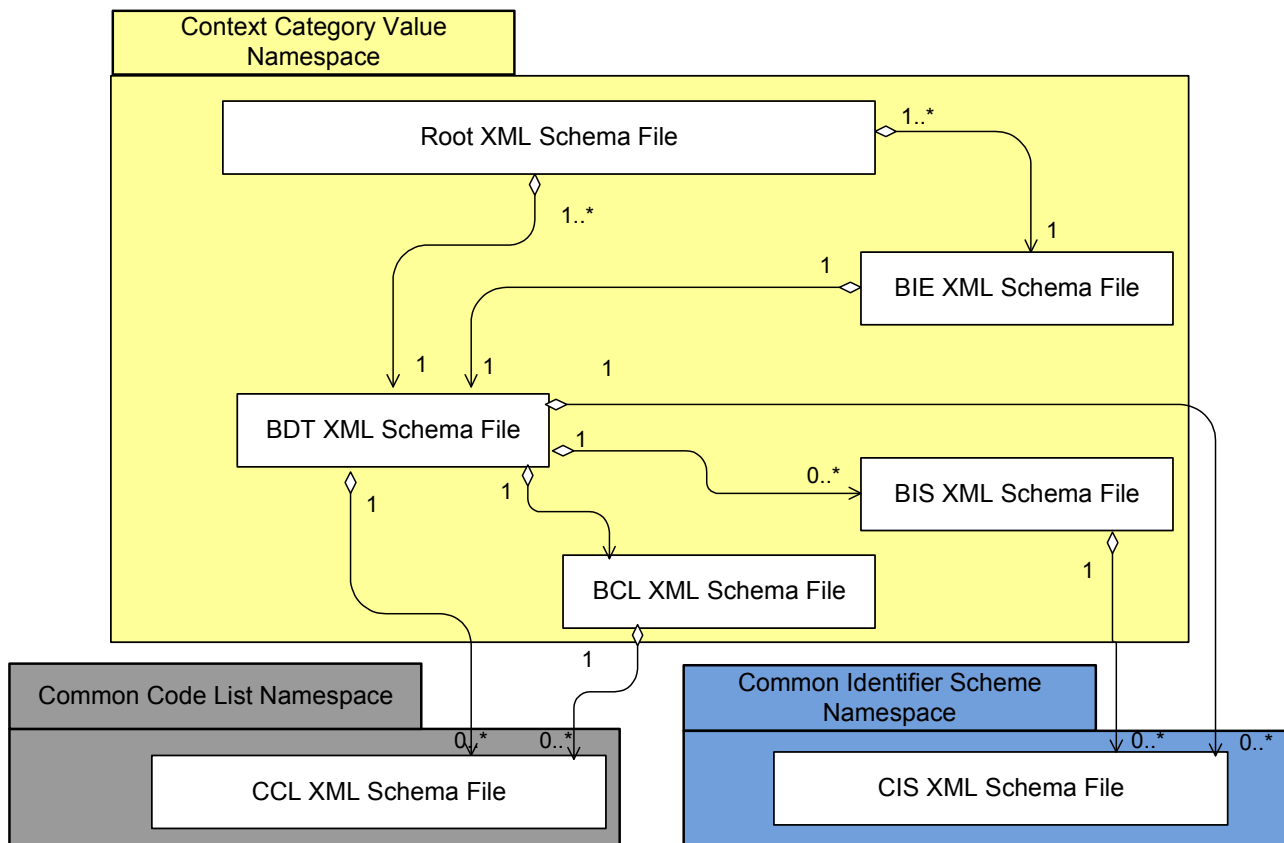
741 Each of the Root XML Schema Files defined within the given context category
742 namespace always includes the BIE XML Schema file and the BDT XML Schema
743 File. The BIE XML Schema File always includes the BDT XML Schema File. The
744 BDT XML Schema File always includes zero or more BCL XML Schema Files and
745 zero or more BIS XML Schema Files. The BDT XML Schema File also always
746 imports zero or more CCL XML Schema Files and zero or more CIS XML Schema

747 Files. The Business Code List XML Schema Files may also import a single Common
 748 Code List XML Schema File, only if it restricts the list of common codes for the given
 749 context category value for the business use case. Dependencies exist among the
 750 various files as shown in Figure 5-7. See section [8 XML Schema Files](#) and the
 751 corresponding sub-sections.

752 Each `xsd:schema` element used to define an XML Schema Document within an
 753 XML Schema File will have the namespace declared using
 754 `xsd:targetNamespace`.

[R B387]	Every XML Schema File MUST have a namespace declared, using the <code>xsd:targetNamespace</code> attribute.	1
----------	---	---

755 The contents of the set of XML Schema within a given namespace are so
 756 interrelated that proper management dictates that versioning of all members of the
 757 set be synchronized so that incompatible definitions are avoided. All schemas of the
 758 set, which are already assigned a single namespace version, are therefore
 759 additionally assigned to a single file version number.



760 **Figure 5-7: UN/CEFACT XML Schema Modularity Scheme**

761 5.7.1 Root XML Schema Files

762 As expressed in section [5.6 Namespace Scheme](#), Root XML Schema Files are
 763 assigned to a namespace that reflect the dominate context category value of the
 764 schema as shown in Figure 5-5. The determination of the dominate context category
 765 is at the discretion of the originating organization. The XML Schema File modularity
 766 scheme also calls for a set of XML Schema Files that support a Root XML Schema
 767 File. This set of XML Schema Files is also assigned to the same dominate context
 768 category namespace This approach enables the use of individual context category
 769 value focused Root XML Schema Files without importing the entire library. Each
 770 Root XML Schema File will define its own dependencies.

771 There maybe a number of UN/CEFACT Root XML Schema Files, each of which
 772 expresses a separate business information payload. The Root XML Schema Files
 773 include the recognized business transactions for the given context category based
 774 namespace.

[R 9354]	A Root XML Schema File MUST be created for each unique business information payload.	1
----------	--	---

775 To ensure uniqueness, Root XML Schema Files will have unique names based on
 776 their business function. This business function is defined in the UN/CEFACT
 777 Requirements Specification Mapping (RSM) document as the target business
 778 information payload.

[R B3E4]	Each Root XML Schema File MUST be named after the <BusinessInformationPayload> that is expressed in the XML Schema File by using the value of <BusinessInformationPayload> followed by the words ' XML Schema File ' as the name and placing the name in the Header documentation section of the file.	1
----------	---	---

779 As defined in Section 5.3, each root XML Schema File will only contain MAs and
 780 ASMA. The Root XML Schema File will not duplicate reusable XML constructs
 781 available in the other XML Schema Files in the same namespace. Instead, the root
 782 XML Schema File uses the **xsd:include** feature.

[R 9961]	A Root XML Schema File MUST NOT replicate reusable constructs available in XML Schema Files that can be referenced through xsd:include .	1
----------	---	---

783 5.7.2 Business Information Entity XML Schema Files

784 A BIE XML Schema File will be created to define all reusable BIEs within a primary
 785 context category value namespace.

786 Each BIE XML Schema File will have a standardized name that uniquely
 787 differentiates it from other UN/CEFACT XML Schema Files.

[R 8238]	A BIE XML Schema File MUST be created within each namespace	1
----------	---	---

	that is defined for the primary context category value.	
[R 8252]	The BIE XML Schema Files MUST be named 'Business Information Entity XML Schema File' by placing the name within the Header documentation section of the file.	1

788 Where desired, these BIE XML Schema Files may be further compressed for runtime
789 performance considerations if necessary through the creation of a runtime version
790 that only includes those ABIEs necessary to support the Root XML Schema File
791 including it.

792 **5.7.3 Business Data Type XML Schema Files**

793 The CCTS Business Data Types (BDTs) define the value domain for a Basic
794 Business Information Entity. The value domain is defined by selecting from one of
795 the allowed primitives for the BDT and providing additional restrictions if desired
796 through the use of Supplementary Components or a business scheme or list.

797 For reference purposes, UN/CEFACT publishes a BDT XML Schema File that
798 consists of all BDTs without restriction to the value domain. This schema file resides
799 in the documentation common namespace and is used for reference purposes only.

[R A2F0]	An unqualified BDT XML Schema File MUST be created in the documentation common namespace to represent the set of unrestricted BDTs.	1
----------	---	---

800 Additional BDT XML Schema Files that contain only the BIEs used in a primary
801 context category namespace will also be published as part of the schema set of that
802 namespace.

[R AA56]	A BDT XML Schema File MUST be created within each namespace that is defined for the primary context category value.	1
[R 847C]	The BDT XML Schema Files MUST be named 'Business Data Type XML Schema File' by placing the name within the header documentation section of the file.	1

803 **5.7.4 Code List XML Schema Files**

804 Code lists published by standards organizations represent a set of commonly
805 accepted codes for use in a variety of business circumstances and contexts. Code
806 lists can be either:

- 807 • Unrestricted by an implementation context category values, defined outside of
808 any implementation context category value and expressed as a CCL XML
809 Schema File.
- 810 • Defined by an implementation context category value and expressed as a
811 BCL XML Schema File.

812 Some owning organizations such as UN/CEFACT publish these code lists as an
 813 XML Schema File, others do not. The modularity model calls for each code list to be
 814 expressed in an XML Schema File. If an external published code list that conforms
 815 to the rules of this specification is not already available as an XML Schema File,
 816 then a CCL XML Schema File will be created.

[R 8A68]	A Code List XML Schema File MUST be created to convey code list enumerations for each code list being used.	1
[R B0AD]	<p>The name of each Code List XML Schema File as defined in the comment within the XML Schema File MUST be of the form:</p> <p><Code List Agency Identifier Code List Agency Name><Code List Identification Identifier Code List Name>” - Code List XML Schema File”</p> <p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Identifier – Identifies the agency that maintains the code list. • Code List Agency Name – Agency that maintains the code list. • Code List Identification Identifier – Identifies a list of the respective corresponding codes. • Code List Name – The name of the code list as assigned by the agency that maintains the code list. 	1

817 Example 5-12 shows an example of using the CCL Identifiers to name the Code List
 818 XML Schema File as described in Rule [R B0AD].

819 **Example 5-12: Name of UN/CEFACT Account Type Code List XML Schema File Name using**
 820 **Identifiers**

```
821 64437 - Code List XML Schema File
822 where:
823 6 = Code list agency identifier for UN/CEFACT as defined in UN/CEFACT code
824 list 3055
825 4437 = Code list identification identifier for Account Type Code in UN/CEFACT
826 directory
```

827 Example 5-13 shows an example of using the CCL Names to name the Code List
 828 XML Schema File as described in Rule [R B0AD].

829 **Example 5-13: Name of UN/CEFACT Security Type Code List XML Schema File Name using**
 830 **Names**

```
831 Security Initiative Document Security Code - Code List XML Schema File
```

832 Additional examples of CCL XML Schema Files can be found at the [UN/CEFACT](#)
 833 [Web site](#).

834 **5.7.4.1 Common Code List XML Schema Files**

835 A code list is considered common if it is published by a recognized standards
 836 organization for use across a broad spectrum of contexts. UN/CEFACT will prepare
 837 a CCL for each common code list used by a BDT. Each CCL XML Schema File will
 838 contain enumerated values for codes and code values.

[R 942D]	Each CCL XML Schema File MUST contain enumeration values for both the actual codes and the code values.	1
----------	---	---

839 **5.7.4.2 Business Code List XML Schema Files**

840 A BCL may be created for a BDT. The BCL can be a restriction or extension to the
 841 set of codes in a CCL, be a new code list, or be a union of code lists.. All BCLs are
 842 expressed as individual XML Schema Files and are assigned to the same
 843 namespace as the XML Schema Files that make use of them. If a BDT that
 844 references a BCL is used in different namespaces, then a BDT will be defined and a
 845 BCL will be included in each namespace.

846 Each BCL XML Schema File contains enumerated values for codes and their code
 847 values. These enumerated values may be a part of a restriction of a CCL, as a new
 848 Code List for the given context category, or as an extension to an existing CCL.

[R A8A6]	<p>Each BCL XML Schema File MUST contain enumeration values for both the actual codes and the code values, through one of the following:</p> <ul style="list-style-type: none"> • The restriction of an imported CCL. • The extension of a CCL where the codes and values of the CCL are included and the new extensions are added. • The creation of a new Code List that is used within the context category value namespace. 	1
----------	--	---

849 **5.7.5 Identifier Schemes**

850 Identifier schemes are different than code lists in both concept and functionality.
 851 Whereas a code has a value, an identifier is a pointer that is typically devoid of any
 852 specific value. Code lists are enumerated lists. Identifier schemes are typically not
 853 enumerated.

854 Identifier schemes will be defined as simple types without enumeration in an
 855 Identifier Scheme XML Schema File following the same approach as is used for
 856 code lists.

[R AB90]	An Identifier Scheme XML Schema File MUST be created to convey identifier scheme metadata for each scheme being used.	1
[R A154]	The name of each Identifier Scheme XML Schema File as defined in the comment within the XML Schema File MUST be of the form: <Identifier Scheme Agency Identifier Identifier	1

	<p>Scheme Agency Name><Identifier Scheme Identification Identifier Identifier Scheme Name>” Identifier Scheme XML Schema File”</p> <p>Where:</p> <ul style="list-style-type: none"> • Identifier Scheme Agency Identifier – Identifies the agency that maintains the identifier scheme. • Identifier Scheme Agency Name – Agency that maintains the identifier scheme. • Identifier Scheme Identification Identifier – Identifies the scheme. • Identifier Scheme Name – The name of the identifier scheme as assigned by the agency that maintains the identifier scheme. 	
--	--	--

857 **Example 5-14: Name of GS1 Global Trade Item Number Identifier Scheme XML Schema File**
858 **Name using Identifiers**

```

859 9GTIN - Code List XML Schema File
860 where:
861 6 = Agency identifier for GS1 as defined in UN/CEFACT code
862 list 3055
863 GTIN = GS1 Identification identifier for Global Trade Item Number

```

864 5.7.5.1 Common Identifier Scheme

865 A common identifier scheme is one that is used for a broad audience in multiple
866 business processes. Common schemes are typically formally published as metadata
867 which fully describe them to enable development of conformant identifiers.

868 5.7.5.2 Business Identifier Scheme

869 A business scheme may be defined for a BDT. In cases where some codes in the
870 source CCL are not needed in the business process, the BCL will be a restriction to
871 the CCL. All BCLs are expressed as individual XML Schema Files and are assigned
872 to the same namespace as the XML Schema Files that make use of them. If a BDT
873 that references a BCL is used in different namespaces, then a BDT will be defined
874 and a BCL will be included in each namespace.

[R BD2F]	A Business Identifier Scheme XML Schema File MUST be created for each Business Scheme used by a BDT.	1
----------	--	---

875 Each Business Scheme XML Schema File contains metadata regarding the scheme.
876 If a business scheme is a restriction on a common scheme, the nature of the
877 restriction will be included in the metadata as a business rule in an xsd:annotation
878 xsd:appInfo element.

[R AFEB]	Each Business Identifier Scheme XML Schema File MUST contain metadata that describes the scheme or points to the scheme.	1
----------	--	---

879 **5.7.6 Other Standard Bodies BIE XML Schema Files**

880 Other Standards Development Organizations (SDO) create and make publicly
 881 available BIE XML Schema Files. UN/CEFACT will only import these other SDO BIE
 882 XML Schema Files when their contents are in strict conformance to the requirements
 883 of the CCTS technical specification and this NDR technical specification. Strict
 884 conformance means that a schema is conformant to category 1, 2, 3, 4 and 7 rules
 885 as defined in rule [\[R B998\]](#).

886 In order to achieve interoperability it is critical that these components are consistently
 887 represented regardless of which organization they originate.

[R B564]	Imported XML Schema Files MUST be fully conformant to category 1, 2, 3, 4 and 7 rules as defined in rule [R B998] .	4
[R 9733]	Imported XML Schema File components MUST be derived using these NDR rules from artifacts that are fully conformant to the latest version of the UN/CEFACT Core Components Technical Specification.	4

888 **5.8 Schema Location**

889 Schema locations:

- 890 • Are required to be in the form of a URI scheme;
- 891 • Are associated to the namespace of the file being accessed;
- 892 • Are typically defined as URLs because of resolvability limitations of URNs;
- 893 • Can be defined as absolute path or relative paths.

894 According to the W3C XML Schema specification, part 0, the `schemaLocation`
 895 attribute "... provides hints from the author to a processor regarding the location of a
 896 schema document. The author warrants that these schema documents are relevant
 897 to checking the validity of the document content, on a namespace by namespace
 898 basis."³ The value provided in the `xsi:schemaLocation` attribute is "...only a hint
 899 and some processors and applications will have reasons to not use it." Thus the
 900 presence of these hints does not require the processor to obtain or use the cited
 901 schema documents, and the processor is free to use other schemas obtained by any
 902 suitable means, or to use no schema at all.

903 In practical implementations XML tools attempt to acquire resources using the
 904 schema location attribute. The implication of the `xsi:schemaLocation` attribute
 905 pointing to an absolute path (e.g., hard-drive location; URL) is that when tools
 906 attempt to acquire the resources and they are not available at the specified location,
 907 the tool may raise errors. In the case of URL-formatted `xsi:schemaLocation`
 908 values, this might occur after a seemingly lengthy timeout period, a period in which
 909 other work cannot be done. On the other hand, relative paths increase the likelihood
 910 that resources will be readily available to tools (assuming well organized schema

³ <http://www.w3.org/TR/xmlschema-0/#schemaLocation>

911 files). Thus using an absolute path approach with URL-formatted
 912 `xsi:schemaLocation` values often represents a challenge in practical
 913 implementations as it requires open internet connections at run-time (due to tool
 914 implementations) and is seen as a security issue by a number of implementers.

915 Providing the `schemaLocation` value as a relative path provides an overall
 916 improvement in user productivity, including off-line use. It is important to note that
 917 this approach doesn't prohibit making resources available on-line (much in the same
 918 way that HTML documents frequently provided references to relative locations for
 919 images).

[R 8F8D]	Each <code>xsd:schemaLocation</code> attribute declaration within an XML Schema File MUST contain a resolvable relative path URL.	2
----------	---	---

920 **Example 5-16: Relative path `schemaLocation`.**

921

```
<xsd:import namespace="urn:un:unece:uncefact:ordermanagementdata:draft:1"  

  922 schemaLocation="../../../data/draft/BusinessDataType_1p0.xsd"/>
```

923 5.9 Versioning Scheme

924 The UN/CEFACT versioning scheme consists of:

- 925 • Status of the XML Schema File,
- 926 • A major version number,
- 927 • A minor version number and
- 928 • A revision number.

929 These values are declared in the version attribute in the `xsd:schema` element.
 930 The major version number is also reflected in the namespace declaration for
 931 each XML Schema File rule [\[R 8E2D\]](#).

[R BF17]	The <code>xsd:schema</code> version attribute MUST always be declared.	1
[R 84BE]	<p>The <code>xsd:schema</code> version attribute MUST use the following template:</p> <pre><xsd:schema ... version=" <major>"p"<minor>["p"<revision>]"></pre> <p>Where:</p> <ul style="list-style-type: none"> • <code><major></code> - sequential number of the major version. • <code><minor></code> - sequential number of the minor version • <code><revision></code> - optional sequential number of the revision. 	2

932 5.9.1 Major Versions

933 A major version of a UN/CEFACT XML Schema File constitutes significant non-
 934 backwards compatible changes. If any XML instance based on an older major
 935 version of UN/CEFACT XML Schema attempts validation against a newer version, it

936 may experience validation errors. A new major version will be produced whenever
937 non-backward compatible changes occur. This would include the following changes:

- 938 • Removing or changing values in enumerations.
- 939 • Changing of element names, type names and attribute names.
- 940 • Changing the structures so as to break polymorphic processing capabilities.
- 941 • Deleting or adding mandatory elements or attributes.
- 942 • Changing cardinality from optional to mandatory.

943 Major version numbers will be based on logical progressions to ensure semantic
944 understanding of the approach and guarantee consistency in representation. Non-
945 negative, sequentially assigned incremental integers satisfy this requirement.

[R 9049]	Every XML Schema File major version number MUST be a sequentially assigned incremental integer greater than zero.	1
----------	---	---

946 5.9.2 Minor Versions

947 The minor versioning of an XML Schema File identifies its compatibility with the
948 preceding and subsequently minor versions within the same major version.

949 Within a major version iteration of a UN/CEFACT XML Schema File there could
950 potentially be a series of minor, or backward compatible, changes. Each minor
951 version will be compatible with both preceding and subsequent minor versions within
952 the same major version. The minor versioning scheme thus helps to identify
953 backward and forward compatibility. Minor versions will only be increased when
954 compatible changes occur, i.e

- 955 • Adding values to enumerations.
- 956 • Optional extensions.
- 957 • Add optional elements.

[R A735]	Minor versioning MUST be limited to declaring new optional XML content, extending existing XML content, or refinements of an optional nature.	1
----------	---	---

958 Minor versions will be declared using the `xsd:version` attribute in the
959 `xsd:schema` element. It is only necessary to declare the minor version in the
960 schema version attribute since instance documents with different minor versions are
961 compatible with the major version held in the same namespace. By using the version
962 attribute in each document instance, the application can provide the appropriate logic
963 switch for different compatible versions without having knowledge of the schema
964 version which the document instance was delivered.

965 Compatibility includes consistency in naming of the schema constructs to include
966 elements, attributes, and types. UN/CEFACT minor version changes will not include
967 renaming XML Schema constructs.

968 For a particular namespace, the major version and subsequent minor versions and
969 revisions create a linear relationship.

[R AFA8]	Minor versions MUST NOT rename existing XML Schema defined artifacts.	1
[R BBD5]	Changes in minor versions MUST NOT break semantic compatibility with prior versions having the same major version number.	1

970 For a particular namespace, the major version and subsequent minor versions and
971 revisions create a linear relationship.

[R 998B]	XML Schema Files for a minor version XML Schema MUST incorporate all XML Schema components from the immediately preceding version of the XML Schema File.	1
----------	---	---

972 6 Application of Context

973 The intent of this NDR is to express everything that is necessary in a UN/CEFACT
974 XML Schema to enable integration of business information within an XML Schema
975 conformant XML instance message. To accomplish this, the XML Schema will
976 address all aspects of the business information to include:

- 977 • Business semantics – The meaning of business information in
978 communication.
 - 979 ○ Meaning can vary between different individuals depending on the
980 context of the sender and the receiver of the information.
 - 981 ○ Meaning can be the same between different individuals depending on
982 the context of the sender and the receiver of the information.
- 983 • Business context – The circumstances that determine the meaning of
984 business information. The business context may change the semantic
985 meaning for the sender and or the receiver of the information.

986 In CCTS, BIEs represent context specific artifacts for a message. CCTS defines
987 different context categories that capture context category values. BIE artifacts may
988 be defined within any number of combinations of context categories and context
989 category values within a category. BIEs may have the same name with different
990 context values and different content models. As identified in Section 5.6, the
991 namespace mechanism using the primary context category will ensure name
992 collision of similarly named components in different contexts does not occur.

993 [Note:]

994 It is possible to extend the namespace described in section [5.6 Namespace Scheme](#)
995 for an implementation set of schemas to include a Context Identifier on the end of
996 the namespace to express the full context of the reduced set of XML Schemas.
997 While this Context Identifier is out side the scope of this technical specification, it is
998 recommended that this identifier be a Univerisally Unique Identifier (UUID).

999 In addition to the primary context category, all other context category values for
1000 every BIE is expressed within the XML Schema definition for each XML Schema
1001 Component as an `xsd:appInfo` declaration following the structure defined in
1002 section [7.5.2 Application Information \(AppInfo\)](#).

1003 7 General XML Schema Definition Language Conventions

1004 The XML Schema language has many constructs that can be used to express a
1005 model. The purpose of this section is to provide a profile and set of rules based on
1006 general best practices for those constructs that can be used and to identify those
1007 constructs that should not be used to include:

- 1008 • Overall XML Schema Structure and Rules
- 1009 • Attribute and Element Declarations
- 1010 • Type Definitions
- 1011 • Use of Extension and Restriction
- 1012 • Annotation

1013 7.1 Overall XML Schema Structure and Rules

1014 7.1.1 XML Schema Declaration

1015 As required by XSD, when defining an XML Schema file the first line indicates the
1016 xml version and the encoding it uses. UN/CEFACT XML Schema will use UTF-8
1017 encoding.

[R 88E2]	Every UN/CEFACT XML Schema File MUST use UTF-8 encoding.	1
----------	--	---

1018 Example 7-1 shows the declaration of encoding for the XML Schema document.

1019 **Example 7-1: XML Schema File Line 1 setting the XML version and encoding**

```
1020 <?xml version="1.0" encoding="UTF-8"?>
```

1021 7.1.2 XML Schema File Identification and Copyright Information

1022 After the first line of the schema documentation in the form of `xsd:comment` lines
1023 will appear. These comments are applicable to the XML Schema file. The template
1024 for this is shown in [Appendix B in section B.2](#)

[R ABD2]	Every XML Schema File MUST contain a comment that identifies its name immediately following the XML declaration using the format defined in Appendix B-2 .	1
[R BD41]	Every XML Schema File MUST contain a comment that identifies its owning agency, version and date immediately following the schema name comment using the format defined in Appendix B-2 .	1

1025 7.1.3 Schema Declaration

1026 The `xsd:schema` element is declared to define an XML Schema document. The
1027 `xsd:schema` element includes attributes that affect how the rest of the document
1028 behaves and how XML parsers and other tools treat it. The XML Schema
1029 Component will have:

- 1030 • `elementFormDefault` set to qualified.
- 1031 • `attributeFormDefault` set to unqualified.
- 1032 • The prefix `xsd` used to refer to the XML Schema namespace.

[R A0E5]	The <code>xsd:elementFormDefault</code> attribute MUST be declared and its value set to qualified.	1
[R A9C5]	The <code>xsd:attributeFormDefault</code> attribute MUST be declared and its value set to unqualified.	1
[R 9B18]	The <code>xsd</code> prefix MUST be used in all cases when referring to the namespace <code>http://www.w3.org/2001/XMLSchema</code> as follows: <code>xmlns:xsd=http://www.w3.org/2001/XMLSchema</code> .	1

1033 Example 7-2 shows a XML Schema snippet declaring `schema` component, set the
 1034 namespace token to `xsd`, set the `elementFormDefault` to qualified and set the
 1035 `attributeFormDefault` to unqualified.

1036 **Example 7-2: Element and Attribute Form Default**

```
1037 <xsd:schema targetNamespace=" ... see namespace ...
1038     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1039     elementFormDefault="qualified" attributeFormDefault="unqualified">
```

1040 **7.1.4 CCTS Artifact Metadata**

1041 CCTS defines specific metadata associated with each CCTS artifact. This metadata
 1042 will be defined in a separate CCTS Metadata XML Schema File.

1043 The CCTS XML Schema File will be named Core Components Technical
 1044 Specification Schema File.

1045 The CCTS XML Schema File will be assigned to its own namespace and use a prefix
 1046 of `ccts`.

[R 90F1]	All required CCTS metadata for ABIEs, BBIEs, ASBIEs, and BDTs must be defined in an XML Schema File.	1
[R 9623]	The name of the CCTS Metadata XML Schema file will be “Core Components Technical Specification Schema File” and will be defined within the header comment within the XML Schema File.	1
[R 9443]	The CCTS Metadata XML Schema File MUST reside in its own namespace and be defined in accordance with rule [R 8E2D] and assigned the prefix <code>ccts</code> .	1

1047 **7.1.5 Constraints on Schema Construction**

1048 In addition to general XML Schema structure, constraints on certain XML Schema
 1049 rules are necessary to ensure maximum interoperability for business-to-business
 1050 and application-to-application interoperability.

[R AD26]	xsd:notation MUST NOT be used.	1
[R ABFF]	The xsd:any element MUST NOT be used.	4 6
[R AEBB]	The xsd:any attribute MUST NOT be used.	4 6
[R 9859]	Mixed content MUST NOT be used.	1
[R B20F]	xsd:redefine MUST NOT be used.	4 6
[R 926D]	xsd:substitutionGroup MUST NOT be used.	4 6
[R 8A83]	xsd:ID/xsd:IDREF MUST NOT be used.	1

1051 **7.2 Attribute and Element Declarations**1052 **7.2.1 Attributes**

1053 Attributes are only used to convey BDT supplementary components as part of a BDT
 1054 **xsd:type** definition. Where the **xsd:attributes** of an XSD data type definition in XSD
 1055 part two exist, the BDT will use the **xsd** data type as its base type and will use the
 1056 **xsd:attributes** to represent supplementary components. Where this is not the case,
 1057 user defined attributes will be declared to represent supplementary components.

[R B221]	Supplementary Components MUST be declared as Attributes.	1
[R AFEE]	User defined attributes MUST only be used for Supplementary Components.	3
[R 9FEC]	An xsd:attribute that represents a Supplementary Component with variable information MUST be based on an appropriate XML Schema built-in simpleType .	1
[R B2E8]	A xsd:attribute that represents a Supplementary Component which uses codes MUST be based on the xsd:simpleType of the appropriate code list.	1
[R 84A6]	A xsd:attribute that represents a Supplementary Component which uses identifiers MUST be based on the xsd:simpleType of the appropriate identifier scheme.	1

1058 **7.2.2 Elements**

1059 Elements are declared for the document level business information payload, ABIEs,
1060 BBIEs, and ASBIEs whose aggregationKind=shared.

1061 **7.2.2.1 Element Declaration**

1062 Every `ccts:BBIE` artefact is declared as an `xsd:element` of the simple or
1063 complex type that instantiates its BDT.

1064 **7.2.2.2 Empty Elements**

1065 In general, the absence of an element in an XML document does not have any
1066 particular meaning - it may indicate that the information is unknown, or not
1067 applicable, or the element may be absent for some other reason. The XML Schema
1068 specification does provide a feature, the `xsd:nillable` attribute, whereby an
1069 element may be transferred with no content, with a `xsi:nil` attribute to indicate that
1070 it is intentionally empty.

1071 In order to respect the principles of the CCTS and to retain semantic clarity, empty
1072 elements and the nillability feature of XML Schema will not be used by UN/CEFACT
1073 XML Schemas.

[R B8B6]	Empty elements MUST NOT be used.	3
[R 8337]	The <code>xsd:nillable</code> attribute MUST NOT be used.	1

1074 **7.3 Type Definitions**

1075 An XML Schema Type defines simple and complex structures used to define an
1076 element.

1077 All elements declared will have a named type that provides the definition of the
1078 structure of the XML Schema Component using it.

[R 8608]	Anonymous types MUST NOT be used.	1
----------	-----------------------------------	---

1079 **7.3.1 Simple Type Definitions**

1080 `xsd:simpleTypes` must always be used where they satisfy the user's business
1081 requirements. Examples 7-3 shows a simple type defined in the BDT XML Schema
1082 File.

1083 **Example 7-3: Simple Types in Business Data Type XML Schema File**

```
1084 <xsd:simpleType name="DateTimeType">
1085   <xsd:annotation>
1086     ... see annotation ...
1087   </xsd:annotation>
1088   <xsd:restriction base="xsd:dateTime"/>
1089 </xsd:simpleType>
```

1090 Example 7-4 shows a simple type defined in a Code List XML Schema File.

1091 **Example 7-4: Simple Types in a Code Lists XML Schema File**

```

1092 <xsd:simpleType name="CurrencyCodeContentType">
1093   <xsd:restriction base="xsd:token">
1094     <xsd:enumeration value="ADP">
1095       ...see enumeration of code lists ...
1096     </xsd:enumeration>
1097   <xsd:annotation>
1098     ... see annotation ...
1099   </xsd:annotation>
1100 </xsd:restriction>
1101 </xsd:simpleType>

```

1102 7.3.2 Complex Type Definitions

1103 A complex type will be defined to express the content model of each CCTS ABIE. A
 1104 complex type will also be defined to express the value domain of a CCTS BDT when
 1105 an XML Schema built-in data type does not meet the business requirements.

[R A4CE]	An xsd:complexType MUST be defined for each CCTS ABIE.	1
[R BC3C]	An xsd:complexType MUST be defined for each CCTS BDT that cannot be fully expressed using an xsd:simpleType .	1

1106 Example 7-5 shows a complex type defined for an Account ABIE.

1107 **Example 7-5: Complex Type of Object Class "AccountType"**

```

1108 <xsd:complexType name="AccountType">
1109   <xsd:annotation>
1110     ... see annotation ...
1111   </xsd:annotation>
1112   <xsd:sequence>
1113     ... see element declaration ...
1114   </xsd:sequence>
1115 </xsd:complexType>

```

1116 In order to increase consistency in use and enable accurate and complete
 1117 representation of what is allowed in the design of CCTS artefacts, the **xsd:sequence**
 1118 and **xsd:choice** compositors will be used to express the content model for
 1119 **xsd:complexType** definitions. The **xsd:a11** XML Schema compositor will not be
 1120 used.

[R A010]	The xsd:a11 element MUST NOT be used.	1
----------	--	---

1121 7.4 Use of Extension and Restriction

1122 In keeping with CCTS, XML Schema Components are based on the concept that the
 1123 underlying semantic structures of the BIEs are normative forms of standards that
 1124 developers are not allowed to alter without coordination with the owner of the
 1125 component at the data model level. As business requirements dictate, new BIE
 1126 artifacts will be created in the data model and represented as XML Schema
 1127 Components by defining new types and declaring new elements. The concept of
 1128 derivation from existing types through the use of **xsd:extension** and

1129 **xsd:restriction** will only be used in limited circumstances where their use does
1130 not violate this principle.

1131 It is understood that other standards organizations using this specification may
1132 choose to use **xsd:extension** and/or **xsd:restriction** to define new
1133 constructs that are extended or restricted from existing constructs.

1134 7.4.1 Extension

1135 UN/CEFACT XML Schema Files may only use **xsd:extension** in the BDT XML
1136 Schema File to declare attributes to accommodate supplementary components.
1137 **xsd:extension** will only be used in an **xsd:complexType** within the BDT XML
1138 Schema File, and only for declaring attributes to support supplementary
1139 components.

[R AB3F]	xsd:extension MUST only be used in the BDT XML Schema File.	4 6
[R 9D6E]	xsd:extension MUST only be used for declaring xsd:attributes to accommodate relevant supplementary components.	4 6

1140 Example 7-6 shows an extension of a simple type using the **xsd:extension**
1141 mechanism.

1142 Example 7-6: Extension of Simple Type

```

1143 <xsd:complexType name="AmountType">
1144   <xsd:annotation>
1145     ... see annotation ...
1146   </xsd:annotation>
1147   <xsd:simpleContent>
1148     <xsd:extension base="xsd:decimal">
1149       <xsd:attribute name="unitCode" type="xsd:token"/>
1150     </xsd:extension>
1151   </xsd:simpleContent>
1152 </xsd:complexType>

```

1153 7.4.2 Restriction

1154 The CCTS specification employs the concept of semantic restriction in creating
1155 specific instantiations of core components. Accordingly, **xsd:restriction** will be
1156 used as appropriate to define qualified BDT types that are derived from less qualified
1157 or unqualified BDT types.

[R 9947]	xsd:restriction MUST only be used in BDT XML Schema Files.	1
----------	---	---

1158 Where used, the derived types must always be named uniquely. Simple and
1159 complex type restrictions may be used. **xsd:restriction** can be used for facet
1160 restriction and/or attribute restriction.

[R 8AF7]	When xsd:restriction is applied to a data type the resulting	1
----------	---	---

	type MUST be uniquely named.	
--	------------------------------	--

1161 Example 7-7 shows a restriction of a simple type.

1162

1163 **Example 7-7: Restriction of Simple Type**

```

1164 <xsd:simpleType name="TaxAmountType">
1165   <xsd:annotation>
1166     ... see annotation ...
1167   </xsd:annotation>
1168   <xsd:restriction base="bdt:AmountType">
1169     <xsd:totalDigits value="10"/>
1170     <xsd:fractionDigits value="3"/>
1171   </xsd:restriction>
1172 </xsd:simpleType>

```

1173 **7.5 Annotation**

1174 All UN/CEFACT XML Schema constructs will use the `xsd:documentation` and
 1175 `xsd:appInfo` elements within an `xsd:annotation` to provide CCTS artifact
 1176 metadata and context values.

[R 847A]	Each defined or declared construct MUST use the <code>xsd:annotation</code> element for required CCTS documentation and application information to communicate context.	1
----------	---	---

1177 **7.5.1 Documentation**

1178 The annotation `xsd:documentation` will be used to convey the metadata specified
 1179 by CCTS for CCTS artefacts. Conversely, all elements specified within an
 1180 `xsd:documentation` element will be limited to expressions of CCTS artifact
 1181 metadata.

1182 The following annotations are required as defined in each of the sub-sections in the
 1183 section [8 XML Schema Files](#) that correspond to the different CCTS artifacts.

- 1184 • **UniqueID** – The unique identifier assigned to the artefact in the library.
 1185 (UniqueID)
 - 1186 ○ The UniqueID is based on EntityUniqueIdentifierType, which refers to
 1187 the schema module "CCIS1 Entity Unique Identification Scheme" that
 1188 provides the suggested schema pattern: "UNBE0-9^{*}{6}"
- 1189 • **VersionID** – The unique identifier assigned to the version of the artefact in the
 1190 library.
 - 1191 ○ The VersionID is based on VersionIdentifierType, which refers to the
 1192 scheme module "CCTS4 Versioning Identification Scheme" that
 1193 provides the suggested schema pattern: 0-9^{*}{1,2}\.0-9^{*}{2}
- 1194 • **ObjectClassQualifierName** – Is a word or words which help define and
 1195 differentiate an ABIE from its associated CC and other BIEs. It enhances the
 1196 semantic meaning of the DEN to reflect a restriction of the concept, conceptual
 1197 domain, content model or data value.
- 1198 • **ObjectClassTermName** – Is a semantically meaningful name for the object
 1199 class. It is the basis for the DEN.
- 1200 • **Cardinality** – Indicates the cardinality of the associated artifact.

- 1201 • **SequencingKey** – Indicates the sequence of the associated artifact within the
1202 larger BIE.
- 1203 • **DictionaryEntryName** – The Data Dictionary Entry Name (DEN) of the
1204 supplementary component or business information payload. (Name)
- 1205 • **Definition** – The semantic meaning of the artefact. (Definition)
- 1206 ○ The Definition is based on BDT "TextType". The language
1207 representation should follow the same approach as described for
1208 name.
- 1209 • **BusinessTermName** – A synonym term under which the artifact is commonly
1210 known and used in business. (BusinessTerm)
- 1211 • **AssociationType** – Indicates the UML Association Kind shared or
1212 composition of the ABIE being associated in the ASBIE.
- 1213 • **PropertyTermName** – Represents a distinguishing characteristic of the object
1214 class and shall occur naturally in the definition.
- 1215 • **PropertyQualifierName** – Is a word or words which help define and
1216 differentiate a property. It further enhances the semantic meaning of the
1217 property.
- 1218 • **RepresentationTermName** – An element of the component name which
1219 describes the form in which the component is represented.
- 1220 • **AssociatedObjectClassTermName** – The Associated Object Class Term
1221 represented by the artefact.
- 1222 • **AssociatedObjectClassQualifierTerm** – A term(s) that qualifies the
1223 Associated Object Class Term.
- 1224 • **PrimitiveTypeName** – The name of the primitive type name from the Data
1225 Type Catalogue.
- 1226 • **DataTypeName** – The name of the DataType. This DataType is defined in the
1227 Data Type Catalogue.
- 1228 • **DataTypeQualifierName** – Is a word or words which help define and
1229 differentiate a Data Type. It further enhances the semantic meaning of the
1230 DataType.
- 1231 • **DefaultIndicator** – Indicates that the specific Code List Value is the default
1232 for the Code List.
- 1233 • **DefaultValue** – Is the default value.
- 1234 • **DefaultValueSource** – The source for the default value.
- 1235 • **SchemeOrListID** – The unique identifier assigned to the scheme or list that
1236 uniquely identifies it.
- 1237 • **SchemeOrListAgencyID** – The unique identifier assigned to the Agency that
1238 owns or is responsible for the Scheme or Code List being referenced.
- 1239 • **SchemeOrListAgencyName** – The name of the Agency that owns or is
1240 responsible for the Scheme or Code List being referenced.

- 1241 • **SchemeOrListModificationAllowed Indicator** – Indicates whether the
 1242 values being validated can be outside the enumerations specified by the
 1243 Scheme or Code List.
- 1244 • **SchemeOrListName** – Name of the Scheme or Code List.
- 1245 • **SchemeOrListBusinessTermName** – A synonym term under which the
 1246 Scheme or Code List is commonly known and used in business.

1247 Table 7-1 provides a summary view of the annotation data as defined in this section
 1248 and the CCTS artifacts in which each is expressed within the resulting XML Schema.

1249 [Note:]

1250 It is important to realize that while this specification lists these artifacts for the
 1251 documentation there are different types of classes. RSM, ABIE, BBIE, ASBIE and
 1252 BDT are all Registry Classes in that they are uniquely identifiable within the Core
 1253 Component Library (CCL).

Basic Business Information Entity, Association Business Information Entity, Code List, Code List Value and Supplementary Components are not Registry Classes therefore the do not include the UniqueID or VersionID from the	rsm:RootSchema	ABIE xsd:complexType	BBIE xsd:element	ASBIE: xsd:element	bdt:BusinessDataType	Supplementary Component	Code List	Code List Value
Unique ID	M	M	M	M	M			
Version ID	M	M	M	M	M			
Object Class Qualifier Name	O R	O R						
Object Class Term Name	M	M						
Cardinality			M	M		M		
Sequencing Key			M	M				
Dictionary Entry Name	M	M	M	M	M			
Definition	M	M	M	M	M			
Business Term Name	O R	O R	O R	O R	O R			
Association Type				M				
Property Term Name			M	M	M	M		
Property Qualifier Name			O R	O R				

Representation Term Name			M			M		
Associated Object Class Term Name				M				
Associated Object Class Qualifier Term Name				O R				
Primitive Type Name						M		
Data Type Name					M	M		
Data Type Qualifier Name					M	M		
Default Indicator					M	M		
Default Value					O	O		
Default Value Source					O	O		
Scheme Or List ID					O	O	M	
Scheme Or List Version ID					O	O	O	
Scheme Or List Agency ID					O	O	O	
Scheme Or List Agency Name					O	O	O	
Scheme Or List Modification Allowed Indicator					O	O	M	
Scheme Or List Name					O	O	O	O
Scheme Or List Business Term Name					O R	O R	O R	O R
Key: M – Mandatory O – Optional R – Repeating Yellow Shading – Not expressed in XML Schema								

1254 **Table 7-1 Annotation Data Summary**

1255 [Section 8 XML Schemas](#) and [Appendix F](#) specify normative information for the
1256 specific annotation required for each of the CCTS artifacts.

1257 This documentation is intended to be used to connect the XML Schema defined
1258 artifact to the model artifact in which it is based. This is important for standard XML
1259 Schemas and for fully expressed XML Schemas for a runtime implementation.

1260 However, XML Schemas directly used in a runtime implementation may choose not
 1261 to include this documentation in order to reduce the size of the XML Schema. This is
 1262 often done in order to increase the throughput of XML Instances and to increase the
 1263 sheer volume. If this is done the runtime XML Schemas may only be an exact copy
 1264 of the fully documented XML Schemas with only the annotation documentation
 1265 (`xsd:documentation`) elements removed.

[R A9EB]	Each defined or declared construct MUST use an <code>xsd:annotation</code> and <code>xsd:documentation</code> element for required CCTS documentation.	3
----------	--	---

1266 As identified in section [7.1.4 CCTS Artifact Metadata](#), the required elements are
 1267 declared in the CCTS Metadata XML Schema File. This file will be imported in all
 1268 Root, BIE, BDT and Code List XML Schema Files in lieu of re-declaring these
 1269 `xsd:documentation` elements.

1270 Example 7-8 provides an example of annotation documentation for an ABIE that
 1271 conforms to the ccts structure.

1272 **Example 7-8: Example of Annotation Documentation of an ABIE**

```

1273 <xsd:annotation>
1274   <xsd:documentation xml:lang="en">
1275     <ccts:UniqueID>UNBE000000</ccts:UniqueID>
1276     <ccts:VersionID>1.0</ccts:VersionID>
1277     <ccts:ObjectClassQualifierName>Customer</ccts:ObjectClassQualifierName>
1278     <ccts:ObjectClassTermName>Account</ccts:ObjectClassTermName>
1279     <ccts:DictionaryEntryName>Customer. Account</ccts:DictionaryEntryName>
1280     <ccts:Definition>The Customer Account.</ccts:Definition>
1281   </xsd:documentation>
1282 </xsd:annotation>
  
```

1283 Each UN/CEFACT construct containing a code must include documentation that will
 1284 identify the code list(s) that must be supported when the construct is used.

1285 [Appendix F section F.1 Annotation Documentation](#) shows the XML Schema
 1286 definition of annotation documentation for each of the types of components from
 1287 CCTS.

1288 **7.5.2 Application Information (AppInfo)**

1289 The annotation `xsd:appInfo` will be used to convey the Usage Rules and the
 1290 Business Context that is applicable for each BIE and BDT artifact and the resulting
 1291 XML Schema artifacts used to express them.

1292 [Note:]

1293 The UN/CEFACT TMG UCM project is defining the context mechanism that will
 1294 support refining context categories in a given business circumstance. Once that
 1295 specification is finalized, this section may change.

1296 Example 7-9 shows the XML Schema definition of the annotation application
 1297 Information structure `ccts:UsageRule`.

1298 **Example 7-9: XML Schema definition for annotation appInfo for ccts:UsageRule**

1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312

```

<xsd:schema
  ...
  <xsd:element name="UsageRule" type="ccts:UsageRuleType"/>
  <xsd:complexType name="UsageRuleType">
    <xsd:sequence>
      <xsd:element name="UniqueID" type="bdt:EntityUniqueIdentifierType"/>
      <xsd:element name="Constraint" type="bdt:TextType"/>
      <xsd:element name="ConstraintTypeCode" type="bdt:CodeType"/>
      <xsd:element name="ConditionTypeCode" type="bdt:ConditionTypeCodeType"/>
    </xsd:sequence>
  </xsd:complexType>
  ...
</xsd:schema>

```

1313 [Appendix F Section F.2 Annotation Application Information](#) shows the XML Schema
1314 definition of the annotation application Information structure for
1315 **ccts:BusinessContext**.

1316 Both **ccts:UsageRule** and **ccts:BusinessContext** are applied to each of the
1317 XML Schema Components **xsd:element**, **xsd:complexType** and
1318 **xsd:simpleType** in order to communicate the usage and context in which the
1319 corresponding CCTS artifacts are applicable.

[R 9B07]	Each of the resulting XML Schema Components (xsd:element , xsd:complexType and xsd:simpleType) MUST have an xsd:annotation xsd:appInfo declared that includes zero or more ccts:UsageRule elements and one or more ccts:BusinessContext elements.	1
----------	--	---

1320 7.5.2.1 Usage Rules

1321 CCTS defines the concept of usage rules to convey instructions on how to use a
1322 CCTS artifact in a given context. Usage rules have a **ccts:ConstraintType**
1323 which classifies the rules as being structured (expressed in a formal language such
1324 as the Object Management Group's Object Constraint Language (OCL)) or
1325 unstructured (free form text).

1326 Usage Rules are communicated through zero or more **ccts:UsageRule** XML
1327 Schema Elements within an **xsd:appInfo**. Usage rules may be either structured or
1328 unstructured. Unstructured usage rule constraint values are expressed as free form
1329 text. Structured usage rule constraint values are expressed in a formal constraint
1330 language such as the Object Management Group (OMG) Object Constraint
1331 Language (OCL).and are suitable for direct application processing.

[R 88DE]	Usage rules MUST be expressed within an xsd:appInfo ccts:UsageRule element.	1
[R B851]	The structure of the ccts:UsageRule element MUST be: <ul style="list-style-type: none"> • ccts:UniqueID [1..1] – A unique identifier for the UsageRule. • ccts:Constraint [1..1] – The actual constraint expression. • ccts:ConstraintType [1..1] – The type of constraint E.g. 	1

	unstructured, OCL. <ul style="list-style-type: none"> • ccts:ConditionType [1..1] – The type of condition. Allowed values are pre-condition, post-condition, and invariant. 	
--	--	--

1332 The ccts:ConstraintType value will be taken from a constraint value code list
1333 schema.

[R A1CF]	A ccts:ConstraintType code list XML Schema File will be created.	1
----------	--	---

1334 7.5.2.2 Business Context

1335 All elements specified within an **xsd:appInfo ccts:BusinessContext** element
1336 will be expressions of CCTS context categories.

1337 The following **xsd:appInfo** structures are required as defined in each of the sub-
1338 sections in the section [8 XML Schema Files](#) that correspond to the different CCTS
1339 artifacts. The BusinessContext defined within each **xsd:appInfo** contains one or
1340 more **ccts:ContextUnit** elements which in turn contains one or more values for
1341 each of the identified context categories recognized by CCTS.

- 1342 • Business Process Context Category
- 1343 • Business Process Role Context Category
- 1344 • Supporting Role Context Category
- 1345 • Industry Classification Context Category
- 1346 • Product Classification Context Category
- 1347 • Geopolitical Context Category
- 1348 • Official Constraints Context Category
- 1349 • System Capabilities Context Category

[R A538]	Each defined or declared XML Schema artifact MUST use an xsd:annotation and xsd:appInfo element to communicate the context of the artifact.	1
----------	---	---

1350 Using this structure it is possible to indicate all of the context categories in which a
1351 BIE is applicable, and all of the applicable context values within a context category
1352 as shown in Example 7-10.

1353 **Example 7-10: Use of the **xsd:appInfo** and **ccts:BusinessContext****

1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364

```

<xsd:element name="<name>" type="<type>">
  <xsd:annotation>
    ... (documentation) ...
  <xsd:appinfo source="urn:un:unece:uncefact:businesscontext...">
    <ccts:UsageRules>
      ...
    </ccts:UsageRules>
    <ccts:BusinessContext>
      <ccts:ContextUnit>
        <ccts:BusinessProcessContextCategory>
          <ccts:BusinessTransactionDocumentCode>0062
  
```


1400 **8 XML Schema Files**

1401 This section describes how the requirements of the various XML Schema files that
 1402 are incorporated within the UN/CEFACT library are built through the application of
 1403 context categories, unique namespaces and the rules of this specification.

- 1404 • XML Schema Files, Context and Namespaces
- 1405 • Root XML Schema Files
- 1406 • Business Information Entities XML Schema Files
- 1407 • Business Data Type XML Schema Files
- 1408 • Code List XML Schema Files
 - 1409 ○ General Code List XML Schema Components
 - 1410 ○ Common Code List XML Schema Components
 - 1411 ○ Business Code List XML Schema Components

1412 **8.1 XML Schema Files, Context and Namespaces**

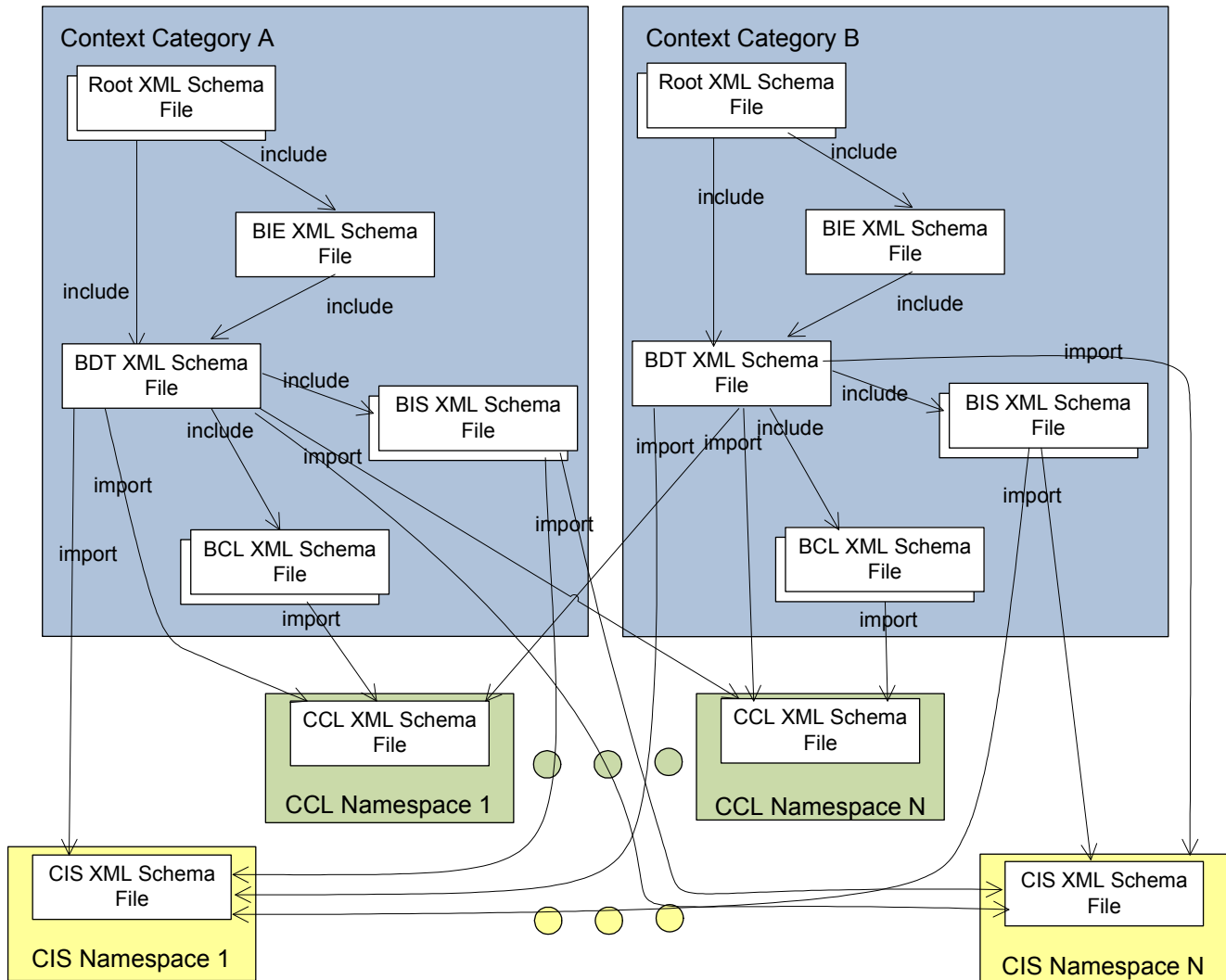
1413 As indicated in section [5.7 XML Schema Files](#) the XML Schema files have
 1414 dependencies upon one another.

1415 Figure 8-1 further shows these dependencies and shows how these dependencies
 1416 are realized using the `xsd:include` and `xsd:import` XML Schema features.
 1417 Since the primary context category values are implemented within the namespace
 1418 scheme, all of the XML Schema Files for the given context category value are
 1419 defined within the corresponding namespace. The XML Schema Files for other
 1420 values of the context categories are defined in namespaces corresponding to those
 1421 values.

1422 Figure 8-1 shows two context category values “A” and “B.” The namespaces used to
 1423 express the two context category values are independently declared and may not
 1424 have any shared dependencies other than Common Code Lists that are independent
 1425 of all context.

1426 All XML Schema Files published by UN/CEFACT will be assigned to a unique
 1427 namespace and a unique token that represents the business process context
 1428 category value in which it is designed.

[R B96F]	Each Root, BIE, BDT and BCL XML Schema File MUST be assigned to a unique namespace that represents the primary context category value of its contents.	1
----------	--	---



1429 **Figure 8-1: Imports and Includes of XML Schema Files for UN/CEFACT**
 1430 **Mouularity Model**

1431 Example 8-1 shows a namespace declaration for the context category Business
 1432 Process Value where the value is Order Management.

1433 **Example 8-1: Namespace for Context Category Business Process – Order Management**

1434

```
"xmlns:ordman="urn:un:unece:uncefact:ordermanagement:data:draft:1"
```

1435 Example 8-2 shows how an XML Schema File that is declared within the context
 1436 category Business Process Value of Order Management.

1437 **Example 8-2: Schema-element target namespace declaration for context category Business**
 1438 **Process Value – Order Management**

1439

```
<xsd:schema  

  1440 targetNamespace=  

  1441 "urn:un:unece:uncefact:ordermangement:data:1:draft"  

  1442 xmlns:ordman=  

  1443 "urn:un:unece:uncefact:ordermanagement:data:1:draft"
```

1444 [Note:]
 1445 Implementations of this specification require the use of a semantically meaningful
 1446 namespace prefix like “**ordman**” for the Business Process – Order Management.

1447 8.2 Root XML Schema Files

1448 The Root XML Schema File serves as the container for all schema defined content
 1449 required to fulfill a business information exchange for the given payload in the
 1450 context category namespace. All of the Root XML Schema Files that are necessary
 1451 to fulfill the context category are defined within the namespace of the context
 1452 category value.

1453 Figure 8-1 shows multiple Root XML Schema Files defined in two context category
 1454 based namespaces. Each primary context category value namespace will have 1 to
 1455 many Root XML Schema Files.

1456 8.2.1 XML Schema Structure

1457 Each Root XML Schema File will be structured in a standardized format as specified
 1458 in Appendix B in order to ensure consistency and ease of use. The specific format is
 1459 shown in Example 8-3. The Root XML Schema File must adhere to the format of the
 1460 relevant sections as detailed in Appendix B.

1461 Example 8-3: Root XML Schema File Structure

```

1462 <?xml version="1.0" encoding="UTF-8"?>
1463 <!-- ===== -->
1464 <!-- ===== [MODULENAME] XML Schema File ===== -->
1465 <!-- ===== -->
1466 <!--
1467 Schema agency:      UN/CEFACT
1468 Schema version:    3.0
1469 Schema date:       18 November 2008
1470
1471 Copyright (C) UN/CEFACT (2008). All Rights Reserved.
1472
1473 ... see copyright information ...
1474 -->
1475 <xsd:schema
1476   targetNamespace="urn:un:unece:unefact:data:ordermanagement:3:draft"
1477   ... see namespaces ...
1478   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1479   elementFormDefault="qualified" attributeFormDefault="unqualified" version="3.0">
1480 <!-- ===== -->
1481 <!-- ===== Include ===== -->
1482 <!-- ===== -->
1483 <!-- ===== Include of [MODULENAME] ===== -->
1484 <!-- ===== -->
1485 ... see includes ...
1486 <!-- ===== -->
1487 <!-- ===== Element Declarations ===== -->
1488 <!-- ===== -->
1489 <!-- ===== Root Element Declarations ===== -->
1490 <!-- ===== -->
1491   See element declarations...
1492 <!-- ===== -->
1493 <!-- ===== Type Definitions ===== -->
1494 <!-- ===== -->
1495 <!-- ===== Type Definitions: [TYPE] ===== -->
1496 <!-- ===== -->
1497 <xsd:complexType name="[TYPENAME]">
1498   <xsd:restriction base="xsd:token">

```

1499
1500
1501
1502

```

        ... see type definition ...
    </xsd:restriction>
</xsd:complexType>
</xsd:schema>

```

1503 **8.2.2 Includes**

1504 Every Root XML Schema File in a namespace will include the BIE XML Schema File,
1505 and the BDT XML Schema File that reside in that namespace for the specified
1506 context category value.

[R B698]	The Root XML Schema File MUST include the BIE and BDT XML Schema Files that reside in its namespace.	1
----------	--	---

1507 **8.2.3 Root Element Declaration**

1508 Each business information payload message has a single root element that is
1509 globally declared in the Root XML Schema File. The global element is named
1510 according to the business information payload that it represents and references the
1511 target information payload that contains the actual business information.⁴

[R BD9F]	A global element known as the root element, representing the business information payload, MUST be declared in the Root XML Schema File using the XML Schema Component xsd:element .	1
[R A466]	The name of the root element MUST be the same as the name of the business information payload data dictionary name, with separators and spaces removed.	1
[R 8062]	The root element declaration MUST be defined using an xsd:complexType that represents the message content contained within the business information payload.	1

1512 Example 8-4 shows an example of Root Element declaration with in a Root XML
1513 Schema File.

1514 **Example 8-4: Root Element declaration**1515
1516
1517
1518
1519
1520
1521
1522

```

<!-- ===== Root Element ===== -->
<!-- ===== Root Element ===== -->
<!-- ===== Root Element ===== -->
    <xsd:element name="Invoice" type="rsm:InvoiceType">
        <xsd:annotation>
            ... see annotation ...
        </xsd:annotation>
    </xsd:element>

```

⁴ All references to root element represent the globally declared element in a UN/CEFACT schema module that is designated as the root element for instances that use that schema.

1523 **8.2.4 Type Definitions**

1524 Root XML Schema Files are limited to defining a single MA **xsd:complexType**
 1525 whose content model contains ASMA that represent the first level BIEs for a
 1526 business information payload.

[R 8837]	Each Root XML Schema File MUST define a single xsd:complexType that fully describes the business information payload.	1
[R 9119]	The name of the root schema xsd:complexType MUST be the name of the root element with the word 'Type' appended.	1

1527 Example 8-5 shows the definition of a Root XML Schema Files complex type
 1528 definition.

1529 **Example 8-5: Root element complex type name**

1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548

```

<!-- =====>
<!-- ===== Root Element =====>
<!-- =====>
  <xsd:element name="Invoice" type="rsm:InvoiceType">
    <xsd:annotation>
      ... see annotation ...
    </xsd:annotation>
  </xsd:element>
<!-- =====>
<!-- ===== ComplexType =====>
<!-- =====>
  <xsd:complexType name="InvoiceType">
    <xsd:annotation>
      ... see annotation ...
    </xsd:annotation>
    <xsd:sequence>
      ...
    </xsd:sequence>
  </xsd:complexType>

```

1549 **8.2.5 Annotations**1550 **8.2.5.1 Annotation Documentation**

1551 In the Root XML Schema File the root element declaration must have a structured
 1552 set of annotation documentation.

[R 8010]	<p>The Root XML Schema File root element declaration MUST have a structured set of annotations documentation (xsd:annotation xsd:documentation) present in that includes:</p> <ul style="list-style-type: none"> • UniqueID (mandatory): The identifier that uniquely identifies the business information payload, the root element. • VersionID (mandatory): The unique identifier that identifies the version of the business information payload, the root element. • ObjectClassQualifierName (zero or more): Is a word or words which help define and differentiate an ABIE from its 	1
----------	--	---

	<p>associated CC and other BIEs. It enhances the semantic meaning of the DEN to reflect a restriction of the concept, conceptual domain, content model or data value.</p> <ul style="list-style-type: none"> • ObjectClassName (mandatory): Is a semantically meaningful name of the Object class. It is the basis for the DEN. • DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the business information payload. • Definition (mandatory): The semantic meaning of the root element. • BusinessTermName (optional, repeating): A synonym term under which the payload object is known by in industry. 	
--	---	--

1553 Example 8-6 shows the definition of the annotation documentation for the Root
1554 Element.

1555 **Example 8-6: Root element annotation documentation**

1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569

```

<xsd:group name="RootSchemaDocumentation">
  <xsd:sequence>
    <xsd:element name="UniqueID"
type="bdt:EntityUniqueIdentifierType"/>
    <xsd:element name="VersionID" type="bdt:VersionIdentifierType"/>
    <xsd:element name="ObjectClassQualifierName" type="bdt:NameType"
minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="ObjectClassTermName" type="bdt:NameType"/>
    <xsd:element name="DictionaryEntryName" type="bdt:NameType"/>
    <xsd:element name="Definition" type="bdt:TextType"/>
    <xsd:element name="BusinessTermName" type="bdt:NameType"
minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:group>

```

1570 8.2.5.2 Annotation Application Information (AppInfo)

1571 The annotation `xsd:appInfo` on the Root Element is used to convey the context
1572 that is applicable for the Root Element. The structure of the context is provided in
1573 section [7.5.2, Application Information \(AppInfo\)](#). The specific context values for the
1574 Root Element represent the context values for the Root XML Schema File.

1575 8.3 Business Information Entity XML Schema Files

1576 A UN/CEFACT BIE XML Schema File contains all of the ABIEs used for the context
1577 category value that is reflected in the namespace. This BIE XML Schema File will be
1578 used (included into) in all of the UN/CEFACT Root XML Schema Files within the
1579 namespace.

1580 8.3.1 Schema Structure

1581 Each BIE XML Schema File will be structured in the standardized format detailed in
1582 [Appendix B](#). The specific format is shown in Example 8-7 and must adhere to the
1583 format of the relevant sections in [Appendix B](#).

1584 **Example 8-7: Structure of BIE XML Schema Files**

```

1584 <?xml version="1.0" encoding="UTF-8"?>
1585 <!-- ===== -->
1586 <!-- ===== ABIEs XML Schema File ===== -->
1587 <!-- ===== -->
1588 <!--
1589 Schema agency:          UN/CEFACT
1590 Schema version:        3.0
1591 Schema date:           18 November 2008
1592
1593 Copyright (C) UN/CEFACT (2008). All Rights Reserved.
1594
1595     ... see copyright information ...
1596 -->
1597 <xsd:schema
1598   targetNamespace=
1599   ... see namespace declaration ...
1600   xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
1601   attributeFormDefault="unqualified">
1602   <!-- ===== -->
1603   <!-- ===== Includes ===== -->
1604   <!-- ===== -->
1605   ... see includes ...
1606   <!-- ===== -->
1607   <!-- ===== Type Definitions ===== -->
1608   <!-- ===== -->
1609   ... see type defintions ...
1610 </xsd:schema>
1611

```

1612 **8.3.2 Includes**

1613 The BIE XML Schema File will include the corresponding BDT XML Schema File that
 1614 resides in the same namespace.

[R 8FE2]	The BIE XML Schema File MUST contain an <code>xsd:include</code> statement for the BDT XML Schema File that resides in the same namespace.	1
----------	---	---

1615 Example 8-8 shows the syntax for including the BDT XML Schema File.

1616 **Example 8-8: Include of BDT XML Schema File**

```

1617 <!-- ===== -->
1618 <!-- ===== Includes ===== -->
1619 <!-- ===== -->
1620 <!-- ===== Include of Business Data Type XML Schema File ===== -->
1621 <!-- ===== -->
1622 <xsd:include schemaLocation="BusinessDataType_1p0.xsd"/>

```

1623 **8.3.3 Type Definitions**1624 **8.3.3.1 ABIE Type Definitions**

1625 Every ABIE with the same primary context category is defined as an
 1626 `xsd:complexType` in the BIE XML Schema File for that primary context category
 1627 namespace.

[R AF95]	For every object class (ABIE) identified in a primary context category, a named <code>xsd:complexType</code> MUST be defined in its corresponding BIE XML Schema File.	1
----------	---	---

1628 The name of the `xsd:complexType` will represent the DEN of the BIE.

[R 9D83]	The name of the ABIE <code>xsd:complexType</code> MUST be the <code>ccts:DictionaryEntryName</code> with the spaces and separators removed, with approved abbreviations and acronyms applied and with the 'Details' suffix replaced with 'Type'.	1
----------	--	---

1629 The content model of the `xsd:complexType` will be defined such that it reflects
 1630 each property of the object class. The content model of the ABIE complex type
 1631 definitions will include element declarations for BBIEs, element declarations for
 1632 ASBIEs whose `associationKind=composite`, or element references for ASBIEs
 1633 whose `associationKind=shared`.

1634 The cardinality and sequencing of each ABIE Property will be determined by the
 1635 **Cardinality** and **Sequencing Key** values of the source ABIE.

[R 90F9]	The cardinality and sequencing of the elements within an ABIE <code>xsd:complexType</code> MUST be as defined by the corresponding ABIE values in the syntax neutral model.	1
----------	---	---

1636 In defining the content model, both `xsd:sequence` and `xsd:choice` compositors are
 1637 allowed.

[R 9C70]	Every aggregate business information entity (ABIE) <code>xsd:complexType</code> definition content model MUST use zero or more <code>xsd:sequence</code> and/or zero or more <code>xsd:choice</code> elements to reflect each property (BBIE or ASBIE) of its class.	1
----------	--	---

1638 When using the `xsd:sequence` and `xsd:choice` content models in a type
 1639 definition their order must be carefully managed. An `xsd:sequence` should not
 1640 contain another `xsd:sequence` directly as there is no additional value. An
 1641 `xsd:choice` should not contain another `xsd:choice` directly as there is no
 1642 additional value. However, it is permissible to interweave `xsd:sequence` and
 1643 `xsd:choice` within a single `xsd:complexType` definition to whatever level of
 1644 nesting is desired.

[R 81F0]	Repeating series of only <code>xsd:sequence</code> MUST NOT occur.	1
[R 8FA2]	Repeating series of only <code>xsd:choice</code> MUST NOT occur.	1

1645 Example 8-9 show an example of using `xsd:sequence`.

1646 **Example 8-9: Sequence compositor within an ABIE type definition**

1647
1648
1649
1650
1651
1652
1653
1654
1655

```

<xsd:complexType name="AccountType" >
  <xsd:annotation>
    ...see annotation...
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="ID" type="bdt:IDType"
      minOccurs="0" maxOccurs="unbounded">
      <xsd:annotation>
        ...see annotation...
    
```

1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672

```

        </xsd:annotation>
      </xsd:element>
      <xsd:element name="Status" type="bie:StatusType"
        minOccurs="0" maxOccurs="unbounded">
        <xsd:annotation>
          ...see annotation...
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="Name" type="bdt:NameType"
        minOccurs="0" maxOccurs="unbounded">
        <xsd:annotation>
          ...see annotation...
        </xsd:annotation>
      </xsd:element>
      ...
    </xsd:sequence>
  </xsd:complexType>

```

1673 Example 8-10 show an example of using **xsd:choice**.

1674 **Example 8-10: Choice compositor within an ABIE type definition**

1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699

```

<xsd:complexType name="LocationType">
  <xsd:annotation>
    ... see annotation ...
  </xsd:annotation>
  <xsd:choice>
    <xsd:element name="GeoCoordinate" type="bie:GeoCoordinateType"
      minOccurs="0">
      <xsd:annotation>
        ... see annotation ...
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="Address" type="bie:AddressType"
      minOccurs="0">
      <xsd:annotation>
        ... see annotation ...
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="Location" type="bie:LocationType"
      minOccurs="0">
      <xsd:annotation>
        ... see annotation ...
      </xsd:annotation>
    </xsd:element>
  </xsd:choice>
</xsd:complexType>

```

1700 Example 8-11 shows an example of interweaving **xsd:sequence** and
1701 **xsd:choice**.

1702 **Example 8-11: Sequence + Choice compositors within an ABIE type definition**

1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718

```

<xsd:complexType name="PeriodType">
  ...
  <xsd:sequence>
    <xsd:element name="DurationDateTime"
      type="qdt:DurationDateTimeType" minOccurs="0"
      maxOccurs="unbounded">
      ...
    </xsd:element>
    ...
  </xsd:sequence>
  <xsd:choice>
    <xsd:sequence>
      <xsd:element name="StartTime" type="bdt:TimeType"
        minOccurs="0">
        ...
      </xsd:element>
      <xsd:element name="EndTime" type="bdt:TimeType"

```

```

1719         minOccurs="0">
1720         ...
1721     </xsd:element>
1722 </xsd:sequence>
1723 <xsd:sequence>
1724     <xsd:element name="StartDate" type="bdt:DateType"
1725         minOccurs="0">
1726         ...
1727     </xsd:element>
1728     <xsd:element name="EndDate" type="bdt:DateType"
1729         minOccurs="0">
1730         ...
1731     </xsd:element>
1732 </xsd:sequence>
1733 <xsd:sequence>
1734     <xsd:element name="StartDateTime"
1735 type="bdt:DateTimeType"
1736         minOccurs="0">
1737         ...
1738     </xsd:element>
1739     <xsd:element name="EndDateTime"
1740 type="bdt:DateTimeType"
1741         minOccurs="0">
1742         ...
1743     </xsd:element>
1744 </xsd:sequence>
1745 </xsd:choice>
1746 </xsd:sequence>
1747 </xsd:complexType>

```

1748 8.3.3.2 BBIE Type Definitions

1749 BBIEs are defined as local elements and are either of `xsd:simpleType` or
1750 `xsd:complexType`.

[R A21A]	Every BBIE within the containing ABIE MUST have a named xsd:simpleType (If the BBIE BDT includes only the content component) or xsd:complexType (If the BBIE BDT includes one or more supplementary components).	1
----------	--	---

1751 The name of the BBIE type will represent the DEN of the BBIE.

[R 8B85]	Every BBIE type MUST be named the property term and qualifiers and the representation term of the basic business information entity (BBIE) it represents with the word ' Type ' appended.	1
----------	--	---

1752 8.3.3.3 ASBIE Type Definitions

1753 ASBIEs are declared as either local or global elements whose `xsd:complexType` is
1754 that of the `xsd:complexType` of the associated ABIE it represents. No additional type
1755 definition is required.

1756 8.3.4 Element Declarations and References

1757 8.3.4.1 ABIE Element Declarations

1758 Every ABIE will have a globally declared element. This global element reflects the
1759 unique DEN of the ABIE within the namespace to which it is assigned and will be of
1760 the **xsd:complexType** that represents it.

[R 9DA0]	For each ABIE, a named xsd:element MUST be globally declared.	1
[R 9A25]	The name of the ABIE xsd:element MUST be the ccts:DictionaryEntryName with the separators and 'Details' suffix removed and approved abbreviations and acronyms applied.	1
[R B27B]	Every ABIE global element declaration MUST be of the xsd:complexType that represents the ABIE.	1

1761 8.3.4.2 BBIE Element Declarations

1762 Every BBIE will have a locally declared element that is part of the content model of
1763 the ABIE to which it belongs.

[R 89A6]	For every BBIE identified in an ABIE, a named xsd:element MUST be locally declared within the xsd:complexType representing that ABIE.	1
----------	---	---

1764 The name of the BBIE element will reflect the name of the BBIE devoid of the object
1765 class and object class qualifiers.

[R AEFE]	Each BBIE element name declaration MUST be the property term and qualifiers and the representation term of the BBIE.	1
----------	--	---

1766 Simplification of the BBIE Property name for the representation terms of
1767 **Identification**, **Indicator**, and **Text** are allowed to improve semantic
1768 expression.

[R 96D9]	For each BBIE element name declaration where the word 'Identification' is the final word of the property term and the representation term is 'Identifier', the term 'Identification' MUST be removed.	1
[R 9A40]	For each BBIE element name declaration where the word 'Indication' is the final word of the property term and the representation term is 'Indicator', the term 'Indication' MUST be removed from the property term.	1
[R A34A]	If the representation term of a BBIE is 'Text', 'Text' MUST be removed from the name of the element or type definition.	1

1769 The BBIE element will be of the **xsd:simpleType** or **xsd:complexType** as
1770 defined in Section 8.3.3.2.

[R BCD6]	Every BBIE element declaration MUST be of the BusinessDataType that represents the source basic business	1
----------	---	---

	information entity (BBIE) data type.	
--	--------------------------------------	--

1771 Example 8-12 shows an Account ABIE complexType declaration that contains BBIE
1772 element declarations that make use of the appropriate BDTs.

1773 Example 8-12: BBIE Element Declaration

```

1774 <xsd:complexType name="AccountType">
1775   <xsd:annotation>
1776     ...see annotation...
1777   </xsd:annotation>
1778   <xsd:sequence>
1779     <xsd:element name="ID" type="bdt:IDType"
1780       minOccurs="0" maxOccurs="unbounded">
1781       <xsd:annotation>
1782         ...see annotation...
1783       </xsd:annotation>
1784     </xsd:element>
1785     <xsd:element name="Status" type="bie:StatusType"
1786       minOccurs="0" maxOccurs="unbounded">
1787       <xsd:annotation>
1788         ...see annotation...
1789       </xsd:annotation>
1790     </xsd:element>
1791     <xsd:element name="Name" type="bdt:NameType"
1792       minOccurs="0" maxOccurs="unbounded">
1793       <xsd:annotation>
1794         ...see annotation...
1795       </xsd:annotation>
1796     </xsd:element>
1797     <xsd:element name="BuyerParty" type="bie:BuyerPartyType"/>
1798   </xsd:sequence>
1799 </xsd:complexType>

```

1800 8.3.4.3 ASBIE Element Declarations

1801 For ASBIEs whose `ccts:AggregationKind` value is **composite**, a local element
1802 for the associated ABIE will be declared in the content model of the associating ABIE
1803 **xsd:complexType**.

[R 9025]	For every ASBIE whose <code>ccts:AggregationKind</code> value = composite , a local element for the associated ABIE MUST be declared in the associating ABIE xsd:complexType content model.	1
----------	---	---

1804 For each ASBIE whose `ccts:AggregationKind` value is **shared**, a global
1805 element is declared. See section [5.5 Reusability Schema](#) earlier this specification.

[R 9241]	For every ASBIE whose <code>ccts:AggregationKind</code> value = shared , a global element MUST be declared.	1
----------	--	---

1806 The name of the ASBIE local or global element will reflect the name of the ASBIE
1807 devoid of the associating object class and qualifiers.

[R A08A]	Each ASBIE element name MUST be the ASBIE property term and qualifier term(s) and the object class term and qualifier term(s) of the associated ABIE.	1
----------	---	---

1808 The ASBIE local or global element will be of the **xsd:complexType** of the
1809 associated ABIE.

[R B27C]	Each ASBIE element declaration MUST use the xsd:complexType that represents its associated ABIE.	1
----------	---	---

1810

1811 Example 8-13 shows an ABIE type definition with a local element declaration for a
1812 BBIE (“ID”), a local element declaration for two ASBIEs (“SellerParty” and
1813 “BuyerParty”) and a global element reference for the Invoice specific ABIE
1814 (“InvoiceTradeLineItem”).

1815 **Example 8-13: ASBIE element declaration and reference within an ABIE type definition**

1816
1817
1818
1819
1820
1821
1822
1823
1824

```
<xsd:element name="InvoiceTradeLineItem" type="InvoiceTradeLineItemType"/>
<xsd:complexType name="InvoiceType">
  <xsd:sequence>
    <xsd:element name="ID" type="bdt:IDType"/>
    <xsd:element name="SellerParty" type="ordman:SellerPartyType"/>
    <xsd:element name="BuyerParty" type="ordman:BuyerPartyType"/>
    <xsd:element ref="ordman:InvoiceTradeLineItem"
maxOccurs="unbounded"/>
  </xsd:sequence>
```

1825 8.3.5 Annotation

1826 8.3.5.1 ABIE Complex Type Definition

1827 Every ABIE complexType definition must include structured annotation
1828 documentation.

[R ACB9]	<p>For every ABIE xsd:complexType definition a structured set of annotations MUST be present in the following pattern:</p> <ul style="list-style-type: none"> • UniqueID (mandatory): The unique identifier that identifies an ABIE instance in a unique and unambiguous way. • VersionID (mandatory): An unique identifier that identifies the version of an ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE. • DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the ABIE. • Definition (mandatory): The semantic meaning of the ABIE. • BusinessTermName (optional, repeating): A synonym term in which the ABIE is commonly known. 	1
----------	--	---

1829 In addition, every ABIE **xsd:complexType** definition will have structured annotation
1830 application information that reflects its context and any usage rules.

[R B0BA]	For every ABIE xsd:complexType definition a structured set of xsd:annotation xsd:appInfo elements MUST be present that fully declare its context.	1
[R BCE9]	For every ABIE usage rule, the ABIE xsd:complexType definition MUST contain a structured set of xsd:annotation xsd:appInfo elements in the following pattern: <ul style="list-style-type: none"> • ccts:UniqueID • ccts:Constraint • ccts:ConstraintType • ccts:ConditionType. 	1

1831 Example 8-14 shows the annotation documentation of an ABIE complexType
1832 definition.

1833 **Example 8-14: ABIE complex type definition annotation**

1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849

```

<xsd:complexType name="AccountType" >
  <xsd:annotation>
    <xsd:documentation xml:lang="en-US">
      <ccts:UniqueID>UNBE000000</ccts:UniqueID>
      <ccts:VersionID>0.00</ccts:VersionID>
      <ccts:ObjectClassQualifierName></ccts:ObjectClassQualifierName>
      <ccts:ObjectClassTermName>Account</ccts:ObjectClassTermName>
      <ccts:DictionaryEntryName>Account</ccts:DictionaryEntryName>
      <ccts:Definition>Communicates the Account information.</ccts:Definition>
      <ccts:BusinessTermName></ccts:BusinessTermName>
    </xsd:documentation>
    <xsd:appInfo>
      As shown in Appendix F
    </xsd:appInfo>
  </xsd:annotation>
</xsd:complexType>

```

1850 [8.3.5.1.1 ABIE Element](#)

1851 Every ABIE element declaration must include structured annotation documentation.

[R 88B6]	For every ABIE xsd:element declaration definition, a structured set of annotations MUST be present in the following pattern: <ul style="list-style-type: none"> • UniqueID (mandatory): The unique identifier that identifies an ABIE instance in a unique and unambiguous way. • VersionID (mandatory): An unique identifier that identifies the version of an ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE. • DictionaryEntryName (mandatory): The Data Dictionary 	1
----------	--	---

	<p>Entry Name (DEN) of the ABIE.</p> <ul style="list-style-type: none"> • Definition (mandatory): The semantic meaning of the ABIE. • BusinessTermName (optional, repeating): A synonym term in which the ABIE is commonly known. 	
--	---	--

1852 The global element declaration for ABIEs is used exclusively for referencing by
 1853 ASMAAs. Since multiple ASMAAs can reference a single global ABIE element
 1854 declaration in different contexts with different usage rules, the context and usage
 1855 rules for global ABIE element declarations can not be explicitly stated in the BIE XML
 1856 Schema File. However, the context and usage rules can be stated when the global
 1857 ABIE element is referenced using `xsd:ref` as part of the content model of the MA.

1858 [8.3.5.1.2 BBIE Element](#)

1859 Every BBIE element declaration will include structured annotation documentation.

[R B8BE]	<p>For every BBIE <code>xsd:element</code> declaration a structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements MUST be present in the following pattern:</p> <ul style="list-style-type: none"> • Cardinality (mandatory): Indicates the cardinality of the BBIE within the containing ABIE. • SequencingKey (mandatory): Indicates the sequence of the BBIE within the containing ABIE. • DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the BBIE. • Definition (mandatory): The semantic meaning of the associated BBIE. • BusinessTermName (optional, repeating): A synonym term in which the BBIE is commonly known. • PropertyTermName (mandatory): Represents a distinguishing characteristic of the BBIE. • PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the BBIE. • RepresentationTermName (mandatory): An element of the component name that describes the form in which the BBIE is represented. 	1
----------	--	---

1860 In addition, every BBIE will have structured annotation application information that
 1861 reflects its context and any defined usage rules.

[R 95EB]	For every BBIE <code>xsd:element</code> declaration a structured set of <code>xsd:annotation</code> <code>xsd:appInfo</code> elements MUST be present that fully declare its context.	1
[R 8BF6]	For every BBIE usage rule, the BBIE <code>xsd:element</code> declaration	1

	<p>MUST contain a structured set of xsd:annotation xsd:appInfo elements in the following pattern:</p> <ul style="list-style-type: none"> • ccts:UniqueID • ccts:Constraint • ccts:ConstraintType • ccts:ConditionType. 	
--	--	--

1862 Example 8-15 shows the annotation documentation of a BBIE Element.

1863 **Example 8-15: BBIE element annotation**

```

1864 <xsd:element name="ID" type="bdt:IDType" minOccurs="0" maxOccurs="unbounded">
1865 <xsd:annotation>
1866 <xsd:documentation xml:lang="en-US">
1867 <ccts:UniqueID>UNBE000000</ccts:UniqueID>
1868 <ccts:VersionID>0.00</ccts:VersionID>
1869 <ccts:Cardinality>1</ccts:Cardinality>
1870 <ccts:SequencingKey>1</ccts:SequencingKey>
1871 <ccts:DictionaryEntryName>Account. Identificaton.
1872 Identifier</ccts:DictionaryEntryName>
1873 <ccts:Definition>The Account Identification Identifier.</ccts:Definition>
1874 <ccts:BusinessTermName></ccts:BusinessTermName>
1875 <ccts:PropertyTermName></ccts:PropertyTermName>
1876 <ccts:PropertyQualifierName></ccts:PropertyQualifierName>
1877 <ccts:RepresentationTermName></ccts:RepresentationTermName>
1878 </xsd:documentation>
1879 <xsd:appInfo>
1880 As shown in Appendix F for context and usage rules
1881 </xsd:appInfo>
1882 </xsd:annotation>
1883 </xsd:element>

```

1884 **8.3.5.1.3 ASBIE Element**

1885 The global element declaration for ASBIEs is used exclusively for referencing by
1886 ABIEs. Since multiple ABIEs can reference a single global ASBIE element
1887 declaration in different contexts with different usage rules, most of the metadata,
1888 context and usage rules for global ASBIE element declarations can not be explicitly
1889 stated in the element declaration and the `xsd:annotation` `xsd:documentation`
1890 elements will be limited.

[R 8D3E]	<p>Every ASBIE global element declaration MUST have a structured set of xsd:annotation xsd:documentation elements in the following pattern:</p> <ul style="list-style-type: none"> • AssociationKind (mandatory): Indicates the UML AssociationKind value of shared or composite of the associated ABIE. • PropertyTermName (mandatory): Represents a distinguishing characteristic of the ASBIE. • PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the ASBIE. • AssociatedObjectClassName (Mandatory): The name of the 	1
----------	--	---

	<p>associated object class.</p> <ul style="list-style-type: none"> AssociatedObjectClassQualifierName (optional, repeating): a name or names that qualify the associated object class. 	
--	---	--

1891 Context and usage rules can be stated when the global ASBIE element is
 1892 referenced using `xsd:ref` as part of the content model of the ABIE. ASBIEs declared
 1893 locally, and every `xsd:ref` occurrence of a ASBIE declared globally, will include
 1894 structured annotation documentation.

1895 Every ASBIE local element declaration or `xsd:ref` occurrence in the content model
 1896 of an ABIE will include structured annotation documentation.

[R 926A]	<p>Every ASBIE <code>xsd:element</code> declaration or <code>xsd:ref</code> occurrence MUST have a structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements present in the following pattern:</p> <ul style="list-style-type: none"> Cardinality (mandatory): Indicates the cardinality of the ASBIE within the containing ABIE. SequencingKey (mandatory): Indicates the sequence of the ASBIE within the containing ABIE. DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the ASBIE. Definition (mandatory): The semantic meaning of the ASBIE. BusinessTermName (optional, repeating): A synonym term in which the ASBIE is commonly known. AssociationKind (mandatory): Indicates the UML AssociationKind value of <code>shared</code> or <code>composite</code> of the associated ABIE. PropertyTermName (mandatory): Represents a distinguishing characteristic of the ASBIE. PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the ASBIE. AssociatedObjectClassName (Mandatory): The name of the associated object class. AssociatedObjectClassQualifierName (optional, repeating): a name or names that qualify the associated object class. 	1
----------	---	---

1897 In addition, every ASBIE `xsd:element` local declaration or `xsd:ref` occurrence in
 1898 the content model of an ABIE will have structured annotation application information
 1899 that reflects its context and any defined usage rules.

[R 9D87]	<p>Every ASBIE <code>xsd:element</code> declaration or ASBIE <code>xsd:ref</code> to an ABIE global element declaration MUST contain a structured set of <code>xsd:annotation</code> <code>xsd:appInfo</code> elements that fully declare its</p>	1
----------	---	---

	context.	
[R A76D]	<p>Every ASBIE usage rule <code>xsd:element</code> declaration or ASBIE <code>xsd:ref</code> to an ABIE global element declaration MUST contain a structured set of <code>xsd:annotation</code> <code>xsd:appInfo</code> elements in the following pattern:</p> <ul style="list-style-type: none"> • <code>ccts:UniqueID</code> • <code>ccts:Constraint</code> • <code>ccts:ConstraintType</code> • <code>ccts:ConditionType</code>. 	1

1900 Example 8-16 shows the annotation documentation of an ASBIE Element. In this
 1901 case the ASBIE is declared as a shared AggregationKind which results in a global
 1902 element.

1903 Example 8-16: ASBIE global element declaration annotation

```

1904 <xsd:element name="DelayedShipmentDeliveryStatus" type="bie:DeliveryStatusType"
1905 minOccurs="0" maxOccurs="unbounded">
1906   <xsd:annotation>
1907     <xsd:documentation xml:lang="en-US">
1908       <ccts:AssociationKind>composite</ccts:AssociationKind>
1909       <ccts:PropertyTermName>Shipment</ccts:PropertyTermName>
1910       <ccts:PropertyQualifierName>delayed</ccts:PropertyQualifierName>
1911       <ccts:AssociatedObjectClassName>Status</ccts:AssociatedObjectClassName>
1912       <ccts:AssociatedObjectClassQualifier>Delivery</ccts:AssociatedObject
1913       ClassQualifier>
1914     </xsd:documentation>
1915   </xsd:annotation>
1916 </xsd:element>
  
```

1917 Example 8-17 shows the annotation documentation of an ASBIE Element. In this
 1918 case the ASBIE is declared as a composite AggregationKind which results in a local
 1919 element.

1920 Example 8-17: ASBIE local element declaration annotation

```

1921 <xsd:element name="DelayedShipmentDeliveryStatus" type="bie:StatusType"
1922 minOccurs="0" maxOccurs="unbounded">
1923   <xsd:annotation>
1924     <xsd:documentation xml:lang="en-US">
1925       <ccts:Cardinality>1</ccts:Cardinality>
1926       <ccts:SequencingKey>1</ccts:SequencingKey>
1927       <ccts:DictionaryEntryName>Order. Delayed_ Shipment. Delivery_
1928       Status</ccts:DictionaryEntryName>
1929       <ccts:Definition>The delivery status of the delayed shipment for this
1930       order.</ccts:Definition>
1931       <ccts:BusinessTermName></ccts:BusinessTermName>
1932       <ccts:AssociationKind>composite</ccts:AssociationKind>
1933       <ccts:PropertyTermName>Shipment</ccts:PropertyTermName>
1934       <ccts:PropertyQualifierName>delayed</ccts:PropertyQualifierName>
1935       <ccts:AssociatedObjectClassName>Status</ccts:AssociatedObjectClassName>
1936       <ccts:AssociatedObjectClassQualifier>Delivery</ccts:AssociatedObjectClassQualifie
1937       r>
1938     </xsd:documentation>
1939     <xsd:appInfo>
1940       As shown in Appendix F for context and usage rules
1941     </xsd:appInfo>
1942   </xsd:annotation>
  
```

1943

```
</xsd:element>
```

1944

Example 8-18 shows the annotation documentation of a reference to an ASBIE

1945

Element.

1946

Example 8-18. ASBIE element REF annotation

1947

```

<xsd:element ref="DelayedShipmentDeliveryStatus" minOccurs="0"
maxOccurs="unbounded">
  <xsd:annotation>
    <xsd:documentation xml:lang="en-US">
      <ccts:Cardinality>1</ccts:Cardinality>
      <ccts:SequencingKey>1</ccts:SequencingKey>
      <ccts:DictionaryEntryName>Order. Delayed_ Shipment. Delivery_
Status</ccts:DictionaryEntryName>
      <ccts:Definition>The delivery status of the delayed shipment for this
order.</ccts:Definition>
      <ccts:BusinessTermName></ccts:BusinessTermName>
      <ccts:AssociationKind>shared</ccts:AssociationKind>
      <ccts:PropertyTermName>Shipment</ccts:PropertyTermName>
      <ccts:PropertyQualifierName>delayed</ccts:PropertyQualifierName>
      <ccts:AssociatedObjectClassName>Status</ccts:AssociatedObjectClassName>
      <ccts:AssociatedObjectClassQualifier>Delivery</ccts:AssociatedObjectQualifie
r>
    </xsd:documentation>
    <xsd:appInfo>
      As shown in Appendix F for context and usage rules
    </xsd:appInfo>
  </xsd:annotation>
</xsd:element>

```

1970

8.4 Business Data Type XML Schema Files

1971

Multiple BDT XML Schema Files are created. One reference BDT XML Schema File will be created that contains all approved BDTs as published in the BDT catalogue.

1972

1973

Additional BDT XML Schema Files will be created that define all BDTs used in a primary context category namespace. The BDT XML Schema File names must

1974

1975

follow the UN/CEFACT XML Schema File naming approach. .

1976

8.4.1 Use of Business Data Type XML Schema Files

1977

The reference BDT XML Schema File will not be included as part of the modularity

1978

model, rather it is used as a reference. The primary context category BDT XML

1979

Schema Files will be used by the BIE XML Schema File and all Root Element XML

1980

Schema Files defined in the same primary context category namespace.

1981

8.4.2 XML Schema Structure

1982

Each BDT XML Schema File will be structured in a standard format to ensure

1983

consistency and ease of use.

1984

The format is shown in Example 8-19. Each BDT XML Schema File must adhere to the format of the relevant sections as detailed in [Appendix B](#).

1985

1986

Example 8-19: BDT XML Schema file structure

1987

```

<?xml version="1.0" encoding="utf-8"?>
<!-- ===== -->
<!-- ===== Business Data Type XML Schema File ===== -->
<!-- ===== -->

```

1988

1989

1990

```

1999 <!--
1999 Schema agency:          UN/CEFACT
1999 Schema version:       3.0
1999 Schema date:         18 November 2008
1999
1999 Copyright (C) UN/CEFACT (2008). All Rights Reserved.
1999
1999 ... see copyright information ...
1999
2000 -->
2000 <xsd:schema targetNamespace=
2000   ... see namespace ...
2000   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
2000   elementFormDefault="qualified" attributeFormDefault="unqualified">
2000   <!-- ===== -->
2000   <!-- ===== Includes ===== -->
2000   <!-- ===== -->
2000   ... see includes ...
2000   <!-- ===== -->
2000   <!-- ===== Imports ===== -->
2000   <!-- ===== -->
2000   ... see imports ...
2000   <!-- ===== -->
2000   <!-- ===== Type Definitions ===== -->
2000   <!-- ===== -->
2000   ... see type definitions ...
2000 </xsd:schema>

```

2020 8.4.3 Imports and Includes

2021 Each BDT XML Schema File will use `xsd:include` to make use of any BCL XML
 2022 Schema Files and BIS XML Schema Files being used by the BDT XML Schema
 2023 Components. Each BDT XML Schema File will also use `xsd:import` to make use
 2024 of any CCL XML Schema Files and CIS XML Schema Files being used by a BDT
 2025 within the BDT XML Schema File.

[R 8E0D]	The BDT XML Schema File MUST include (<code>xsd:include</code>) the BCL XML Schema Files and BIS XML Schema Files that are defined in the same namespace.	1
[R B4C0]	The BDT XML Schema File MUST import (<code>xsd:import</code>) the CCL XML Schema Files and CIS XML Schema Files that are used by a BDT contained within the file.	1

2026 8.4.4 Type Definitions

2027 The BDT XML Schema Components are defined as either an `xsd:complexType` or
 2028 `xsd:simpleType`.

[R AE00]	Each CCTS BDT artifact within the UN/CEFACT Data Type Catalogue used by the Root XML Schema Files and the BIE XML Schema File within a given namespace MUST be defined as an <code>xsd:simpleType</code> or <code>xsd:complexType</code> in the BDT XML Schema File with the given namespace.	1
----------	---	---

2029 As defined in the Data Type Catalogue a BDT content component Business Value
 2030 Domain (BVD) can contain either a set of primitives or a code list or point to an

2031 identifier scheme. This means that a data type can be defined to have one of several
 2032 primitives or one or more code lists or one or more identifier schemes. When the
 2033 BDT is defined in the BDT XML Schema File it will be defined to reflect a single
 2034 primitive, single code list, the list of code list combinations, or a single identifier
 2035 scheme.

2036 8.4.4.1 Business Value Domain Expressed By Primitives

2037 When a BDT content component Business Value Domain (BVD) is defined by a
 2038 primitive, and the primitive facets are supported by the facets of an XSD built-in data
 2039 type, the BDT will be defined as an `xsd:simpleType`.

[R 9908]	For every BDT whose content component BVD is defined by a primitive whose facets map directly to the facets of an XSD built-in data type, the BDT MUST be defined as a named <code>xsd:simpleType</code> .	1
[R B91F]	Every BDT whose content component BVD is defined by a primitive whose facets map directly to the facets of an <code>xsd:simpleType</code> MUST contain one <code>xsd:restriction</code> element.	1
[R 9910]	The <code>xsd:restriction</code> element used in a BDT content component BVD defined by a primitive MUST include an <code>xsd:base</code> attribute that defines the specific XSD built-in data type required for the content component.	1

2040 If a BDT uses a primitive type to express its content component BVD, it is defined
 2041 with a name that reflects the data type qualifiers and data type term and the primitive
 2042 type name.

[R A7B8]	The name of a BDT that uses a primitive to define its content component BVD MUST be the BDT <code>ccts:DataTypeQualifier(s)</code> if any, plus the <code>ccts:DataTypeTerm</code> , plus the primitive type name, followed by the word 'Type' with the separators removed and approved abbreviations and acronyms applied.	1
----------	---	---

2043 Example 8-20 provides three examples of BDT names, where primitives are used.

2044 Example 8-20: BDT Type Definition Names when Primitive is used

2045
 2046
 2047
 2048
 2049
 2050
 2051
 2052
 2053
 2054

```
CodeTokenType
Where Code is the Data Type Term and Token is the primitive.

PercentDecimalType
Where Percent is the Data Type Term and Decimal is the primitive.

AstronomicalUnitFloatType
Where Astronomical Unit is the Data Type Qualifier, Amount is the Data Type Term,
and Float is the primitive.
```

2055 **8.4.4.2 Content Component Business Value Domain Expressed By Code List**

2056 If a BDT uses a single BCL or CCL to define its content component BVD, it is defined
 2057 as an **xsd:simpleType** that contains an **xsd:restriction** element whose
 2058 **xsd:base** attribute is set to the code lists defined **xsd:simpleType** (See Section
 2059 8.5.1.4).

[R AA60]	A BDT whose content component BVD is defined as an xsd:simpleType whose base is a single code list MUST contain an xsd:restriction element with the xsd:base attribute set to the code lists defined xsd:simpleType .	1
----------	--	---

2060 The name of a BDT that uses a single code list directly reflects the data type
 2061 qualifiers and data type term and a code list suffix.

[R 8DB1]	<p>The name of A BDT that uses a single code list to define its content component BVD MUST be its ccts:DataTypeQualifier(s) if any, plus the ccts:DataTypeTerm, plus the code list suffix, followed by the word 'Type' with the separators removed and approved abbreviations and acronyms applied.</p> <p>The code list suffix MUST be the following: (Any repeated words are eliminated.)</p> <ul style="list-style-type: none"> • <Code List Agency Identifier Code List Agency Name><Code List Identification Identifier Code List Name> <p>Where.</p> <ul style="list-style-type: none"> • Code List Agency Identifier – is the identifier for the agency that code list is from. • Code List Agency Name – is the name of the agency that maintains the code list. • Code List Identification Identifier – is the identifier for the given code list. • Code List Name – is the name for the code list. 	1
----------	--	---

2062 Example 8-21 shows examples of BDT definition names where the code lists used
 2063 are expressed in the type definition names.

2064 **Example 8-21: BDT Type Definition Names**

```

2065 Amount54217Type
2066 Where:
2067 5 is the Code List Agency Identifier
2068 4217 is the Code List Identification Identifier
  
```

2069 Example 8-22 shows a declaration using a code list in a BDT.

2070 **Example 8-22: BDT type definition using one code list**

```

2071 <xsd:simpleType name="TemperatureMeasureUnitCodeContentType">
2072   <xsd:annotation>
2073     ... see annotation ...
2074   </xsd:annotation>
2075   <xsd:restriction
2076     base="clm6Recommendation20:MeasurementUnitCommonCodeContentType">
2077     <xsd:length value="3"/>
2078     <xsd:enumeration value="BTU">
2079       <xsd:annotation>
2080         <xsd:documentation xml:lang="en">
2081           <ccts:Name>British thermal unit</ccts:Name>
2082         </xsd:documentation>
2083       </xsd:annotation>
2084     </xsd:enumeration>
2085     <xsd:enumeration value="CEL">
2086       <xsd:annotation>
2087         <xsd:documentation xml:lang="en">
2088           <ccts:Name>degree Celsius</ccts:Name>
2089         </xsd:documentation>
2090       </xsd:annotation>
2091     </xsd:enumeration>
2092     <xsd:enumeration value="FAH">
2093       <xsd:annotation>
2094         <xsd:documentation xml:lang="en">
2095           <ccts:Name>degree Fahrenheit</ccts:Name>
2096         </xsd:documentation>
2097       </xsd:annotation>
2098     </xsd:enumeration>
2099   </xsd:restriction>
2100 </xsd:simpleType>

```

2101 **8.4.4.3 Business Value Domain Expressed By Multiple Code Lists**

2102 If a BDT content component BVD is defined as a choice of two or more code lists, it
 2103 will be defined as a **xsd:simpleType** that contains an **xsd:union** element whose
 2104 **xsd:memberType** attribute includes the **xsd:simpleType** definitions of the code
 2105 lists to be included.

[R AAD1]	A BDT whose content component BVD is defined by a choice of two or more code lists MUST be defined as an xsd:simpleType that contains an xsd:union element whose xsd:memberType attribute includes the xsd:simpleType definitions of the code lists to be included.	1
----------	--	---

2106 The name of a BDT that uses multiple code lists reflects the data type qualifiers and
 2107 data type term and a suffix that uniquely points to the unioned code list.

[R 973C]	<p>The name of a BDT that uses multiple code lists MUST be it's ccts:DataTypeQualifier(s) if any, plus the ccts:DataTypeTerm, plus the code list suffix, followed by the word 'Type' with the separators removed and approved abbreviations and acronyms applied.</p> <p>The suffix MUST be the following: (Any repeated words are eliminated)</p> <ul style="list-style-type: none"> • <Code List Agency Identifier Code List Agency Name><Code List Identification 	1
----------	--	---

	Identifier Code List Name>	
	<p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Identifier – is the identifier for the agency that code list is from. • Code List Agency Name – is the name of the agency that maintains the code list. • Code List Identification Identifier – is the identifier for the given code list. • Code List Name – is the name for the code list. 	

2108 Example 8-23 shows an example of using two code lists in a BDT.

2109 **Example 8-23: Combination of Two Code Lists**

```

2110 <xsd:simpleType name="AccountDutyCodeclm64437clm65153Type">
2111 <xsd:annotation>
2112 ... see annotation ...
2113 </xsd:annotation>
2114 <xsd:union memberType="clm64437:AccountTypeCodeContentType
2115 clm65153:DutyTaxFeeTypeCodeContentType"/>
2116 </xsd:simpleType>

```

2117 8.4.4.4 Content Component Business Value Domain Expressed By Identifier 2118 Scheme

2119 If a BDT uses an identifier scheme to define its content component BVD, the BDT is
2120 defined as an **xsd:simpleType** that contains an **xsd:restriction** element
2121 whose **xsd:base** attribute is set to the identifier scheme defined **xsd:simpleType**
2122 (See Section X.X).

[R A861]	<p>If a BDT content component BVD is defined as an xsd:simpleType whose base is an identifier scheme, it MUST contain an xsd:restriction element with the xsd:base attribute set to the identifier scheme defined xsd:simpleType.</p>	1
----------	--	---

2123 The name of a BDT that uses an identifier scheme to define its content component
2124 BVD reflects the data type qualifiers and data type term and an identifier scheme
2125 suffix.

[R 8F96]	<p>The name of A BDT that uses an identifier scheme to define its content component BVD MUST be its ccts:DataTypeQualifier(s) if any, plus the ccts:DataTypeTerm, plus the identifier scheme suffix, followed by the word 'Type' with the separators removed and approved abbreviations and acronyms applied.. The code list suffix MUST be the following: (Any repeated words are eliminated.)</p> <ul style="list-style-type: none"> • <Identifier Scheme Agency Identifier Identifier Scheme Agency Name><Identifier Scheme Identification 	1
----------	---	---

	Identifier Identifier Scheme Name>	
	<p>Where.</p> <ul style="list-style-type: none"> • Identifier Scheme Agency Identifier – is the identifier for the agency that code list is from. • Identifier Scheme Agency Name – is the name for the Agency that owns the identifier scheme. • Identifier Scheme Identification Identifier – is the identifier for the given identifier scheme. • Identifier Scheme Name – is the name for the identifier scheme. 	

2126 Example 8-24 shows examples of BDT definition names where the identifier scheme
2127 used are expressed in the type definition names.

2128 Example 8-24: BDT Type Definition Names

2129 `Amount54217Type`
2130 Where:
2131 5 is the Code List Agency Identifier
2132 4217 is the Code List Identification Identifier

2133 Example 8-25 shows an example of a BDT that uses an Identifier Scheme type.

2134 Example 8-25: Using an Identifier Scheme for a BDT content component BVD

2135 `<xsd:simpleType name="SocialSecurityIdentifierType">`
2136 `<xsd:annotation>`
2137 `... see annotation ...`
2138 `</xsd:annotation>`
2139 `<xsd:restriction base="xxxxx ContentType">`
2140 `<xsd:length value="9"/>`
2141 `</xsd:restriction>`
2142 `</xsd:simpleType>`

2143 8.4.4.5 BDT with Supplementary Components

2144 Supplementary components refine the BDT content component by providing
2145 additional information. Every BDT has zero or more Supplementary Components. If
2146 a BDT has supplementary components, and those supplementary components do
2147 not map directly to the facets of an XSD built-in datatype, the BDT will be defined as
2148 an **xsd:complexType** with **xsd:simpleContent** and an **xsd:extension**
2149 element whose **base** attribute is set to either a primitive type or an identifier scheme
2150 or a code list. Each Supplementary Component is expressed as an
2151 **xsd:attribute** whose **name** is set to the DEN of the given Supplementary
2152 Component.

[R AB05]	Every BDT that includes one or more Supplementary Components MUST be defined as an xsd:complexType	1
[R AAA5]	Every BDT xsd:complexType definition MUST have an xsd:simpleContent expression whose xsd:extension base	1

	attribute is set to the primitive type or scheme or list that defines its Content Component Business Value Domain.	
[R 890A]	Every BDT <code>xsd:complexType</code> definition MUST include an <code>xsd:attribute</code> declaration for each Supplementary Component.	1
[R ABC1]	The name of the Supplementary Component <code>xsd:attribute</code> must be the DEN of the Supplementary Component with periods, spaces, and other separators removed.	1

2153 The name of a BDT that is defined as an `xsd:complexType` will be unique and will
 2154 reflect the primitive or scheme or list that represents its content component business
 2155 value domain.

[R 90FB]	<p>The name of a BDT that includes one or more Supplementary Components MUST be:</p> <ul style="list-style-type: none"> • The BDT <code>ccts:DataTypeQualifier(s)</code> if any, plus • The <code>ccts:DataTypeTerm</code>, plus • The suffix of the Content Component Business Value Domain where: <ul style="list-style-type: none"> ○ The suffix is the primitive type name, the code list token, the series of code list tokens, or the identifier scheme token. <p>plus</p> <ul style="list-style-type: none"> • The <code>ccts:DictionaryEntryName</code> for each Supplementary Component present following the order defined in the Data Type Catalogue, plus • The suffix that represents the Supplementary Component BVD where the suffix is the primitive type name, the code list token, the series of code list tokens, or the identifier scheme token, plus • The word 'Type'. • With all separators removed and approved abbreviations and acronyms applied. 	1
----------	---	---

2156 Example 8-26 shows an example of a data type with a content component primitive
 2157 and a Supplementary Component that contains a code list.

2158 **Example 8-26: Business Data type with a content component primitive BVD and a**
 2159 **Supplementary Component that contains a code list**

2160
 2161
 2162
 2163
 2164
 2165
 2166
 2167
 2168

```

<xsd:complexType name="AmountDecimalCurrencyCodeC1m54217Type">
  <xsd:annotation>
    ... see annotation ...
  </xsd:annotation>
  <xsd:simpleContent>
    <xsd:extension base="xsd:decimal">
      <xsd:attribute name="currencyCode" type="clm54217:CurrencyCodeContentType" use="optional">
        <xsd:annotation>
          ... see annotation ...
        </xsd:annotation>
      </xsd:attribute>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>

```

2169
2170
2171
2172
2173

```

</xsd:annotation>
  </xsd:attribute>
  </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>

```

2174 8.4.4.6 Restricted BDTs

2175 BDTs may have either their content component, and/or supplementary component
2176 restricted. At the data model level, restrictions can take the form of restrictions to the
2177 Business Value Domain (BVD) of the BDT content component or supplementary
2178 component. Restrictions can also take the form of restrictions to the cardinality of the
2179 BDT supplementary component – to include the presence or absence of the
2180 supplementary component. Restrictions to the BVD can be in the form of restrictions
2181 to the primitive facets or to the scheme or list used to define the value domain.

2182 At the XML level, restrictions can take the form of restrictions to the BDT content
2183 component BVD. This is accomplished by creating a new restricted BDT
2184 `xsd:simpleType` derived from the less restricted or unrestricted BDT `xsd:simpleType`.
2185 Restrictions can also take the form of restrictions to the supplementary component
2186 BVD. This is accomplished by creating a new restricted BDT `xsd:complexType`
2187 that is derived from from the less qualified or unqualified BDT `xsd:complexType`.

2188 Restrictions can also take the form of restrictions to the BDT content or
2189 supplementary component BVD. This is also accomplished by creating a new
2190 restricted BDT that is derived from the less restricted or unrestricted BDT
2191 `xsd:complexType`.

[R 80FD]	Every restricted BDT XML Schema Component <code>xsd:type</code> definition MUST be derived from its base type using <code>xsd:restriction</code> unless a non-standard variation from the base type is required.	1
----------	--	---

2192 Non-standard variations are defined as those that are outside the bounds of the
2193 normally defined BVD for the underlying BDT. If non-standard variations from the
2194 base type are required, these will be defined as an `xsd:restriction` derivation
2195 from a custom type.

[R A9F6]	Every restricted BDT XML Schema Component <code>xsd:type</code> definition requiring a non-standard variation from its base type MUST be derived from a custom type.	1
----------	--	---

2196 [Note:]

2197 If a non-standard variation of the standard date time built-in data types is required,
2198 for example year month, then a BDT of the Core Data Type `TextType` needs to be
2199 defined, with the appropriate restrictions specified, e.g. a pattern, to specify the
2200 required format.

2201 Example 8-27 shows a restricted BDT definition.

2202 Example 8-27: Restricted BDT Type Definitions

2203

```

<!-- ----- -->

```

```

2204 <!-- ===== Type Definitions ===== -->
2205 <!-- ===== Business Data Type based on DateTime Type ===== -->
2206 <!-- ===== Day_Date. Type ===== -->
2207 <!-- ===== Day_Date. Type ===== -->
2208 <!-- ===== Day_Date. Type ===== -->
2209 <!-- ===== Day_Date. Type ===== -->
2210 <xsd:simpleType name="DayDateType">
2211 <xsd:annotation>
2212 <!-- ... see annotation ... -->
2213 </xsd:annotation>
2214 <xsd:restriction base="xsd:gDay"/>
2215 </xsd:simpleType>
2216 ...
2217 <!-- ===== Description Text. Type ===== -->
2218 <!-- ===== Description Text. Type ===== -->
2219 <!-- ===== Description Text. Type ===== -->
2220 <xsd:complexType name="DescriptionTextType">
2221 <xsd:annotation>
2222 <!-- ... see annotation ... -->
2223 </xsd:annotation>
2224 <xsd:simpleContent>
2225 <xsd:restriction base="bdt:TextType"/>
2226 </xsd:simpleContent>
2227 </xsd:complexType>
2228 ...
2229 <!-- ===== Uniform Resource Identifier. Type ===== -->
2230 <!-- ===== Uniform Resource Identifier. Type ===== -->
2231 <!-- ===== Uniform Resource Identifier. Type ===== -->
2232 <xsd:simpleType name="URIType">
2233 <xsd:annotation>
2234 <!-- ... see annotation ... -->
2235 </xsd:annotation>
2236 <xsd:restriction base="xsd:anyURI"/>
2237 </xsd:simpleType>
2238 ...
2239 <!-- ===== Country Identifier. Type ===== -->
2240 <!-- ===== Country Identifier. Type ===== -->
2241 <!-- ===== Country Identifier. Type ===== -->
2242 <xsd:simpleType name="CountryIDType">
2243 <xsd:annotation>
2244 <!-- ... see annotation ... -->
2245 </xsd:annotation>
2246 <xsd:restriction base="ids53166:CountryCodeContentType"/>
2247 </xsd:simpleType>
2248 ...

```

2249 8.4.4.6.1 Restrictions to Content Component

2250 Restrictions to the content component result in the creation of a new qualified BDT.
 2251 through restriction to the allowed **ccts:ContentComponent** and/or
 2252 **ccts:SupplementaryComponent** primitive facets of the unrestricted BDT type
 2253 definition, or through restrictions to the common code list, business code list,
 2254 common identifier scheme or business identifier scheme used to define the BVD
 2255 when those are used in lieu of a primitive.

2256 8.4.4.6.2 Restrictions to Supplementary Component

2257 Restrictions to the supplementary component result in the creation of a new qualified
 2258 BDT
 2259 through restriction to the allowed **ccts:ContentComponent** and/or
 2260 **ccts:SupplementaryComponent** primitive facets of the unrestricted BDT type
 2261 definition, or through restrictions to the common code list, business code list,
 2262 common identifier scheme or business identifier scheme used to define the BVD
 2263 when those are used in lieu of a primitive.

2264 **8.4.5 Attribute and Element Declarations**

2265 There are no element declarations in the BDT XML Schema Files. The only allowed
 2266 attributes are supplementary components, which are defined locally in the BDT.

[R 8B3D]	Global <code>xsd:element</code> declarations MUST NOT occur in the BDT XML Schema File.	1
[R B340]	Global <code>xsd:attribute</code> declarations MUST NOT occur in the BDT XML Schema File.	1
[R ACA7]	In the BDT XML Schema File, local <code>xsd:attribute</code> declarations MUST only represent CCTS Supplementary Components for the BDT for which they are declared.	1

2267 **8.4.6 Annotations**2268 **8.4.6.1 Annotation Documentation**2269 **8.4.6.1.1 BDT Types**

2270 Every BDT element and type declaration must include structured annotation
 2271 documentation.

[R BFE5]	<p>Every BDT definition MUST contain a structured set of annotation documentation in the following sequence and pattern:</p> <ul style="list-style-type: none"> • UniqueID (mandatory): The unique identifier that identifies the BDT in a unique and unambiguous way. • VersionID (mandatory): An unique identifier that identifies the version of the BDT. • DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the BDT. • Definition (mandatory): The semantic meaning of the BDT. • BusinessTermName (optional, repeating): A synonym term in which the BDT is commonly known. • PropertyTermName (mandatory): Represents a distinguishing characteristic of the BDT and shall occur naturally in the definition. • DataTypeName (mandatory): The name of the DataType. The possible values for the DataType are defined in the Data Type Catalogue. • DataTypeQualifierName (mandatory): Is a word or words which help define and differentiate a Data Type. It further enhances the semantic meaning of the DataType. • DefaultIndicator (mandatory): Indicates that the specific Code List Value is the default for the Code List. 	1
----------	--	---

	<ul style="list-style-type: none"> • DefaultValue (optional): Is the default value. • DefaultValueSource (optional): Indicates the source for the default value. • SchemeOrListID (optional): The unique identifier assigned to the scheme or list that uniquely identifies it. • SchemeOrListAgencyID (optional): The unique identifier assigned to the Agency that owns or is responsible for the Scheme or Code List being referenced. • SchemeOrListAgencyName (optional): The name of the Agency that owns or is responsible for the Scheme or Code List being referenced. • SchemeOrListModificationAllowedIndicator (optional): Indicates whether the values being validated can be outside the enumerations specified by the Scheme or Code List. • SchemeOrListName (optional): Name of the Scheme or Code List. • SchemeOrListBusinessTermName (optional, repeating): A synonym term under which the Scheme or Code List is commonly known and used in business. (BusinessTerm) 	
--	---	--

2272 Example 8-28 shows the annotation documentation structure declaration for each
2273 BDT.

2274 **Example 8-28: BDT annotation documentation definition**

```

2275 <xsd:group name="BDTDocumentation">
2276   <xsd:sequence>
2277     <xsd:element name="UniqueID"
2278     type="bdt:EntityUniqueIdentifierType"/>
2279     <xsd:element name="VersionID" type="bdt:VersionIdentifierType"/>
2280     <xsd:element name="DictionaryEntryName" type="bdt:NameType"/>
2281     <xsd:element name="Definition" type="bdt:TextType"/>
2282     <xsd:element name="BusinessTermName" minOccurs="0"
2283     maxOccurs="unbounded"/>
2284     <xsd:element name="PropertyTermName" type="bdt:NameType"/>
2285     <xsd:element name="DataTypeName" type="bdt:NameType"/>
2286     <xsd:element name="DataTypeQualifierName" type="bdt:NameType"/>
2287     <xsd:element name="DefaultIndicator" type="bdt:IndicatorType"/>
2288     <xsd:element name="DefaultValue" type="bdt:TextType"
2289     minOccurs="0"/>
2290     <xsd:element name="DefaultValueSource" type="bdt:TextType"
2291     minOccurs="0"/>
2292     <xsd:element name="SchemeOrListID" type="bdt:IDType"
2293     minOccurs="0"/>
2294     <xsd:element name="SchemeOrListAgencyID" type="bdt:IDType"
2295     minOccurs="0"/>
2296     <xsd:element name="SchemeOrListAgencyName" type="bdt:NameType"
2297     minOccurs="0"/>
2298     <xsd:element name="SchemeOrListModificationAllowedIndicator"
2299     type="bdt:IndicatorType" minOccurs="0"/>
2300     <xsd:element name="SchemeOrListName" type="bdt:NameType"
2301     minOccurs="0"/>
2302     <xsd:element name="SchemeOrListBusinessTermName"
2303     type="bdt:NameType" minOccurs="0" maxOccurs="unbounded"/>
2304   </xsd:sequence>
2305 </xsd:group>

```

2306 Example 8-29 shows an example annotation documentation of a BDT.

2307 **Example 8-29: BDT annotation element**

```

2308 ... see type definition ...
2309 <xsd:annotation>
2310   <ccts:UniqueID>UNDT000000-000</ccts:UniqueID>
2311   <ccts:VersionID>0.00</ccts:VersionID>
2312   <ccts:DictionaryEntryName></ccts:DictionaryEntryName>
2313   <ccts:Definition></ccts:Definition>
2314   <ccts:DataTypeName></ccts:DataTypeName>
2315   <ccts:DataTypeQualifierName></ccts:DataTypeQualifierName>
2316   <ccts:DefaultIndicator>true</ccts:DefaultIndicator>
2317   <ccts:DefaultValue></ccts:DefaultValue>
2318   <ccts:DefaultValueSource></ccts:DefaultValueSource>
2319   <ccts:SchemeOrListID></ccts:SchemeOrListID>
2320   <ccts:SchemeOrListVersionID></ccts:SchemeOrListVersionID>
2321   <ccts:SchemeOrListAgencyID></ccts:SchemeOrListAgencyID>
2322   <ccts:SchemeOrListAgencyName></ccts:SchemeOrListAgencyName>
2323 <ccts:SchemeOrListModificationAllowedIndicator><ccts:SchemeOrListModificationAllowe
2324 dIndicator>
2325   <ccts:SchemeOrListName></ccts:SchemeOrListName>
2326   <ccts:SchemeOrListBusinessTermName></ccts:SchemeOrListBusinessTermName>
2327 </xsd:documentation>
2328 </xsd:annotation>
2329 ... see type definition ...

```

2330 8.4.6.1.2 BDT Type Supplementary Components

2331 Every BDT Supplementary Component attribute declaration must include structured
2332 annotation documentation.

[R 9C95]	<p>Every supplementary component xsd:attribute declaration MUST contain a structured set of annotation documentation MUST in the following pattern:</p> <ul style="list-style-type: none"> • Cardinality (mandatory): Indicates the cardinality of the SC within the containing BDT. • PropertyTermName (mandatory): Represents a distinguishing characteristic of the SC and shall occur naturally in the definition. • RepresentationTermName (mandatory): An element of the component name that describes the form in which the SC is represented. • PrimitiveTypeName (mandatory): The name of the SC PrimitiveType. • DataTypeName (mandatory): The name of the DataType. The possible values for the DataType are defined in the Data Type Catalogue. • DataTypeQualifierName (mandatory): A word or words which help define and differentiate a Data Type. It further enhances the semantic meaning of the DataType. • DefaultIndicator (mandatory): Indicates that the specific Code List Value is the default for the Code List or identifier scheme. 	1
----------	--	---

	<ul style="list-style-type: none"> • DefaultValue (optional): Is the default value. • DefaultValueSource (optional): Indicates the source for the default value. • SchemeOrListID (optional): The unique identifier assigned to the scheme or list that uniquely identifies it. • SchemeOrListAgencyID (optional): The unique identifier assigned to the Agency that owns or is responsible for the identifier scheme or code list being referenced. • SchemeOrListAgencyName (optional): The name of the Agency that owns or is responsible for the identifier scheme or code list being referenced. • SchemeOrListModificationAllowedIndicator (optional): Indicates whether the values being validated can be outside the enumerations specified by the identifier scheme or code list. • SchemeOrListName (optional): Name of the identifier scheme or code list. • SchemeOrListBusinessTermName (optional, repeating): A synonym term under which the identifier scheme or code list is commonly known and used in business. (BusinessTerm) 	
--	--	--

2333 Example 8-30 shows the annotation documentation definition for each BDT SC.

2334 **Example 8-30: BDT SC annotation documentation definition**

2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361

```

<xsd:group name="BDTSCDocumentation">
  <xsd:sequence>
    <xsd:element name="Cardinality" type="bdt:NumericType"/>
    <xsd:element name="PropertyTermName" type="bdt:NameType"/>
    <xsd:element name="RepresentationTermName" type="bdt:NameType"/>
    <xsd:element name="PrimitiveTypeName" type="bdt:NameType"/>
    <xsd:element name="DataTypeName" type="bdt:NameType"/>
    <xsd:element name="DataTypeQualifierName" type="bdt:NameType"/>
    <xsd:element name="DefaultIndicator" type="bdt:IndicatorType"/>
    <xsd:element name="DefaultValue" type="bdt:TextType"
minOccurs="0"/>
    <xsd:element name="DefaultValueSource" type="bdt:TextType"
minOccurs="0"/>
    <xsd:element name="SchemeOrListID" type="bdt:IDType"
minOccurs="0"/>
    <xsd:element name="SchemeOrListAgencyID" type="bdt:IDType"
minOccurs="0"/>
    <xsd:element name="SchemeOrListAgencyName" type="bdt:NameType"
minOccurs="0"/>
    <xsd:element name="SchemeOrListModificationAllowedIndicator"
type="bdt:IndicatorType" minOccurs="0"/>
    <xsd:element name="SchemeOrListName" type="bdt:NameType"
minOccurs="0"/>
    <xsd:element name="SchemeOrLisBusinessTermName"
type="bdt:NameType" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:group>

```

2362 8.4.6.2 Annotation Application Information (AppInfo)

2363 The annotation `xsd:appInfo` is expressed for all BDT artifacts defined in BDT XML
 2364 Schema Files. The UsageRules and the context is communicated as defined in
 2365 section [7.5.2, Application Information \(AppInfo\)](#). All UsageRules and contexts in
 2366 which the BDT is applicable is expressed in the `xsd:appInfo`.

2367 8.5 Code List XML Schema Files

2368 Codes are an integral component of any information flow. Codes have been
 2369 developed over time to facilitate the flow of compressed, standardized values that
 2370 can be easily validated for correctness to ensure consistent data. In order for XML
 2371 Instance documents to be fully validated by parsers, any codes used within the XML
 2372 document need to be available as part of the schema validation process. Many
 2373 international, national and sectorial agencies create and maintain code lists relevant
 2374 to their area. If required to be used within an information flow, these code lists will be
 2375 stored in their own XML Schema File, and are referred to as Common Code Lists.
 2376 For example, many of the code lists that exist in the United Nations Code List
 2377 (UNCL) will be stored as Common Code List XML Schema Files for use within other
 2378 UN/CEFACT XML Schema Files.

[R 9E40]	Each code list used by a BDT or BBIE MUST be defined in its own XML Schema File.	2
----------	--	---

2379 UN/CEFACT recognizes two basic types of code lists:

- 2380 • Common Code List (CCL) – Universally defined for use in all contexts.
 2381 Generally maintained by UN/CEFACT and other standards bodies.
- 2382 • Business Code List (BCL) which are defined within a given context of their
 2383 use. They may be defined as:
 - 2384 ○ A subset of an existing CCL or
 - 2385 ○ Additions to an existing CCL or
 - 2386 ○ A new Code List that is needed within the context of use for a given
 2387 context category namespace

2388 8.5.1 General Code List XML Schema Components

2389 Both Common Code List XML Schema Files and Business Code List XML Schema
 2390 Files define codes using a consistent approach.

2391 8.5.1.1 Code List XML Schema File Structure

2392 Each Code List XML Schema File will be structured in a standard format in order to
 2393 ensure consistency and ease of use. This structure is show in Example 8-31.

2394 **Example 8-31: Code List XML Schema File structure**

```

2395 <?xml version="1.0" encoding="UTF-8"?>
2396 <!-- ===== -->
2397 <!-- ===== 6Recommendation20 - Code List XML Schema File ===== -->
```

2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426

```

<!-- ===== -->
<!--
Schema agency:      UN/CEFACT
Schema version:     2.0
Schema date:        16 January 2006

Code list name:     Measurement Unit Common Code
Code list agency:   UNECE
Code list version:   3

Copyright (C) UN/CEFACT (2006). All Rights Reserved.

... see copyright information ...

-->
<xsd:schema targetNamespace=" ... see namespace ...
             xmlns:xsd="http://www.w3.org/2001/XMLSchema"
             elementFormDefault="qualified" attributeFormDefault="unqualified">
<!-- ===== -->
<!-- ===== Root Element ===== -->
<!-- ===== -->
... see root element declaration ...
<!-- ===== -->
<!-- ===== Type Definitions ===== -->
<!-- ===== -->
<!-- ===== Type Definition: Measurement Unit Common Code Content Type == -->
<!-- ===== -->
... see type definition ...
</xsd:schema>

```

2427 8.5.1.2 Code List XML Schema Name

2428 The name of Code List XML Schema Files are dependent upon the agency that
2429 defines them and the name of the code list itself.

[R 849E]	<p>Code List XML Schema File names MUST be of the form:</p> <p><Agency Identifier Agency Name>_<List Identification Identifier List Name>_<Version Identifier>.xsd</p> <p>All periods, spaces, or other separators are removed except for the “.” before xsd and the “_” between the names.</p> <p>Where:</p> <ul style="list-style-type: none"> • Agency Identifier – identifies the agency that manages the list. The default agencies used are those from DE 3055 but roles defined in DE 3055 cannot be used. • Agency Name – the name of the agency that maintains the list. • List Identification Identifier – identifies a list of the respective corresponding codes or ids. • List Name – the name of a list of codes. • Version Identifier – identifies the version. 	2
----------	--	---

2430 **8.5.1.3 Element Declarations**

2431 A Code List XML Schema File contains one global element declaration. This global
 2432 element is a unique identifier for the code list and is mandatory for UN/CEFACT
 2433 Code List XML Schema Files. Other organizations using this specification may
 2434 choose to not provide the Code List Root Element and still be in compliance with this
 2435 specification.

[R 8D1D]	Each Code List XML Schema File MUST declare a single global element.	3
----------	--	---

2436 The global element serves as the root element and is of the one `xsd:simpleType`
 2437 that is defined in the Code List XML Schema File.

[R BE84]	The Code List XML Schema File global element MUST be of the <code>xsd:simpleType</code> that is defined in the Code List XML Schema File.	3
----------	---	---

2438 Example 8-32 shows a root element declaration for a code list.

2439 **Example 8-32: Code list global root element declaration**

2440
2441
2442
2443

```
<!-- =====>
<!-- ===== Root Element =====>
<!-- =====>
<xsd:element name="AccountTypeCode" type="clm64437:AccountTypeCodeContentType"/>
```

2444 The actual implementation of the code list is through the use of its
 2445 `xsd:simpleType` by a BDT BVD or BBIE.

2446 **8.5.1.4 Type Definitions**

2447 Each Code List XML Schema File will have one named `xsd:simpleType` defined.
 2448 The name of this type will correspond to the code list name with the word
 2449 'ContentType' appended.

[R A8EF]	Each Code List XML Schema File MUST define one, and only one, named <code>xsd:simpleType</code> for the content component.	1
[R 92DA]	The Code List XML Schema File <code>xsd:simpleType</code> name MUST be the name of the code list root element with the word 'ContentType' appended.	1

2450 Code List contents are expressed using `xsd:enumeration`, where each value of
 2451 the code list is defined using `xsd:value`.

[R 962C]	Each code in a Code List XML Schema File MUST be expressed as <code>xsd:enumeration</code> , where the <code>xsd:value</code> for the enumeration is the actual code value.	1
----------	---	---

2452 Example 8-33 shows a simple type definition used in a code list.

2453 **Example 8-33: Code list xsd:simpleType definition**

```

2454 <!-- ===== -->
2455 <!-- ===== Type Definitions ===== -->
2456 <!-- ===== -->
2457 <!-- ===== Type Definition: Account Type Code ===== -->
2458 <!-- ===== -->
2459 <xsd:simpleType name="AccountTypeCodeContentType">
2460     <xsd:restriction base="xsd:token">
2461         <xsd:enumeration value="2">
2462             ... see enumeration ...
2463         </xsd:enumeration>
2464     </xsd:restriction>
2465 </xsd:simpleType>

```

2466 **8.5.1.5 Annotation**2467 **8.5.1.5.1 Annotation Documentation**2468 **8.5.1.5.1.1 Code List Documentation**

2469 Every Code List XML Schema file must include structured annotation documentation.

[R A142]	<p>Every Code List MUST contain a structured set of annotation documentation in the following sequence and pattern:</p> <ul style="list-style-type: none"> • SchemeOrListID (mandatory): The unique identifier assigned to the code list. • SchemeOrListAgencyID (optional): The unique identifier assigned to the Agency that owns or is responsible for the code list being referenced. • SchemeOrListAgencyName (optional): The name of the Agency that owns or is responsible for the code list being referenced. • SchemeOrListModificationAllowedIndicator (optional): Indicates whether the values being validated can be outside the enumerations specified by the code list. • SchemeOrListName (optional): Name of the code list. • SchemeOrListBusinessTermName (optional, repeating): A synonym term under which the code list is commonly known and used in business. (BusinessTerm) 	1
----------	---	---

2470 Example 8-34 shows the declaration of the code list documentation structure.

2471 **Example 8-34: Code list documentation structure**

```

2472 <xsd:group name="CodeListDocumentation">
2473     <xsd:sequence>
2474         <xsd:element name="SchemeOrListID" type="bdt:IDType"/>
2475         <xsd:element name="SchemeOrListVersionID" type="bdt:IDType"
2476 minOccurs="0"/>
2477         <xsd:element name="SchemeOrListAgencyID" type="bdt:IDType"
2478 minOccurs="0"/>
2479         <xsd:element name="SchemeOrListAgencyName" type="bdt:NameType"
2480 minOccurs="0"/>
2481         <xsd:element name="SchemeOrListName" type="bdt:NameType"
2482 minOccurs="0"/>
2483         <xsd:element name="SchemeOrListModificationAllowedIndicator"
2484 type="bdt:IndicatorType"/>

```

2485
2486
2487
2488

```

        <xsd:element name="SchemeOrListBusinessTermName"
type="bdt:NameType" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:group>

```

2489 **8.5.1.5.1.2 Code List Value Documentation**

2490 In order to facilitate a clear and unambiguous understanding of the list of allowable
2491 codes within an element, annotation documentation will be provided for each
2492 enumeration. This documentation will be the name of the value and a description of
2493 the code.

[R A814]	<p>Each code list <code>xsd:enumeration</code> MUST contain a structured set of annotations in the following sequence and pattern:</p> <ul style="list-style-type: none"> • Name (mandatory): The name of the code. • Description (optional): Descriptive information concerning the code. 	1
----------	--	---

2494 Example 8-35 shows the annotation documentation definition for the enumerations
2495 values of a code list.

2496 **Example 8-35: Code list enumeration annotation documentation**2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512

```

<xsd:simpleType name="PaymentMethodCodeContentType">
  <xsd:restriction base="xsd:token">
    <xsd:enumeration value="1"> Name (mandatory): The name of the
code.
Description (optional): Descriptive information concerning the code.
    <xsd:annotation>
      <xsd:documentation xml:lang="en">
        <ccts:Name>Direct payment</ccts:Name>
        <ccts:Description>An assigned invoice has
been paid by the buyer to the factor.</ccts:Description>
      </xsd:documentation>
    </xsd:annotation>
  </xsd:enumeration>
</xsd:restriction>
</xsd:simpleType>

```

2513 **8.5.2 Common Code List XML Schema Components**

2514 CCL's are universally defined for all contexts and maintained by standards bodies.
2515 CCL XML Schema Files will be imported into the context specific namespaces that
2516 use them.

2517 **8.5.2.1 Namespace Name for Common Code Lists**

2518 The namespace name for a CCL is somewhat unique in order to convey some of the
2519 supplementary components rather than including them as attributes. Specifically, the
2520 namespace structure for a code list extends the earlier rules for namespace names
2521 to include the code list name in the namespace.

[R 992A]	Code list XML Schema File namespaces MUST use the following pattern:	1
----------	--	---

	<table border="1"> <tr> <td style="width: 15%;">URN:</td> <td><code>urn:<organization>:<org hierarchy> *[:<org hierarchy level n>]:codelist:common:<major>:<status>:<name></code></td> </tr> <tr> <td>URL:</td> <td><code>http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/codelist/common/<major>/<status>/<name></code></td> </tr> </table>	URN:	<code>urn:<organization>:<org hierarchy> *[:<org hierarchy level n>]:codelist:common:<major>:<status>:<name></code>	URL:	<code>http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/codelist/common/<major>/<status>/<name></code>	
URN:	<code>urn:<organization>:<org hierarchy> *[:<org hierarchy level n>]:codelist:common:<major>:<status>:<name></code>					
URL:	<code>http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/codelist/common/<major>/<status>/<name></code>					
<p>Where:</p> <ul style="list-style-type: none"> • organization – Identifier of the organization providing the standard. • org hierarchy – The first level of the hierarchy within the organization providing the standard. • org hierarchy level – Zero to n level hierarchy of the organization providing the standard. • codelist – A fixed value token for common codelists. • common – A fixed value token for common codelists. • major – The Major version number of the codelist. • status – The status of the schema as: draft standard • name – The name of the XML Schema File (using upper camel case) with periods, spaces, or other separators and the words 'schema module' removed. <ul style="list-style-type: none"> ○ Code list names are further defined as: <Code List Agency Identifier Code List Agency Name> ><divider><Code List Identification Identifier Code List Name> <p>Where:</p> <ul style="list-style-type: none"> ▪ Code List Agency Identifier – is the identifier for the agency that code list is from. ▪ Code List Agency Name – is the name of the agency that maintains the code list. ▪ Divider – the divider character for URN is ':' the divider character for URL is '/'. ▪ Code List Identification Identifier – is the identifier for the given code list. ▪ Code List Name – is the name for the code list. 						

2522 Example 8-36 shows a namespace name of a code list using an agency and a code
2523 list identifier at draft status.

2524 **Example 8-36: Code list namespace name with an agency and a code list**
2525 **identifier at draft status**

```
2526 "urn:un:unece:unefact:codelist:common:D.04A:draft:6:3403: "  
2527 where  
2528 D.04A = the version of the UN/CEFACT directory  
2529 6 = the value for UN/ECE in UN/CEFACT data element 3055 representing  
2530 the Code List. Agency. Identifier  
2531 3403 = UN/CEFACT data element tag for Name type code representing  
2532 the Code List. Identification. Identifier
```

2533 Example 8-37 shows a namespace name of a proprietary code list at draft status.

2534

2535 **Example 8-37: Proprietary code list namespace name at draft status**

```

2536 "urn:un:unece:unefact:odelist:common:1:draft:Security_Initiative:Document_Securit
2537 y"
2538 where
2539 SecurityInitiative = the code list agency name of a responsible agency, which
2540 is not defined in UN/CEFACT data element 3055
2541 representing the Code List. Agency. Identifier
2542 DocumentSecurity = the value for Code List. Name. Text
2543 1.2 = the value for Code List. Version. Identifier

```

2544 Example 8-38 shows a namespace name of a code list with and agency and code
 2545 list identifier at standard status.

2546 **Example 8-38: Code list namespace name with an agency and a code list
 2547 identifier at standard status**

```

2548 "urn:un:unece:unefact:odelist:common:D.04A:standard:6:3403"
2549 where
2550 6 = the value for UN/ECE in UN/CEFACT data element 3055 representing
2551 the Code List. Agency. Identifier
2552 3403 = UN/CEFACT data element tag for Name status code representing
2553 the Code List. Identification. Identifier
2554 D.04A = the version of the UN/CEFACT directory

```

2555 Example 8-39 shows a namespace name of a proprietary code list with a status of
 2556 standard.

2557 **Example 8-39: Namespace name of proprietary code list at standard status**

```

2558 "urn:un:unece:unefact:odelist:common:1:standard:Security_Initiative:Document_Secu
2559 rity"
2560 where
2561 SecurityInitiative = the code list agency name of a responsible agency, which
2562 is not defined in UN/CEFACT data element 3055
2563 representing the Code List. Agency. Identifier
2564 DocumentSecurity = the value for Code List. Name. Text
2565 1.2 = the value for Code List. Version. Identifier

```

2566 While the versioning of code lists published by external organisations is outside of
 2567 the control of UN/CEFACT, UN/CEFACT published code lists expressed in XML
 2568 Schema Files will follow the rules expressed in this specification.

2569 **8.5.2.2 XML Schema Namespace Token for Common Code Lists**

2570 A unique token will be defined for each namespace for common code lists. The
 2571 token is constructed based on the identifier of the agency maintaining the code list
 2572 and the identifier of the specific code list as issued by the maintenance agency,
 2573 except where there is no identifier. When there is no identifier, the name for the
 2574 agency and/or code list should be used instead. This will typically be true when
 2575 proprietary code lists are used. This method of token construction will provide
 2576 uniqueness with a reasonably short token.

2577 The agency maintaining the code list will be identified either by the agency code as
 2578 specified in data element 3055 in the UN/CEFACT Code List directory, or the agency
 2579 name if the agency does not have a code value in 3055. The identifier of the specific
 2580 code list will be the data element tag of the corresponding list in the UN/CEFACT

2581 directory. If there is no corresponding data element, then the name of the code list
2582 will be used.

[R 9FD1]	<p>Each UN/CEFACT maintained CCL XML Schema File MUST be represented by a unique token constructed as follows:</p> <p>clm<Code List Agency Identifier Code List Agency Name><Code List Identification Identifier Code List Name></p> <p>Such that any repeated words are eliminated.</p> <p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Identifier – is the identifier for the agency that code list is from. • Code List Agency Name – is the name of the agency that maintains the code list. • Code List Identification Identifier – is the identifier for the given code list. • Code List Name – is the name for the code list. 	2
----------	--	---

2583 Example 8-40 shows a code list token with an agency and code list identifier.

2584 **Example 8-40: Code list token with an agency and a code list identifier**

```
2585 The code list token for Name Type. Code is clm63403
2586 where
2587 6 = the value for UN/ECE in UN/CEFACT data element 3055 representing
2588 the Code List. Agency. Identifier
2589 3403 = UN/CEFACT data element tag for Name status code representing
2590 the Code List. Identification. Identifier
```

2591 Example 8-41 shows a code list token for a business data type with an agency and
2592 code list identifiers.

2593 **Example 8-41: Code list token for a qualified BDT with an agency and code list**
2594 **identifiers**

```
2595 Code list token for Person_Name Type. Code is clmPersonNameType63403
2596 where
2597 PersonNameType = name of the qualified data type
2598 6 = the value for UN/ECE in UN/CEFACT data element 3055 representing
2599 the Code List. Agency. Identifier
2600 3403 = UN/CEFACT data element tag for Name status code representing
2601 the Code List. Identification. Identifier
```

2602 Example 8-42 shows a code list token for a proprietary code list.

2603 **Example 8-42: Code list token for a proprietary code list**

```
2604 Code list token for a proprietary code list for Document Security is
2605 clmSecurityInitiativeDocumentSecurity
2606 where
2607 SecurityInitiative = the code list agency name of a repsonsible agency, which is
2608 not defined in UN/CEFACT data element 3055
2609 representing the Code List. Agency. Identifier
2610 DocumentSecurity = the value for Code List. Name. Text
```

2611 Based on the constructs identified in the above examples, a namespace declaration
2612 for a code list would appear as shown in Example 8-43.

2613 **Example 8-43: Target namespace declaration for a code list**

```
2614 <xsd:schema
2615     targetNamespace="urn:un:unece:uncefact:codelist:common:D.04A:draft:6:4437"
2616     xmlns:clm64437="urn:un:unece:uncefact:codelist:common:D.04A:draft:6:4437"
2617     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
2618     elementFormDefault="qualified" attributeFormDefault="unqualified">
```

2619 [Note:]

2620 Developers are encouraged to follow the above rules when customizing XML
2621 Schema for code lists to ensure that there are no namespace conflicts.

2622 **8.5.2.3 Imports and Includes**

2623 UN/CEFACT CCL XML Schema Files are standalone XML Schema Files and will not
2624 import or include any other XML Schema Files.

[R 86C8]	CCL XML Schema Files MUST NOT import or include any other XML Schema Files.	1
----------	---	---

2625 **8.5.2.4 Type Definitions**

2626 Each CCL XML Schema file will have a single `xsd:simpleType` defined. This type
2627 definition will have an `xsd:restriction` expression whose base is an XML
2628 Schema built-in data type. The `xsd:restriction` will be used to convey the
2629 content component enumeration value(s).

[R B40B]	Each CCL XML Schema File <code>xsd:simpleType</code> MUST use an <code>xsd:restriction</code> element whose base attribute is <code>xsd:token</code> .	1
----------	--	---

2630 Example 8-44 shows the simple type definition for a code list.

2631 **Example 8-44: CCL `xsd:simpleType` definition**

```
2632 <xsd:simpleType name="PaymentMethodCodeContentType">
2633     <xsd:restriction base="xsd:token">
2634         <xsd:enumeration value="1">
2635             <xsd:annotation>
2636                 See annotation
2637             </xsd:annotation>
2638         </xsd:enumeration>
2639     </xsd:restriction>
2640 </xsd:simpleType>...
```

2641 **8.5.2.5 Annotation**

2642 **8.5.2.5.1 Annotation Documentation**

2643 CCL XML Schema documentation follows the same structure as defined in section
2644 [8.5.1.4.1 Annotation Documentation](#) of this specification.

2645

2646 [8.5.2.5.2 Annotation Application Information \(AppInfo\)](#)

2647 Common code lists are applicable to all contexts and therefore do not have context
2648 specified within an `xsd:appInfo` element.

2649 **8.5.3 Business Code List XML Schema Components**

2650 Business code lists are Code List XML Schema Files that contain codes that are
2651 applicable within the context category for the namespace where it is defined. A BCL
2652 XML Schema file maybe used where an existing CCL XML Schema File needs to be
2653 extended, where no suitable CCL XML Schema exists, or where the context in which
2654 the code list is to be used only needs to make use of a subset of a CCL. This is
2655 accomplished by:

- 2656 • A combination of several individual code lists using `xsd:union`,
- 2657 • A new code list that is applicable for the context, or
- 2658 • Sub setting an existing code list using `xsd:restriction`.

[R 8F2D]	BCL XML Schema file MUST be used to <ul style="list-style-type: none"> • Extend existing CCL or • Define a codelist where one does not exist or • Restrict the value of a CCL for a context category 	1
----------	---	---

2659 **8.5.3.1 Namespace Name for Business Code Lists**

2660 BCLs use the namespace name for the context category in which it is defined. This
2661 is described earlier in this specification in section [5.6 Namespace Scheme](#).

2662 **8.5.3.2 UN/CEFACT XML Schema Namespace Token for Business Code Lists**

2663 BCL use the namespace token for the context category in which it is defined. This is
2664 described earlier in this specification in section [5.6.2 Namespace Tokens](#). In cases
2665 where the BCL is a restricted set of values of a published CCL, the BCL will be
2666 associated with a business data type, and the name of the business data type will be
2667 included as part of the namespace token to ensure uniqueness from the CCL XML
2668 Schema File.

2669 **8.5.3.3 Imports and Includes**

2670 BCL Schema Files may import CCL XML Schema File(s) if the BCL restricts the CCL
2671 Schema File content or unions multiple CCL content to create a new BCL.

[R 87A9]	BCL XML Schema Files MUST import only CCL XML Schema Files it uses directly.	1
----------	--	---

2672 **8.5.3.4 Type Definitions**

2673 Each BCL XML Schema file will have a single `xsd:simpleType` defined. This type
 2674 definition will have a `xsd:restriction` expression whose base is an XML
 2675 Schema built-in data type or the `ContentType` (s) of the CCL the BCL is using. The
 2676 `xsd:restriction` will be used to convey the content component enumeration
 2677 value(s).

[R 882D]	In each BCL XML Schema File the <code>xsd:restriction</code> element base attribute value MUST be set to <code>xsd:token</code> or the ' <code>ContentType</code> ' from the CCL that is being used.	1
----------	--	---

2678 **8.5.3.5 Annotation**2679 **8.5.3.5.1 Annotation Documentation**

2680 BCL XML Schema documentation is the same as CCL XML Schema documentation
 2681 described in Section [8.5.1.4.1 Annotation Documentation](#).

2682 **8.5.3.5.2 Annotation Application Information (AppInfo)**

2683 BCL usage rules and context information is as defined in section [7.5.2, Application](#)
 2684 [Information \(AppInfo\)](#).

2685 **8.6 Identifier Scheme XML Schema Files**

2686 Identifiers are an integral component of managing business objects. Identifiers have
 2687 been developed over time to provide for uniquely identifying one object from another.
 2688 When identifiers are part of an XML based business information exchange, any
 2689 identifiers used within the XML document need to be able to be validated by the XML
 2690 parser as to the identifiers adherence to the scheme that defines it.

2691 Many international, national and sectorial agencies create and maintain identifier
 2692 schemes. If required to be used within an information flow, these schemes will be
 2693 defined in their own XML Schema File.

[R A1EE]	Each identifier scheme used by a BDT or BBIE MUST be defined in its own XML Schema file.	2
----------	--	---

2694 UN/CEFACT recognizes two basic types of identifier schemes:

- 2695 • Common Identifier Scheme (CIS) – Universally defined for use in all contexts.
 2696 Generally maintained by UN/CEFACT and other standards bodies.
- 2697 • Business Identifier Scheme (BIS) These are identifiers that are defined within
 2698 a given context of their use. The may be defined as:
 - 2699 ○ A restriction on the pattern or allowed values of an existing CIS
 - 2700 ○ An extension on the pattern or allowed values of an existing CIS
 - 2701 ○ A new CIS that is needed within the context of use for a given context
 2702 category namespace

2703 8.6.1 General Identifier Scheme XML Schema Components

2704 Both Common Identifier Scheme XML Schema Files and Business Identifier Scheme
2705 XML Schema Files define the schemes using a consistent approach.

2706 8.6.1.1 Identifier Scheme XML Schema File Structure

2707 Each Identifier Scheme XML Schema File will be structured in a standard format
2708 in order to ensure consistency and ease of use. This structure is show in Example 8-
2709 45.

2710 Example 8-45: Identifier scheme XML Schema File structure

```

2711 <?xml version="1.0" encoding="UTF-8"?>
2712 <!-- ===== -->
2713 <!-- ===== Global Trade Identification Number - Identifier Scheme XML Schema
2714 File===== -->
2715 <!-- ===== -->
2716 <!--
2717 Schema agency:      GS1
2718 Schema version:    1.0
2719 Schema date:       21 December 2008
2720
2721 Identifier Scheme name:      Global Trade Identification Number
2722 Identification Scheme agency:      GS1
2723 Identification Scheme version:    1
2724
2725 Copyright (C) UN/CEFACT (2008). All Rights Reserved.
2726
2727 ... see copyright information ...
2728
2729 -->
2730 <xsd:schema targetNamespace=" ... see namespace ...
2731 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
2732 elementFormDefault="qualified" attributeFormDefault="unqualified">
2733 <!-- ===== -->
2734 <!-- ===== Root Element ===== -->
2735 <!-- ===== -->
2736 ... see root element declaration ...
2737 <!-- ===== -->
2738 <!-- ===== Type Definitions ===== -->
2739 <!-- ===== -->
2740 <!--= Type Definition: Global Trade Identification Number Content Type =-->
2741 <!-- ===== -->
2742 ... see type definition ...
2743 </xsd:schema>

```

2744 8.6.1.2 Identifier Scheme XML Schema Name

2745 The name of Identifier Scheme XML Schema Files are dependent upon the agency
2746 that defines them and the name of the identifier scheme itself.

[R A50B]	<p>Identifier Scheme XML Schema File names MUST be of the form:</p> <p><Agency Identifier Agency Name>_<Scheme Identification Identifier Scheme Name>_<Version Identifier>.xsd</p> <p>All periods, spaces, or other separators are removed except for the “.” before xsd and the “_” between the names.</p> <p>Where:</p> <ul style="list-style-type: none"> • Agency Identifier – identifies the agency that manages the 	2
----------	---	---

	<p>identifier scheme. The default agencies used are those from DE 3055 but roles defined in DE 3055 cannot be used.</p> <ul style="list-style-type: none"> • Agency Name – the name of the agency that maintains the scheme. • Scheme Identification Identifier – identifies the identifier scheme. • Scheme Name – the name of the identifier scheme. • Version Identifier – identifies the version of the scheme. 	
--	---	--

2747 8.6.1.3 Element Declarations

2748 An Identifier Scheme XML Schema File contains one global element declaration.
 2749 This global element is a unique identifier for the identifier scheme and is mandatory
 2750 for UN/CEFACT Identifier Scheme XML Schema Files. Other organizations using
 2751 this specification may choose to not provide the Identifier Scheme Root Element and
 2752 still be in compliance with this specification.

[R BFEB]	Each Identifier Scheme XML Schema File MUST declare a single global element.	3
----------	--	---

2753 The global element serves as the root element and is of the one `xsd:simpleType`
 2754 that is defined in the Identifier Scheme XML Schema File.

[R B236]	The Identifier Scheme XML Schema File root element MUST be of the <code>xsd:simpleType</code> that is defined in the Identifier Scheme XML Schema File.	3
----------	---	---

2755 Example 8-46 shows a root element declaration for an identifier scheme.

2756 Example 8-46: Identifier scheme root element declaration

```

2757 <!-- =====>
2758 <!-- ===== Root Element =====>
2759 <!-- =====>
2760 <xsd:element name="GlobalTradeIdentificationNumber"
2761 type="ism8GTIN:GlobalTradeIdentificationNumberType"/>
  
```

2762 The actual implementation of the identifier scheme is through the use of its
 2763 `xsd:simpleType` by a BDT BVD or BBIE.

2764 8.6.1.4 Type Definitions

2765 Each Identifier XML Schema File will have one named `xsd:simpleType` defined.
 2766 The name of this type will correspond to the identifier scheme name with the word
 2767 'ContentType' appended.

[R 9451]	Each Identifier Scheme XML Schema File MUST define one, and only one, named <code>xsd:simpleType</code> for the content component.	1
----------	--	---

[R 92DA]	The Identifier Scheme XML Schema File <code>xsd:simpleType</code> name MUST be the name of the identifier scheme root element with the word 'ContentType' appended.	1
----------	---	---

2768 The identifiers created by an identifier scheme are never enumerated as shown in
2769 Example 8-47.

2770 **Example 8-47: Identifier scheme `xsd:simpleType` name**

```

2771
2772 <!-- ===== Root Element ===== -->
2773 <!-- ===== Root Element ===== -->
2774 <!-- ===== Root Element ===== -->
2775 <xsd:element name="GlobalTradeIdentificationNumber"
2776 type="ism8GTIN:GlobalTradeIdentificationNumberType"/>
2777 <!-- ===== Type Definitions ===== -->
2778 <!-- ===== Type Definitions ===== -->
2779 <!-- ===== Type Definitions ===== -->
2780 <!-- == Type Definition: Global Trade Identification Number Identifier= -->
2781 <!-- ===== Type Definitions ===== -->
2782 <xsd:simpleType name="GlobalTradeIdentificationNumberContentType">
2783 See type definition
2784 </xsd:simpleType>

```

2785 **8.6.1.5 Annotation**

2786 **8.6.1.5.1 Annotation Documentation**

2787 **8.6.1.5.1.1 Identifier Scheme Documentation**

2788 Every Identifier Scheme XML Schema file must include structured annotation
2789 documentation.

[R B30A]	<p>Every Identifier Scheme MUST contain a structured set of annotation documentation in the following sequence and pattern:</p> <ul style="list-style-type: none"> • SchemeOrListID (mandatory): The unique identifier assigned to the Identifier Scheme. • SchemeOrListAgencyID (optional): The unique identifier assigned to the Agency that owns or is responsible for the identifier scheme being referenced. • SchemeOrListAgencyName (optional): The name of the Agency that owns or is responsible for the identifier scheme being referenced. • SchemeOrListModificationAllowedIndicator (optional): Indicates whether the values being validated can be outside the pattern specified by the scheme. • SchemeOrListName (optional): Name of the identifier scheme. • SchemeOrListBusinessTermName (optional, repeating): A synonym term under which the identifier scheme is commonly known and used in business. (BusinessTerm) 	1
----------	---	---

2790 Example 8-48 shows the declaration of the annotation documentation for each
2791 Identifier Scheme.

2792 **Example 8-48: Identifier scheme documentation structure**

```

2793 <xsd:group name="CodeListDocumentation">
2794   <xsd:sequence>
2795     <xsd:element name="SchemeOrListID" type="bdt:IDType"/>
2796     <xsd:element name="SchemeOrListVersionID" type="bdt:IDType"
2797     minOccurs="0"/>
2798     <xsd:element name="SchemeOrListAgencyID" type="bdt:IDType"
2799     minOccurs="0"/>
2800     <xsd:element name="SchemeOrListAgencyName" type="bdt:NameType"
2801     minOccurs="0"/>
2802     <xsd:element name="SchemeOrListName" type="bdt:NameType"
2803     minOccurs="0"/>
2804     <xsd:element name="SchemeOrListModificationAllowedIndicator"
2805     type="bdt:IndicatorType"/>
2806     <xsd:element name="SchemeOrListBusinessTermName"
2807     type="bdt:NameType" minOccurs="0" maxOccurs="unbounded"/>
2808   </xsd:sequence>
2809 </xsd:group>

```

2810 **8.6.2 Common Identifier Scheme XML Schema Components**

2811 CIS are universally defined for all contexts and maintained by standards bodies. CIS
2812 XML Schema Files will be imported into the context specific namespaces that use
2813 them.

2814 **8.6.2.1 Namespace Name for Common Identifier Scheme**

2815 The namespace name for a CIS is somewhat unique in order to convey some of the
2816 supplementary components rather than including them as attributes. Specifically, the
2817 namespace structure for an identifier scheme extends the earlier rules for
2818 namespace names to include the identifier scheme name in the namespace.

[R 9CCF]	<p>Identifier scheme XML Schema File namespaces MUST use the following pattern:</p> <table border="1" data-bbox="435 1293 1304 1629"> <tr> <td data-bbox="443 1293 548 1461">URN:</td> <td data-bbox="548 1293 1304 1461">urn:<organization>:<org hierarchy> *[:<org hierarchy level n>]:identifierscheme:common:<major>:<status>:<name></td> </tr> <tr> <td data-bbox="443 1461 548 1629">URL:</td> <td data-bbox="548 1461 1304 1629">http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/identifierscheme/common/<major>/<status>/<name></td> </tr> </table> <p>Where:</p> <ul style="list-style-type: none"> • organization – Identifier of the organization providing the standard. • org hierarchy – The first level of the hierarchy within the organization providing the standard. • org hierarchy level – Zero to n level hierarchy of the 	URN:	urn:<organization>:<org hierarchy> *[:<org hierarchy level n>]:identifierscheme:common:<major>:<status>:<name>	URL:	http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/identifierscheme/common/<major>/<status>/<name>	1
	URN:	urn:<organization>:<org hierarchy> *[:<org hierarchy level n>]:identifierscheme:common:<major>:<status>:<name>				
URL:	http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/identifierscheme/common/<major>/<status>/<name>					

	<p>organization providing the standard.</p> <ul style="list-style-type: none"> • identifierscheme – A fixed value token for common identifier schemes. • common – A fixed value token for common identifier schemes. • major – The Major version number of the identifier scheme. • status – The status of the schema as: draft standard • name – The name of the XML Schema File (using upper camel case) with periods, spaces, or other separators and the words ‘schema module’ removed. <ul style="list-style-type: none"> ○ Identifier scheme names are further defined as: <Identifier Scheme Agency Identifier Identifier Scheme Agency Name> ><divider><Identifier Scheme Identification Identifier Identifier Scheme Name> <p>Where:</p> <ul style="list-style-type: none"> ▪ Identifier Scheme Agency Identifier – is the identifier for the agency that identifier scheme is from. ▪ Identifier Scheme Agency Name – is the name of the agency that maintains the identifier scheme. ▪ Divider – the divider character for URN is ‘:’ the divider character for URL is ‘/’. ▪ Identifier Scheme Identification Identifier – is the identifier for the given identifier scheme. ▪ Identifier Scheme Name – is the name for the identifier scheme. 	
--	--	--

2819 Example 8-49 shows an identifier scheme namespace where the status of the
2820 identifier scheme is in draft status.

2821 **Example 8-49: Identifier scheme namespace name with an agency and a**
2822 **identifier scheme identifier at draft status**

```

2823 "urn:un:unece:uncefact:identifierscheme:common:D.04A:draft:8:GTIN: "
2824 where
2825 D.04A = the version of the UN/CEFACT directory
2826 8 = the value for GS1 in UN/CEFACT data element 3055 representing
2827 the Identifier Scheme. Agency. Identifier
2828 GTIN = GS1 data element tag for Global Trade Identification Number representing
2829 the Identifier Scheme. Identification. Identifier

```

2830 While the versioning of identifier schemes published by external organisations is
2831 outside of the control of UN/CEFACT, UN/CEFACT published code lists expressed
2832 in XML Schema Files will follow the rules expressed in this specification.

2833 8.6.2.2 XML Schema Namespace Token for Common Identifier Schemes

2834 A unique token will be defined for each namespace for common identifier schemes.
 2835 The token is constructed based on the identifier of the agency maintaining the
 2836 identifier scheme and the identifier of the specific identifier scheme as issued by the
 2837 maintenance agency – except where there is no identifier. When there is no
 2838 identifier, the name for the agency and/or identifier scheme should be used instead.
 2839 This will typically be true when proprietary identifier schemes are used. This method
 2840 of token construction will provide uniqueness with a reasonably short token.

2841 The agency maintaining the identifier scheme will be identified either by the agency
 2842 code as specified in data element 3055 in the UN/CEFACT Code List directory, or
 2843 the agency name if the agency does not have a code value in 3055. The identifier of
 2844 the specific identifier scheme will be the data element tag of the corresponding list in
 2845 the UN/CEFACT directory. If there is no corresponding data element, then the name
 2846 of the identifier scheme will be used.

[R B2BC]	<p>Each UN/CEFACT maintained CIS XML Schema File MUST be represented by a unique token constructed as follows:</p> <pre> clm<Identifier Scheme Agency Identifier Identifier Scheme Agency Name><Identifier Scheme Identification Identifier Identifier Scheme Name> </pre> <p>Such that any repeated words are eliminated.</p> <p>Where:</p> <ul style="list-style-type: none"> • Identifier Scheme Agency Identifier – is the identifier for the agency that the identifier scheme is from. • Identifier Scheme Agency Name – is the name of the agency that maintains the identifier scheme. • Identifier Scheme Identification Identifier – is the identifier for the given identifier scheme. • Identifier Scheme Name – is the name for the identifier scheme. 	2
----------	--	---

2847 Example 8-50 shows an identifier scheme token.

2848 **Example 8-50: Identifier scheme token with an agency and an identifier** 2849 **scheme identifier**

```

2850 The identifier scheme token for Global Trade Identification Number Identifier is
2851 ism8gtin
2852 where
2853 8 = the value for GS1 in UN/CEFACT data element 3055 representing
2854 the Identifier Scheme. Agency. Identifier
2855 gtin = GS1 data element tag for Global Trade Identification Number representing
2856 the Identifier Scheme. Identification. Identifier
2857 ="unqualified">

```

2858 [Note:]
 2859 Developers are encouraged to follow the above rules when customizing XML
 2860 Schema for code lists to ensure that there are no namespace conflicts.

2861 8.6.2.3 Imports and Includes

2862 UN/CEFACT CIS XML Schema Files are standalone XML Schema Files and will not
 2863 import or include any other XML Schema Files.

[R A6C0]	CIS XML Schema Files MUST NOT import or include any other XML Schema Files.	1
----------	---	---

2864 8.6.2.4 Type Definitions

2865 Each CIS XML Schema file will have a single `xsd:simpleType` defined. This type
 2866 definition will have an `xsd:restriction` expression whose base is an XML
 2867 Schema built-in data type of `xsd:token`.

[R 9DDA]	Each CIS XML Schema File <code>xsd:simpleType</code> MUST use an <code>xsd:restriction</code> element whose base attribute value = <code>xsd:token</code> .	1
----------	---	---

2868 Example 8-51 shows an CIS simpleType definition.

2869 Example 8-51: CIS `xsd:simpleType` definition

2870
 2871
 2872

```
<xsd:simpleType name="GlobalTradeIdentificationNumberContentType">
  <xsd:restriction base="xsd:token"/>
</xsd:simpleType>
```

2873 A CIS XML Schema File is only identifying the metadata about the identifier scheme,
 2874 it is not defining the actual scheme itself since that information is publicly available.

2875 8.6.2.5 Annotation

2876 8.6.2.5.1 Annotation Documentation

2877 CIS XML Schema documentation follows the same structure as defined in section
 2878 [8.6.1.4.1 Annotation Documentation](#) of this specification.

2879 8.6.2.5.2 Annotation Application Information (AppInfo)

2880 Common identifier schemes are applicable to all context and therefore do not have
 2881 context specified within `xsd:appInfo`.

2882 8.6.3 Business Identifier Scheme XML Schema Components

2883 Business identifier schemes are Identifier Scheme XML Schema Files that define a
 2884 scheme that is applicable within a context category namespace. A BIS XML Schema
 2885 file may be used where an existing CIS XML Schema identifier scheme needs to be

2886 modified, or where no suitable CIS XML Schema exists. In all cases this is
2887 accomplished by creating a new identifier scheme. The BIS will:

- 2888 ○ Define a new CIS that is needed within the context of use for a given
2889 context category namespace
- 2890 ○ Redefine an existing CIS by defining:
- 2891 ▪ a restriction on the pattern or allowed values of an existing CIS
- 2892 ▪ An extension on the pattern or allowed values of an existing CIS

[R A1E3]	BIS XML Schema file MUST be used to <ul style="list-style-type: none"> • Define an identifier scheme where one does not exist or • Redefine an existing CIS 	1
----------	---	---

2893 **8.6.3.1 Namespace Name for Business Information Scheme**

2894 A BIS uses the namespace name for the context category in which it is defined. This
2895 is described earlier in this specification in section [5.6 Namespace Scheme](#).

2896 **8.6.3.2 UN/CEFACT XML Schema Namespace Token for Business Information** 2897 **Scheme**

2898 A BIS uses the namespace token for the context category in which it is defined. This
2899 is described earlier in this specification in section [5.6.2 Namespace Tokens](#).

2900 **8.6.3.3 Imports and Includes**

2901 BIS XML Schema Files do not import or include other XML Schema Files.

[R A4BF]	BIS XML Schema Files MUST NOT use <code>xsd:import</code> or <code>xsd:include</code> .	1
----------	--	---

2902 **8.6.3.4 Type Definitions**

2903 Each BIS XML Schema file will have a single `xsd:simpleType` defined. This type
2904 definition will have a `xsd:restriction` expression whose base is an XML
2905 Schema built-in data type of `xsd:token`. The `xsd:restriction` `xsd:token`
2906 facets may be used to define the actual identifier scheme as part of the type
2907 definition.

[R 96B0]	Each CIS XML Schema File <code>xsd:simpleType</code> MUST use an <code>xsd:restriction</code> element whose base attribute value is <code>xsd:token</code> .	1
----------	--	---

2908 Example 8-52 shows a BIS `simpleType` definition.

2909 **Example 8-52: BIS `xsd:simpleType` definition**

2910

```
<xsd:simpleType name="SupplyWarehouseIdentificationNumberContentType">
```

2911
2912

```
<xsd:restriction base="xsd:token">  
</xsd:simpleType>
```

2913

8.6.3.5 Annotation

2914

8.6.3.5.1 Annotation Documentation

2915

BIS XML Schema documentation is the same as CIS XML Schema documentation described in section [8.5.2.4.1 Annotation Documentation](#).

2916

2917

8.6.3.5.2 Annotation Application Information (AppInfo)

2918

BIS usage rules and context information is as defined in section [7.5.2, Application Information \(AppInfo\)](#).

2919

2920 9 XML Instance Documents

2921 In order to be UN/CEFACT conformant, an instance document must be valid against
 2922 the relevant UN/CEFACT compliant XML Schema file(s). The XML instance
 2923 documents should be readable and understandable by both humans and
 2924 applications, and should enable reasonably intuitive interactions. An XPath
 2925 navigation path should describe the complete semantic understanding by
 2926 concatenating the nested elements. This navigation path should also reflect the
 2927 meaning of each dictionary entry name of a ABIE, BBIE or ASBIE.

2928 This section further describes the requirements XML Instance documents:

- 2929 • Character Encoding
- 2930 • xsi:schemaLocation
- 2931 • Empty Content
- 2932 • xsi:type

2933 9.1 Character Encoding

2934 In conformance with ISO/IETF/ITU/UNCEFACT Memorandum of Understanding
 2935 Management Group (MOUMG) Resolution 01/08 (MOU/MG01n83) as agreed to by
 2936 UN/CEFACT, all UN/CEFACT XML will be instantiated using UTF. UTF-8 is the
 2937 preferred encoding, but UTF-16 may be used where necessary to support other
 2938 languages.

[R ACE9]	All XML MUST be instantiated using UTF. UTF-8 should be used if possible, if not UTF-16 should be used.	1
----------	---	---

2939 9.2 xsi:schemaLocation

2940 The `xsi:schemaLocation` and `xsi:noNamespaceLocation` attributes are part
 2941 of the XML schema instance namespace ([http://www.w3.org/2001/XMLSchema-
 2942 instance](http://www.w3.org/2001/XMLSchema-instance)). To ensure consistency, the token `xsi` will be used to represent the XML
 2943 schema instance namespace.

[R A1B9]	The <code>xsi</code> namespace prefix MUST be used to reference the " http://www.w3.org/2001/XMLSchema-instance " namespace and anything defined by the W3C XMLSchema-instance namespace.	1
----------	---	---

2944 9.3 Empty Content

2945 Empty elements do not provide the level of assurance necessary for business
 2946 information exchanges and as such, will not be used.

2947 The only case in which elements maybe empty are in cases of where the key and
 2948 keyRef attributes are used to reference other entities in a given XML instance.

[R 9277]	The <code>xsi:nil</code> attribute MUST NOT appear in any conforming instance.	1
----------	--	---

2949 **9.4 xsi:type**

2950 The `xsi:type` attribute allows for substitution during an instantiation of a xml
2951 document. In the same way that substitution groups are not allowed, the `xsi:type`
2952 attribute is not allowed.

[R 8250]	The <code>xsi:type</code> attribute MUST NOT be used within an XML Instance.	1
----------	--	---

2953 **9.5 Supplementary Components**

2954 Code lists and identifier schemes can be defined for a business value domain either
2955 at model design time or at instance run time. When the code list or identifier scheme
2956 is defined at model design time, it is included as part of the BDT definition in the BDT
2957 XML Schema File. If a code list or identifier scheme is defined at instance run time,
2958 the supplementary component attributes are used to identify the list or scheme. To
2959 maximize interoperability and minimize human intervention required at runtime, the
2960 preferred approach is to define the scheme or list at model design time. Only in very
2961 rare circumstances should the supplementary component attributes for identifying a
2962 scheme or list be used.

[R A884]	The attributes for scheme or list supplementary components SHOULD NOT be used within an XML Instance.	1
----------	---	---

2963

2964 **Appendix A. Related Documents**

2965 The following documents provided significant levels of influence in the development
2966 of this document:

- 2967 • UN/CEFACT Core Components Technical Specification Version 3.0 ODP 6
2968 Implementation Verification
- 2969 • UN/CEFACT Core Components Technical Specification, Part 8 of the ebXML
2970 Framework Version 2.01
- 2971 • ebXML Technical Architecture Specification v1.04
- 2972 • OASIS/ebXML Registry Information Model v2.0
- 2973 • ebXML Requirements Specification v1.06
- 2974 • Information Technology - Metadata registries: Framework for the Specification
2975 and Standardization of Data Elements, International Standardization
2976 Organization, ISO 11179-1
- 2977 • Information Technology - Metadata registries: Classification of Concepts for
2978 the Identification of Domains, International Standardization Organization,
2979 ISO 11179-2
- 2980 • Information Technology - Metadata registries: Registry Metamodel,
2981 International Standardization Organization, ISO 11179-3
- 2982 • Information Technology - Metadata registries: Rules and Guidelines for the
2983 Formulation of Data Definitions, International Standardization Organization,
2984 ISO 11179-4
- 2985 • Information Technology - Metadata registries: Naming and Identification
2986 Principles for Data Elements, International Standardization Organization, ISO
2987 11179-5
- 2988 • Information Technology - Metadata registries: Framework for the Specification
2989 and Standardization of Data Elements, International Standardization
2990 Organization, ISO 11179-6

2991 **Appendix B. Overall Structure**

2992 The structure of an UN/CEFACT compliant XML schema must contain one or more
2993 of the following sections as relevant. Relevant sections must appear in the order
2994 given:

- 2995 • XML Declaration
- 2996 • Schema Module Identification and Copyright Information
- 2997 • Schema Start-Tag
- 2998 • Includes
- 2999 • Imports
- 3000 • Element
- 3001 • Root Element
- 3002 • Global Elements
- 3003 • Type Definitions

3004 **B.1 XML Declaration**

3005 A UTF-8 encoding is adopted throughout all UN/CEFACT XML Schema.

3006 **Example B-1: XML Declaration**

```
3007 <?xml version="1.0" encoding="UTF-8"?>
```

3008 **B.2 Schema Module Identification and Copyright Information**

3009 **Example B-2: Schema Module Identification and Copyright Information**

```
3010 <!-- ===== -->
3011 <!-- ===== Example - Schema Module Name ===== -->
3012 <!-- ===== -->
3013 <!--
3014 Schema agency: UN/CEFACT
3015 Schema version: 3.0
3016 Schema date: 18 November 2008
3017
3018
3019
3020
3021 Copyright (C) UN/CEFACT (2008). All Rights Reserved.
3022
3023 This document and translations of it may be copied and furnished to others, and
3024 derivative works that comment on or otherwise explain it or assist in its
3025 implementation may be prepared, copied, published and distributed, in whole or in
3026 part, without restriction of any kind, provided that the above copyright notice and
3027 this paragraph are included on all such copies and derivative works. However, this
3028 document itself may not be modified in any way, such as by removing the copyright
3029 notice or references to UN/CEFACT, except as needed for the purpose of developing
3030 UN/CEFACT specifications, in which case the procedures for copyrights defined in
3031 the UN/CEFACT Intellectual Property Rights document must be followed, or as
3032 required to translate it into languages other than English.
3033
3034 The limited permissions granted above are perpetual and will not be revoked by
3035 UN/CEFACT or its successors or assigns.
```

3036
3037
3038
3039
3040
3041
3042

```
This document and the information contained herein is provided on an "AS IS" basis
and UN/CEFACT DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT
LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE
ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR
PURPOSE.
-->
```

3043 **B.3 Schema Start-Tag**

3044 The Schema Start-Tag section of an UN/CEFACT compliant XML schema must
3045 contain one or more of the below declarations as relevant. Relevant declarations
3046 must appear in the order given:

- 3047 • Version
- 3048 • Namespaces
- 3049 • targetNamespace attribute
- 3050 • xmlns:xsd attribute
- 3051 • namespace declaration for current schema
- 3052 • namespace declaration for reusable ABIEs actually used in the schema
- 3053 • namespace declaration for unqualified data types actually used in the schema
- 3054 • namespace declaration for qualified data types actually used in the schema
- 3055 • namespace declaration for code lists actually used in the schema
- 3056 • namespace declaration for identifier schemes actually used in the schema
- 3057 • namespace declaration for CCTS
- 3058 • Form Defaults
- 3059 • elementFormDefault
- 3060 • attributeFormDefault
- 3061 • Others
- 3062 • other schema attributes with schema namespace
- 3063 • other schema attributes with non-schema namespace

3064 **Example B-3: XML Schema Start Tag**

3065
3066
3067
3068
3069
3070
3071
3072

```
<xsd:schema
targetNamespace="urn:un:unece:unefact:documentation:common:3:draft"
xmlns:rsm=" urn:un:unece:unefact:documentation:common:3:draft"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:com=" urn:un:unece:unefact:documentation:common:3:draft"
urn:un:unece:unefact:codelist:common:2001:standard:5:4217
elementFormDefault="qualified"
attributeFormDefault="unqualified">
```

3073 B.4 Includes

3074 The Include section of an UN/CEFACT compliant XML schema must contain one or
3075 more of the below declarations as relevant. Relevant declarations must appear in the
3076 order given:

- 3077 • Inclusion of the context category specific BIE XML Schema file.
- 3078 • Inclusion of the context category specific BDT XML Schema file.
- 3079 • Inclusion of the context category specific Business Code List XML Schema
3080 Files if used

3081 All schemaLocations are relative from the XML Schema File that is referencing. For
3082 the purposes of this appendix we are assuming the references are from a Root
3083 Schema File within the same namespace as the includes.

3084 Example B-4: Includes

```

3085 <!-- =====>
3086 <!-- ===== Include =====>
3087 <!-- =====>
3088 <!-- ===== Inclusion of context category BIE XML Schema File =====>
3089 <!-- =====>
3090 <xsd:include schemaLocation="BusinessInformationEntity_3p0.xsd"/>
3091 <!-- =====>
3092 <!-- ===== Inclusion of context category BDT XML Schema File =====>
3093 <!-- =====>
3094 <xsd:include schemaLocation="BusinessDataType_3p0.xsd"/>
3095 <!-- =====>
3096 <!-- Inclusion of context specific Business Code List XML Schema File = -->
3097 <!-- =====>
3098 <xsd:include schemaLocation="BusinessCodeList_1p0.xsd"/>

```

3099 B.5 Imports

3100 The Import section of an UN/CEFACT compliant XML Schema File must contain one
3101 or more of the below declarations as relevant. Relevant declarations must appear in
3102 the order given:

- 3103 • Import of Common Code List XML Schema Files actually used

3104 Example B-5: Imports

```

3105 <!-- =====>
3106 <!-- ===== Import of Code lists =====>
3107 <!-- =====>
3108 <xsd:import namespace="urn:un:unece:unefact:odelist:common:2001:standard:5:4217"
3109 schemaLocation="../../../odelist/common/2001/standard/ISO_CurrencyCode_2001.xsd"/>
3110

```

3111 B.6 Elements

3112 The root element is declared first when needed in schema that are used to support
3113 instance documents. Global elements are then declared following the root element
3114 when it is present.

3115 **Example B-6:**

```

3116 <!-- =====>
3117 <!-- ===== Element Declarations =====>
3118 <!-- =====>
3119 <!-- ===== Root element =====>
3120 <!-- =====>
3121 <xsd:element name="[ELEMENTNAME]" type="[TOKEN]:[TYPENAME]">
3122 <!-- =====>
3123 <!-- ===== Global Element Declarations =====>
3124 <!-- =====>
3125 <xsd:element name="[ELEMENTNAME]" type="[TOKEN]:[TYPENAME]">
3126 <!-- =====>
3127 <!-- =====>

```

3128 **B.7 Root element**

3129 The root element's type definition is defined immediately following the definition of
3130 the global root element to provide clear visibility of the root element's type, of which
3131 this particular schema is all about.

3132 **Example B-7:**

```

3133 <!-- =====>
3134 <!-- ===== Root element =====>
3135 <!-- =====>
3136 <xsd:element name="Invoice" type="rsm:InvoiceType">
3137 <xsd:annotation>
3138 <xsd:documentation>
3139 <ccts:UniqueID>UNM0000001</ccts:UniqueID>
3140 <ccts:VersionID>3.0</ccts:VersionID>
3141 <ccts:ObjectClassTermName>Invoice</ccts:ObjectClassTermName>
3142 <ccts:DictionaryEntryName>Invoice</ccts:DictionaryEntryName>
3143 <ccts:Definition>Document used to communicate the Invoice for a
3144 Purchase.</ccts:Definition>
3145 </xsd:documentation>
3146 </xsd:annotation>
3147 </xsd:element>

```

3148 **Example B-8: Global elements**

```

3149 <!-- =====>
3150 <!-- ===== Global element =====>
3151 <!-- =====>
3152 <xsd:element name="BuyerParty" type="bie:BuyerPartyType"/>
3153 <xsd:annotation>
3154 <xsd:documentation>
3155 <ccts:UniqueID>UNM0000002</ccts:UniqueID>
3156 <ccts:VersionID>3.0</ccts:VersionID>
3157 <ccts:ObjectClassQualifierName>Party</ccts:ObjectClassQualifierName>
3158 <ccts:ObjectClassTermName>Party</ccts:ObjectClassTermName>
3159 <ccts:DictionaryEntryName>Buyer. Party</ccts:DictionaryEntryName>
3160 <ccts:Definition>The Party that initiated the a
3161 Purchase.</ccts:Definition>
3162 </xsd:documentation>
3163 </xsd:annotation>
3164 </xsd:element>

```

3165 **B.8 Type Definitions**

3166 The definition of the BIEs used within the specific XML Schema File or by the XML
3167 Schema Files that make use of a common XML Schema File.

- 3168 • Definition of types for Basic Business Information Entities in alphabetical
3169 order, if applicable.
- 3170 • Definition of types for Aggregate Business Information Entities in alphabetical
3171 order, if applicable.

3172 **Example B-9: Type Definitions**

```

733 <!-- =====>>
734 <!-- ===== Type Definitions =====>>
735 <!-- =====>>
736 <!-- ===== Type Definition: Account type =====>>
737 <!-- =====>>
738 <xsd:complexType name="AccountType">
739   <xsd:annotation>
740     <xsd:documentation xml:lang="en">
741       <ccts:UniqueID>UN00000001</ccts:UniqueID>
742       <ccts:Acronym>ABIE</ccts:Acronym>
743       <ccts:DictionaryEntryName>Account.
744       Details</ccts:DictionaryEntryName>
745       <ccts:Version>1.0</ccts:Version>
746       <ccts:Definition>A business arrangement whereby debits and/or
747       credits arising from transactions are recorded. This could be with a bank, i.e. a
748       financial account, or a trading partner offering supplies or services 'on account',
749       i.e. a commercial account</ccts:Definition>
750       <ccts:ObjectClassTerm>Account</ccts:ObjectClassTerm>
751     </xsd:documentation>
752   </xsd:annotation>
753   <xsd:sequence>
754     <xsd:element name="ID" type="bdt:IDType" minOccurs="0"
755     maxOccurs="unbounded">
756       <xsd:annotation>
757         <xsd:documentation xml:lang="en">
758           <ccts:UniqueID>UN00000002</ccts:UniqueID>
759           <ccts:Acronym>BBIE</ccts:Acronym>
760           <ccts:DictionaryEntryName>Account.
761           Identifier</ccts:DictionaryEntryName>
762           <ccts:Version>1.0</ccts:Version>
763           <ccts:Definition>The identification of a
764           specific account.</ccts:Definition>
765           <ccts:Cardinality>0..n</ccts:Cardinality>
766         </xsd:documentation>
767       </xsd:annotation>
768     </xsd:element>
769     <xsd:element name="Status" type="bie:StatusType" minOccurs="0"
770     maxOccurs="unbounded">
771       <xsd:annotation>
772         <xsd:documentation xml:lang="en">
773           <ccts:UniqueID>UN00000003</ccts:UniqueID>
774           <ccts:Acronym>ASBIE</ccts:Acronym>
775           <ccts:DictionaryEntryName>Account.
776           Status</ccts:DictionaryEntryName>
777           <ccts:Version>1.0</ccts:Version>
778           <ccts:Definition>Status information related
779           to account details.</ccts:Definition>
780           <ccts:Cardinality>0..n</ccts:Cardinality>
781         </xsd:documentation>
782       </xsd:annotation>
783     </xsd:element>
784   </xsd:sequence>
785 </xsd:complexType>

```

```

233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

```

        <cts:Version>1.0</cts:Version>
        <cts:Definition>Country information
related to account details.</cts:Definition>
        <cts:Cardinality>0..n</cts:Cardinality>

        <cts:ObjectClassTerm>Account</cts:ObjectClassTerm>

        <cts:PropertyTerm>Country</cts:PropertyTerm>
            <cts:AssociatedObjectClassTerm>Country
            </cts:AssociatedObjectClassTerm>

        <cts:AssociationType>Aggregate</cts:AssociationType>
            </xsd:documentation>
        </xsd:annotation>
    </xsd:element>
    <xsd:element name="Person" type="bie:PersonType" minOccurs="0"
maxOccurs="unbounded">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">
                <cts:UniqueID>UN00000008</cts:UniqueID>
                <cts:Acronym>ASBIE</cts:Acronym>
                <cts:DictionaryEntryName>Account.
Person</cts:DictionaryEntryName>
                <cts:Version>1.0</cts:Version>
                <cts:Definition>Associated person
information related to account details. This can be used to identify multiple
people related to an account, for instance, the account holder.</cts:Definition>
                <cts:Cardinality>0..n</cts:Cardinality>

                <cts:ObjectClassTerm>Account</cts:ObjectClassTerm>

                <cts:PropertyTerm>Person</cts:PropertyTerm>
                    <cts:AssociatedObjectClassTerm>Person
                    </cts:AssociatedObjectClassTerm>

                <cts:AssociationType>Aggregate</cts:AssociationType>
                    </xsd:documentation>
                </xsd:annotation>
            </xsd:element>
            <xsd:element name="Organisation" type="bie:OrganisationType"
minOccurs="0" maxOccurs="unbounded">
                <xsd:annotation>
                    <xsd:documentation xml:lang="en">
                        <cts:UniqueID>UN00000009</cts:UniqueID>
                        <cts:Acronym>ASBIE</cts:Acronym>
                        <cts:DictionaryEntryName>Account.
Organisation</cts:DictionaryEntryName>
                        <cts:Version>1.0</cts:Version>
                        <cts:Definition>The associated
organisation information related to account details. This can be used to identify
multiple organisations related to this account, for instance, the account
holder.</cts:Definition>
                        <cts:Cardinality>0..n</cts:Cardinality>

                        <cts:ObjectClassTerm>Account</cts:ObjectClassTerm>

                        <cts:PropertyTerm>Organisation</cts:PropertyTerm>
                            <cts:AssociatedObjectClassTerm>Organisation
                            </cts:AssociatedObjectClassTerm>

                        <cts:AssociationType>Composition</cts:AssociationType>
                            </xsd:documentation>
                        </xsd:annotation>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>

```

3380 Example B-10: Complete Structure

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- ===== -->
<!-- ===== [SCHEMA MODULE TYPE] Schema Module ===== -->
<!-- ===== -->
<!-- -->

```


3458 **Appendix C. ATG Approved Acronyms and Abbreviations**

3459 The following constitutes a list of ATG approved acronyms and abbreviations which
3460 must be used within tag names when these words are part of the dictionary entry
3461 name:

- 3462 ABIE – Aggregate Business Information Entity
- 3463 ACC – Aggregate Core Component
- 3464 BBIE – Basic Business Information Entity
- 3465 BCC – Basic Core Component
- 3466 BDT – Business Data Type
- 3467 BIE – Business Information Entity
- 3468 CC – Core Component
- 3469 ID – Identifier
- 3470 URI – Uniform Resource Identifier
- 3471 URL – Uniform Resource Locator
- 3472 URN – Uniform Resource Name
- 3473 UUID – Universally Unique Identifier

3474 **Appendix D. Core Component XML Schema File**

3475 The Core Component XML Schema File is published as a separate file –
3476 CoreComponentType_3p0.xsd.

3477 **Appendix E. Business Data Type XML Schema File**

3478 The Business Data Type XML Schema File is published as a separate file –
3479 BusinessDataType_3p0.xsd.

3480

Appendix F. Annotation Templates

3481
3482
3483
3484
3485
3486
3487
3488
3489
3490
3491
3492
3493
3494
3495
3496
3497
3498
3499
3500
3501
3502
3503
3504
3505
3506
3507
3508
3509
3510
3511
3512
3513
3514
3515
3516
3517
3518
3519
3520
3521
3522
3523
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536
3537
3538
3539
3540
3541
3542
3543
3544
3545
3546

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- ===== -->
<!-- ===== XMLNDR Documentation Schema File ===== -->
<!-- ===== -->
<!-- ===== -->
Schema agency:      UN/CEFACT
Schema version:     3.0
Schema date:        18 November 2008

Copyright (C) UN/CEFACT (2008). All Rights Reserved.

This document and translations of it may be copied and furnished to others,
and derivative works that comment on or otherwise explain it or assist
in its implementation may be prepared, copied, published and distributed,
in whole or in part, without restriction of any kind, provided that the
above copyright notice and this paragraph are included on all such copies
and derivative works. However, this document itself may not be modified in
any way, such as by removing the copyright notice or references to
UN/CEFACT, except as needed for the purpose of developing UN/CEFACT
specifications, in which case the procedures for copyrights defined in the
UN/CEFACT Intellectual Property Rights document must be followed, or as
required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked
by UN/CEFACT or its successors or assigns.

This document and the information contained herein is provided on an "AS IS"
basis and UN/CEFACT DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING
BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL
NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR
FITNESS FOR A PARTICULAR PURPOSE.
-->
<xsd:schema xmlns:ccts="urn:un:unece:uncefact:documentation:common:3:standard"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:clm54217="urn:un:unece:uncefact:odelist:common:2001:standard:5:4217"
xmlns:bdt="urn:un:unece:uncefact:data:common:3:standard"
targetNamespace="urn:un:unece:uncefact:documentation:common:3:standard"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <!-- ===== -->
  <!-- ===== Imports ===== -->
  <!-- ===== -->
  <!-- ===== Import of Common Business Data Type ===== -->
  <!-- ===== -->
  <xsd:import namespace="urn:un:unece:uncefact:data:common:3:standard"
schemaLocation="../../../data/common/3/standard/BusinessDataType_3p0.xsd"/>
  <!-- ===== -->
  <!-- ===== Import of Code Lists ===== -->
  <!-- ===== -->
  <xsd:import
namespace="urn:un:unece:uncefact:odelist:common:2001:standard:5:4217"
schemaLocation="../../../odelist/common/2001/standard/ISO_CurrencyCode_2001.xsd"
/>
  <!-->
  <!-->
```

3540

F.1 Annotation Documentation

3541
3542
3543
3544
3545
3546

```
<xsd:group name="RootSchemaDocumentation">
  <xsd:sequence>
    <xsd:element name="UniqueID"
type="bdt:EntityUniqueIdentifierType"/>
    <xsd:element name="VersionID" type="bdt:VersionIdentifierType"/>
  
```

```

        <xsd:element name="ObjectClassQualifierName" type="bdt:NameType"
minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="ObjectClassTermName" type="bdt:NameType"/>
        <xsd:element name="DictionaryEntryName" type="bdt:NameType"/>
        <xsd:element name="Definition" type="bdt:TextType"/>
        <xsd:element name="BusinessTermName" type="bdt:NameType"
minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:group>
<xsd:group name="ABIEDocumentation">
    <xsd:sequence>
        <xsd:element name="UniqueID"
type="bdt:EntityUniqueIdentifierType"/>
        <xsd:element name="VersionID" type="bdt:VersionIdentifierType"/>
        <xsd:element name="ObjectClassQualifierName" type="bdt:NameType"
minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="ObjectClassTermName" type="bdt:NameType"/>
        <xsd:element name="Cardinality" type="bdt:NumericType"/>
        <xsd:element name="SequencingKey" type="bdt:NumericType"/>
        <xsd:element name="DictionaryEntryName" type="bdt:NameType"/>
        <xsd:element name="Definition" type="bdt:TextType"/>
        <xsd:element name="BusinessTermName" type="bdt:NameType"
minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:group>
<xsd:group name="BBIEDocumentation">
    <xsd:sequence>
        <xsd:element name="UniqueID"
type="bdt:EntityUniqueIdentifierType"/>
        <xsd:element name="VersionID" type="bdt:VersionIdentifierType"/>
        <xsd:element name="Cardinality" type="bdt:NumericType"/>
        <xsd:element name="SequencingKey" type="bdt:NumericType"/>
        <xsd:element name="DictionaryEntryName" type="bdt:NameType"/>
        <xsd:element name="Definition" type="bdt:TextType"/>
        <xsd:element name="BusinessTermName" type="bdt:NameType"
minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="PropertyTermName" type="bdt:NameType"/>
        <xsd:element name="PropertyQualifierName" type="bdt:NameType"
minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="RepresentationTermName" type="bdt:NameType"/>
    </xsd:sequence>
</xsd:group>
<xsd:group name="ASBIEDocumentation">
    <xsd:sequence>
        <xsd:element name="UniqueID"
type="bdt:EntityUniqueIdentifierType"/>
        <xsd:element name="VersionID" type="bdt:VersionIdentifierType"/>
        <xsd:element name="Cardinality" type="bdt:NumericType"/>
        <xsd:element name="SequencingKey" type="bdt:TextType"/>
        <xsd:element name="DictionaryEntryName" type="bdt:NameType"/>
        <xsd:element name="Definition" type="bdt:TextType"/>
        <xsd:element name="BusinessTermName" type="bdt:NameType"
minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="AssociationType"
type="bdt:AssociationTypeCodeType"/>
        <xsd:element name="PropertyTermName" type="bdt:NameType"/>
        <xsd:element name="PropertyQualifierName" type="bdt:NameType"
minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="RepresentationTermName" type="bdt:NameType"/>
    </xsd:sequence>
</xsd:group>
<xsd:group name="BDTDocumentation">
    <xsd:sequence>
        <xsd:element name="VersionID" type="bdt:VersionIdentifierType"/>
        <xsd:element name="UniqueID"
type="bdt:EntityUniqueIdentifierType"/>
        <xsd:element name="DictionaryEntryName" type="bdt:NameType"/>
        <xsd:element name="Definition" type="bdt:TextType"/>
        <xsd:element name="BusinessTermName" minOccurs="0"
maxOccurs="unbounded"/>
        <xsd:element name="PropertyTermName" type="bdt:NameType"/>
        <xsd:element name="DataTypeName" type="bdt:NameType"/>
        <xsd:element name="DataTypeQualifierName" type="bdt:NameType"/>
        <xsd:element name="DefaultIndicator" type="bdt:IndicatorType"/>
        <xsd:element name="DefaultValue" type="bdt:TextType"
minOccurs="0"/>

```

```

<xsd:element name="DefaultValueSource" type="bdt:TextType"
minOccurs="0"/>
<xsd:element name="SchemeOrListID" type="bdt:IDType"
minOccurs="0"/>
<xsd:element name="SchemeOrListAgencyID" type="bdt:IDType"
minOccurs="0"/>
<xsd:element name="SchemeOrListAgencyName" type="bdt:NameType"
minOccurs="0"/>
<xsd:element name="SchemeOrListModificationAllowedIndicator"
type="bdt:IndicatorType" minOccurs="0"/>
<xsd:element name="SchemeOrListName" type="bdt:NameType"
minOccurs="0"/>
<xsd:element name="SchemeOrListBusinessTermName"
type="bdt:NameType" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:group>
<xsd:group name="BDTSCDocumentation">
<xsd:sequence>
<xsd:element name="Cardinality" type="bdt:NumericType"/>
<xsd:element name="PropertyTermName" type="bdt:NameType"/>
<xsd:element name="RepresentationTermName" type="bdt:NameType"/>
<xsd:element name="PrimitiveTypeName" type="bdt:NameType"/>
<xsd:element name="DataTypeName" type="bdt:NameType"/>
<xsd:element name="DataTypeQualifierName" type="bdt:NameType"/>
<xsd:element name="DefaultIndicator" type="bdt:IndicatorType"/>
<xsd:element name="DefaultValue" type="bdt:TextType"
minOccurs="0"/>
<xsd:element name="DefaultValueSource" type="bdt:TextType"
minOccurs="0"/>
<xsd:element name="SchemeOrListID" type="bdt:IDType"
minOccurs="0"/>
<xsd:element name="SchemeOrListAgencyID" type="bdt:IDType"
minOccurs="0"/>
<xsd:element name="SchemeOrListAgencyName" type="bdt:NameType"
minOccurs="0"/>
<xsd:element name="SchemeOrListModificationAllowedIndicator"
type="bdt:IndicatorType" minOccurs="0"/>
<xsd:element name="SchemeOrListName" type="bdt:NameType"
minOccurs="0"/>
<xsd:element name="SchemeOrListBusinessTermName"
type="bdt:NameType" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:group>
<xsd:group name="CodeListDocumentation">
<xsd:sequence>
<xsd:element name="SchemeOrListID" type="bdt:IDType"/>
<xsd:element name="SchemeOrListVersionID" type="bdt:IDType"/>
<xsd:element name="SchemeOrListAgencyID" type="bdt:IDType"/>
<xsd:element name="SchemeOrListAgencyName" type="bdt:NameType"
minOccurs="0"/>
<xsd:element name="SchemeOrListModificationAllowedIndicator"
type="bdt:IndicatorType" minOccurs="0"/>
<xsd:element name="SchemeOrListName" type="bdt:NameType"/>
<xsd:element name="SchemeOrListBusinessTermName"
type="bdt:NameType" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:group>
<xsd:group name="CodeValueDocumentation">
<xsd:sequence>
<xsd:element name="SchemeOrListName" type="bdt:NameType"/>
<xsd:element name="SchemeOrListBusinessTermName"
type="bdt:NameType" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:group>
<xsd:group name="IdentifierSchemeDocumentation">
<xsd:sequence>
<xsd:element name="SchemeOrListName" type="bdt:NameType"/>
<xsd:element name="SchemeOrListBusinessTermName"
type="bdt:NameType" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:group>
<!-->
<!-->

```

3696

F.2 Annotation Application Information

3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769

```

<xsd:element name="BusinessContext">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="ContextUnit" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element
name="BusinessProcessContextCategory"
type="ccts:BusinessProcessContextCategoryType" minOccurs="0"
maxOccurs="unbounded"/>
            <xsd:element
name="BusinessProcessRoleContextCategory"
type="ccts:BusinessProcessRoleContextCategoryType" minOccurs="0"
maxOccurs="unbounded"/>
            <xsd:element
name="SupportingRoleContextCategory" type="ccts:SupportingRoleContextCategoryType"
minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element
name="IndustryClassificationContextCategory"
type="ccts:IndustryClassificationContextCategoryType" minOccurs="0"
maxOccurs="unbounded"/>
            <xsd:element
name="ProductClassificationContextCategory"
type="ccts:ProductClassificationContextCategoryType" minOccurs="0"
maxOccurs="unbounded"/>
            <xsd:element
name="GeopoliticalContextCategory" type="ccts:GeopoliticalContextCategoryType"
minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element
name="OfficialConstraintsContextCategory"
type="ccts:OfficialConstraintsContextCategoryType" minOccurs="0"
maxOccurs="unbounded"/>
            <xsd:element
name="SystemCapabilitiesContextCategory"
type="ccts:SystemCapabilitiesContextCategoryType" minOccurs="0"
maxOccurs="unbounded"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="id" type="bdt:EntityUniqueIdentifierType"/>
    <xsd:attribute name="versionID" type="bdt:VersionIdentifierType"/>
  </xsd:complexType>
</xsd:element>
<xsd:complexType name="BusinessInformationContextCategoryType">
  <xsd:sequence>
    <xsd:element name="BusinessInformationEntityID" type="bdt:IDType"
maxOccurs="unbounded"/>
    <xsd:element name="ContextExclusion" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element
name="BusinessInformationEntityID" type="bdt:IDType" maxOccurs="unbounded"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean"/>
  </xsd:complexType>
<xsd:complexType name="BusinessProcessContextCategoryType">
  <xsd:sequence>
    <xsd:element name="BusinessProcessCode" minOccurs="0"
maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:complexContent>
          <xsd:extension base="bdt:CodeType"/>
        </xsd:complexContent>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="ContextExclusion" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>

```


3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845

```

                                <xsd:element name="BusinessProcessTypeCode"
type="bdt:CodeType" maxOccurs="unbounded"/>
                                </xsd:sequence>
                                </xsd:complexType>
                                </xsd:element>
                                </xsd:sequence>
                                <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean"/>
</xsd:complexType>
<xsd:complexType name="BusinessProcessRoleContextCategoryType">
  <xsd:sequence>
    <xsd:element name="BusinessProcessRoleCode" type="bdt:CodeType"
minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="ContextExclusion" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="PartyFunctionCode"
type="bdt:CodeType" maxOccurs="unbounded"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean"/>
</xsd:complexType>
<xsd:complexType name="SupportingRoleContextCategoryType">
  <xsd:sequence>
    <xsd:element name="SupporterFunctionCode" minOccurs="0"
maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:complexContent>
          <xsd:extension base="bdt:CodeType"/>
        </xsd:complexContent>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="ContextExclusion" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="SupporterFunctionCode"
type="bdt:CodeType" maxOccurs="unbounded"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean"/>
</xsd:complexType>
<xsd:complexType name="IndustryClassificationContextCategoryType">
  <xsd:sequence>
    <xsd:element name="IndustryClassificationCode" type="bdt:CodeType"
minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="ContextExclusion" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="IndustryTypeCode"
type="bdt:CodeType" maxOccurs="unbounded"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean"/>
</xsd:complexType>
<xsd:complexType name="ProductClassificationContextCategoryType">
  <xsd:sequence>
    <xsd:element name="ProductClassificationCode" type="bdt:CodeType"
minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="ContextExclusion" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="ProductTypeCode"
type="bdt:CodeType" maxOccurs="unbounded"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean"/>
</xsd:complexType>
<xsd:complexType name="GeopoliticalContextCategoryType">
  <xsd:sequence>

```

```

33         <xsd:element name="GeopoliticalCode" minOccurs="0"
34 maxOccurs="unbounded"/>
35         <xsd:element name="ContextExclusion" minOccurs="0">
36             <xsd:complexType>
37                 <xsd:sequence>
38                     <xsd:element ref="clm54217:CurrencyCode"
39 maxOccurs="unbounded"/>
40                 </xsd:sequence>
41             </xsd:complexType>
42         </xsd:element>
43     </xsd:sequence>
44     <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean"/>
45 </xsd:complexType>
46 <xsd:complexType name="OfficialConstraintsContextCategoryType">
47     <xsd:sequence>
48         <xsd:element name="OfficialConstraintsCode" minOccurs="0"
49 maxOccurs="unbounded">
50             <xsd:complexType>
51                 <xsd:complexContent>
52                     <xsd:extension base="bdt:CodeType"/>
53                 </xsd:complexContent>
54             </xsd:complexType>
55         </xsd:element>
56         <xsd:element name="ContextExclusion" minOccurs="0">
57             <xsd:complexType>
58                 <xsd:sequence>
59                     <xsd:element name="LawTypeCode"
60 type="bdt:CodeType" maxOccurs="unbounded"/>
61                 </xsd:sequence>
62             </xsd:complexType>
63         </xsd:element>
64     </xsd:sequence>
65     <xsd:attribute name="inAllContextsListIndicator" type="xsd:boolean"/>
66 </xsd:complexType>
67 <xsd:complexType name="SystemCapabilitiesContextCategoryType">
68     <xsd:sequence>
69         <xsd:element name="SystemCapabilitiesID" type="bdt:IDType"
70 minOccurs="0" maxOccurs="unbounded"/>
71         <xsd:element name="ContextExclusion" minOccurs="0">
72             <xsd:complexType>
73                 <xsd:sequence>
74                     <xsd:element name="SoftwareSolutionID"
75 type="bdt:IDType" maxOccurs="unbounded"/>
76                 </xsd:sequence>
77             </xsd:complexType>
78         </xsd:element>
79     </xsd:sequence>
80     <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean"/>
81 </xsd:complexType>
82 <xsd:element name="UsageRule" type="ccts:UsageRuleType"/>
83 <xsd:complexType name="UsageRuleType">
84     <xsd:sequence>
85         <xsd:element name="UniqueID"
86 type="bdt:EntityUniqueIdentifierType"/>
87         <xsd:element name="Constraint" type="bdt:TextType"/>
88         <xsd:element name="ConstraintTypeCode" type="bdt:CodeType"/>
89         <xsd:element name="ConditionTypeCode"
90 type="bdt:ConditionTypeCodeType"/>
91         <xsd:element name="Name" type="bdt:NameType" minOccurs="0"/>
92         <xsd:element name="BusinessTerm" type="bdt:TextType" minOccurs="0"
93 maxOccurs="unbounded"/>
94     </xsd:sequence>
95 </xsd:complexType>
96 </xsd:schema>

```

3911 **Appendix G. Mapping of CCTS Representation Terms to**
 3912 **CCT and BDT Data Types**

3913 The following table represents the mapping between the representation terms as
 3914 defined in CCTS and their equivalent data types as declared in the CCT schema
 3915 module and the BDT schema module.

Representation Term	Data Type for CCT	Data Type for BDT
Amount	xsd:decimal	xsd:decimal
Binary Object	xsd:base64Binary	xsd:base64Binary
Graphic		xsd:base64Binary
Sound		xsd:base64Binary
Video		xsd:base64Binary
Code	xsd:token	xsd:token
Date Time	xsd:string	xsd:dateTime
Date		xsd:date
Time		xsd:time
Identifier	xsd:token	xsd:token
Indicator	xsd:string	xsd:boolean
Measure	xsd:decimal	xsd:decimal
Value		xsd:decimal
Percent		xsd:decimal
Rate		xsd:decimal

Numeric	xsd:string	xsd:decimal
Quantity	xsd:decimal	xsd:decimal
Text	xsd:string	xsd:string
Name		xsd:string

3916

3917 Appendix H. Use Cases for Code Lists

3918 Code lists provide mechanisms for conveying data in a consistent fashion where all
 3919 parties to the information – originator, sender, receiver, processor – fully understand
 3920 the purpose, use, and meaning of the data. This specification support flexible use of
 3921 code lists. This appendix details the mechanisms for this use.

3922 The five alternative uses for code lists are:

- 3923 • Referencing a predefined standard code list, such as ISO 4217 currency
 3924 codes as a supplementary component in an BDT, such as bdt:AmountType.
- 3925 • Referencing any code list, standard or proprietary, by providing the required
 3926 identification as attributes in the BDT bdt:CodeType.
- 3927 • Referencing a predefined code list by declaring a specific BDT.
- 3928 • Choosing or combining values from several code lists.
- 3929 • Restricting the set of allowed code values from an established code list.

3930 Example H-1 is a code snippet from an XML Schema File that uses each of these.

3931 Example H-1: Code Use Example Schema

```

3932 <xsd:schema xmlns:ordman="un:unece:cefact:data:ordermanagement:1:draft"
3933 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
3934 targetNamespace="urn:un:unece:cefact:data:ordermanagement:1:draft"
3935 elementFormDefault="qualified" attributeFormDefault="unqualified">
3936 <!-- ===== Include ===== -->
3937 <xsd:include
3938 schemaLocation="http://www.unece.org/unecefact/data/ordermanagement/1/draft/Business
3939 InformationEntity_lp3p6.xsd"/>
3940 <xsd:include
3941 schemaLocation="http://www.unece.org/unecefact/data/ordermanagement/1/draft/Business
3942 DataType_lp3p6.xsd"/>
3943
3944 <!-- Root element -->
3945 <xsd:element name="Invoice" type="ordman:InvoiceType"/>
3946 <!-- Messase type declaration -->
3947 <xsd:complexType name="InvoiceType">
3948 <xsd:sequence>
3949 <xsd:element name="Product" type="ordman:ProductType"/>
3950 <xsd:element name="CustomerParty" type="ordman:PartyType"/>
3951 </xsd:sequence>
3952 </xsd:complexType>
3953 <!-- The below type declaration would normally appear in a separate schema module
3954 for all reusable components (ABIE) but is included here for completeness -->
3955 <xsd:complexType name="ProductType">
3956 <xsd:sequence>
3957 <xsd:element name="TotalAmount" type="ordman:AmountDecimalType"/>
3958 <xsd:element name="TaxCurrencyCode" type="ordman:CodeType"/>
3959 <xsd:element name="ChangeCurrencyCode"
3960 type="ordman:CurrencyCodeType"/>
3961 <xsd:element name="CalculationCurrencyCode"
3962 type="ordman:CalculationCurrencyCodeType"/>
3963 <xsd:element name="RestrictedCurrencyCode"
3964 type="ordman:RestrictedCurrencyCodeType"/>
3965 </xsd:sequence>
3966 </xsd:complexType>
3967 </xsd:schema>
  
```

3968 This schema includes:

- 3969 • The BDT XML Schema File defined for the given context category (business
 3970 process value which is order management).

3971 ○ The two specific data types CurrencyCodeType and
3972 CalculationCurrencyCodeType are defined as Business Code List that
3973 are included through the BDT XML Schema File.

3974 • The BIE XML Schema File defined for the given context category.

3975 The **xsd:complexType** named "ProductType" includes five local elements. Each of
3976 these elements represents one of the five different code list options.

3977 **H.1 Referencing a Common Code List as a Supplementary** 3978 **Component in a Business Data Types**

3979 In Example H-1, the element TotalAmount is declared as shown in Example H-2.

3980 **Example H-2: Declaration of TotalAmount Element**

3981

```
<xsd:element name="TotalAmount" type="ordman:AmountDecimalclm5ISO42173AType"/>
```

3982 As shown in the element declaration, TotalAmount is of the generic CCT
3983 AmountType that is implemented in the the context category using the primitive
3984 decimal and the CCL ISO code list 42173A resulting in the BDT
3985 AmountDecimalclm5ISO42173AType which has been defined in the BDT XML
3986 Schema File. The AmountDecimalclm5ISO42173A Type declaration is as show in
3987 Example H-3.

3988 **Example H-3: Declaration of AmountDecimal DataTypes in the BDT**

```
3989           <xsd:schema targetNamespace="urn:un:unece:uncefact:data:ordermanagement:1:draft"
3990           xmlns:clm54217="urn:un:unece:uncefact:codelist:common:1:draft:5:4217:2001" ...
3991           elementFormDefault="qualified" attributeFormDefault="unqualified">
3992           <!-- ===== Imports ===== -->
3993           <!-- ===== Imports of Code Lists ===== -->
3994           <!-- ===== Imports of Code Lists ===== -->
3995           <!-- ===== Imports of Code Lists ===== -->
3996           <!-- ===== Imports of Code Lists ===== -->
3997           <xsd:import namespace="urn:un:unece:uncefact:codelist:common:1:draft:5:4217:2001"
3998           schemaLocation="
3999           http://www.unece.org/unecefact/codelist/common/1/draft/5/4217_2001_.xsd "/>
4000           <!-- ===== Type Definitions ===== -->
4001           <!-- ===== Type Definitions ===== -->
4002           <!-- ===== Amount Decimal. Type ===== -->
4003           <!-- ===== Amount Decimal. Type ===== -->
4004           -->
4005           <!-- ===== -->
4006           <xsd:complexType name="AmountDecimalclm5ISO42173AType">
4007           <xsd:simpleContent>
4008           <xsd:extension base="xsd:decimal">
4009           <xsd:attribute name="currencyCode"
4010           type="clm5ISO42173A:ISO3AlphaCurrencyCodeContentType" use="optional"/>
4011           </xsd:extension>
4012           </xsd:simpleContent>
4013           </xsd:complexType>
```

4015 The AmountType has attributes declared that represent the supplementary
4016 components defined in CCTS for this data type. These attributes include
4017 currencyCode for the supplementary component of Amount. Currency. Code. This
4018 currencyCode attribute is declared to be of the **xsd:simpleType**
4019 **clm5ISO42173A:ISO3AlphaCurrencyCodeContentType**. The
4020 **clm5ISO42173A:ISO3AlphaCurrencyCodeContentType** has been declared in

4021 the code list schema module for ISO Currency Codes, and the allowed code values
 4022 for the `currencyCode` attribute have been defined as enumeration facets in the
 4023 `clm5ISO42173A:ISO3AlphaCurrencyCodeContentType` type definition.

4024 An extract of the CCL XML Schema File for the ISO Currency Codes is shown in H-
 4025 4.

4026 **Example H-4: Declaration of a Currency Code List**

```

4027 <!-- ===== -->
4028 <!-- ===== Root Element Declarations ===== -->
4029 <!-- ===== -->
4030 <xsd:element name="CurrencyCode" type="clm54217:CurrencyCodeContentType"/>
4031 <!-- ===== -->
4032 <!-- ===== Type Definitions ===== -->
4033 <!-- ===== -->
4034 <!-- ===== Code List Type Definition: Currency Codes ===== -->
4035 <!-- ===== -->
4036 <xsd:simpleType name="CurrencyCodeContentType">
4037   <xsd:restriction base="xsd:token">
4038     <xsd:enumeration value="AED">
4039       <xsd:annotation>
4040         <xsd:documentation>
4041           ... see the section for Code Value Documentation ...
4042         </xsd:documentation>
4043       </xsd:annotation>
4044     </xsd:enumeration>
4045     <xsd:enumeration value="AFN">
4046       <xsd:annotation>
4047         <xsd:documentation>
4048           ... see the section for Code Value Documentation ...
4049         </xsd:documentation>
4050       </xsd:annotation>
4051     </xsd:enumeration>
4052   </xsd:restriction>
4053 </xsd:simpleType>
4054 </xsd:schema>

```

4055 The `currencyCode` attribute has a fixed value of ISO 4217 Currency Code as defined
 4056 in CCTS. Only code values from this code list are allowed in a CEFACT conformant
 4057 instance documents. The resulting instance documents conveyance currency code
 4058 values are represented as:

```

4059 <TotalAmount currencyCode="AED">3.14</TotalAmount>

```

4060 [Note:]

4061 When using this option no information about the code list used is carried in the
 4062 instance document as this is already defined in the XML Schema.

4063 H.2 Referencing any code list using BDT CodeType

4064 The second element in our example message – `TaxCurrencyCode` – is of the BDT
 4065 `bdt:CodeType`.

```

4066 <xsd:element name="TaxCurrencyCode" type="bdt:CodeType"/>

```

4067 This `bdt:CodeType` data type includes a number of supplementary components
 4068 required in order to uniquely identify the code list to be used for validation.

4069 The **bdt:CodeType** is declared in the BDT XML Schema File as shown in Figure H-
4070 5

4071 **Example H-5: Declaration of a Code Type in the BDT XML Schema File**

```
4072 <xsd:complexType name="CodeType">  
4073   <xsd:simpleContent>  
4074     <xsd:extension base="xsd:token">  
4075       <xsd:attribute name="listID" type="xsd:token" use="optional"/>  
4076       <xsd:attribute name="listAgencyID" type="xsd:token" use="optional"/>  
4077       <xsd:attribute name="listVersionID" type="xsd:token" use="optional"/>  
4078     </xsd:extension>  
4079   </xsd:simpleContent>  
4080 </xsd:complexType>
```

4081 When the **bdt:CodeType** is used, either the listID indicates the Code List
4082 identification. The listAgencyID is the Agency identification that made the code list
4083 available. The listVersionID indicates the version of the code list.

4084 The association to the specific values must be made at runtime. In an instance
4085 document this element could be represented as:

```
4086 <TaxCurrencyCode listID="ISO 4217" listVersionID="2001"  
4087 listAgencyID="5">AED</TaxCurrencyCode>
```

4088 It should be noted that when applying this option, validation of code values in the
4089 instance document will not be done by the XML parser.

4090 **H.3 Referencing a Common Code List in a BDT**

4091 The third element in our example message ChangeCurrencyCode is based on the
4092 business data type **bdt:CurrencyCodeType**.

```
4093 <xsd:element name="ChangeCurrencyCode" type="bdt:CurrencyCodeclm54217-A Type"/>
```

4094 The **bdt:CurrencyCodeType** would be defined in the BDT XML Schema File as:

```
4095 <xsd:simpleType name="CurrencyCodeclm54217-AType">  
4096   <xsd:restriction base="clm54217-A:CurrencyCodeContentType"/>  
4097 </xsd:simpleType>
```

4098 This means that the value of the ChangeCurrencyCode element can only have code
4099 values from the identified ISO 4217 code list. In an instance document this element
4100 would be represented as:

```
4101 <ChangeCurrencyCode>AED</ChangeCurrencyCode>
```


4102 [Note:]

4103 When using this option no information about the code list used is carried in the
4104 instance document as this is already defined in the XML Schema.

4105 H.4 Choosing or Combining Values from Several Code Lists

4106 The fourth option is to combine values from diverse code lists by using the
4107 `xsd:union` element.

4108 The `xsd:union` code list approach enables multiple code lists to be used for a
4109 single element or attribute. The element declaration in the XML Schema, the element
4110 `CalculationCurrencyCode` is based on the namespace specific BCL type
4111 defined in the context category specific namespace BCL XML Schema File where
4112 the `ordman:CalculationCurrencyCodeclm54217-Nclm54217-AType` is
4113 declared.

```
4114 <xsd:element name="CalculationCurrencyCode"  
4115 type="ordman:CalculationCurrencyCodeType"/>
```

4116 The `ordman:CalculationCurrencyCodeclm54217-Nclm54217-AType` is
4117 defined in the BCL XML Schema File with in the context category namespace for
4118 Order Management, using an `xsd:union` element that unions the code lists
4119 together.

```
4120 <xsd:simpleType name="CalculationCurrencyCodeclm54217-Nclm54217-AType">  
4121 <xsd:union memberTypes="clm54217-N:CurrencyCodeContentType  
4122 clm54217-A:CurrencyCodeContentType"/>  
4123 </xsd:simpleType>
```

4124 This allows values to come from either the `clm54217-`
4125 `N:CurrencyCodeContentType` or from the `clm54217-`
4126 `A:CurrencyCodeContentType`. The CCL XML Schema File for `clm54217-`
4127 `A:CurrencyCodeContentType` is the same as the one used earlier in this
4128 Appendix. The CCL XML Schema File for `clm54217-`
4129 `N:CurrencyCodeContentType` is the same as the one used earlier in this
4130 Appendix.

4131 The `xsd:union` allows the use of code values from different pre-defined code lists
4132 in instance documents. The code lists must be imported once in the BCL XML
4133 Schema File. The specific code list will be represented by the namespace prefixes
4134 (`clm54217-A` or `clm54217-N`), the element in the instance document will not have
4135 the specific code list tokens conveyed as the first part of the element name. The
4136 recipient of the instance does not know unambiguously which code list each code
4137 value is defined. This is because a reference to the specific code lists comes from
4138 different Code List XML Schema Files, in this case, `clm54217-N` and `clm54217-A`.

4139 In an instance document this element could be represented as:

```
4140 <Invoice >  
4141 ...  
4142 <CalculationCurrencyCode>840</CalculationCurrencyCode>  
4143 ...
```

4144 `</Invoice>`

4145 The advantage of the `xsd:union` is that attributes can also make use of these code
4146 lists.

4147 [Note:]

4148 When using this option no information about the code list used is carried in the
4149 instance document as this is already defined in the XML Schema.

4150 **H.5 Restricting the Allowed Code Values**

4151 This option is used when it is desired to reduce the number of allowed code values
4152 from an existing code list. For example, a trading partner community may only
4153 recognize certain code values from the ISO 4217 Currency Code list. To accomplish
4154 this, create a BCL XML Schema File within the specific context category namespace
4155 of the XML Schema Files that use it. This BCL XML Schema File simply contains the
4156 restricted set of values used by the context category.

4157 This is accomplished by importing the CCL XML Schema File and using
4158 `xsd:restriction` to restrict the values to the set of values required. For more
4159 please section [8.5.3.4 Type Definitions](#).

4160 **Appendix I. Alternative Business Message Syntax**

4161 **Binding**

4162 UN/CEFACT will create the XML syntax binding of its CCTS conformant BIE data
4163 models directly from the associations and hierarchies expressed in the Business
4164 Message Template for each business message exchange. This approach is based
4165 on traditional nesting of all components of the data model.

4166 The XML Schema Specification also supports an alternative to nesting. This
4167 alternative, using schema identity constraints (`xsd:key/xsd:unique/xsd:keyRef`),
4168 enables referencing and reuse of a given element in instance documents.
4169 UN/CEFACT is currently evaluating this alternative for future use to include a method
4170 for application at the data model level. In anticipation that the data model issues will
4171 be resolved, UN/CEFACT has already developed a set of rules for its XML
4172 implementation. These rules and the supporting narrative are presented in this
4173 Appendix. Organizations using this Alternative Method will still be considered
4174 conformant to this specification, if they adhere to all other conformance requirements
4175 and use the rules defined in this Appendix.

4176 **I.1 XML Schema Architecture**

4177 **I.1.1 Message Assembly Considerations**

4178 If referencing between specific ABIE's is required in the scope of the root Message
4179 Assembly (MA) or of a lower level ABIE, the Business Message Template must
4180 specify the list of ABIE's that are implemented as referenced rather than nested
4181 properties. This will allow the identity constraints to be generated in the message
4182 schema.

4183 **I.1.2. Requirements for XML Element Referencing**

4184 **I.1.2.1 Implementation of Aggregations – Nesting or Referencing**

4185 Since aggregations relate ABIEs that have independent life cycles, the same
4186 instance of a particular ABIE may be referenced more than once within a message.
4187 The ClaimNotify message shown below, taken from the Insurance Industry, illustrate
4188 this.

4189 In Example K-1 and Example K-2 the same Person 'John Smith' can play the role of
4190 "Insured" in the Policy ABIE and the role of "Claimant" in the Claim ABIE. In order to
4191 reduce redundancy in the message, it is possible to use XML referencing to relate
4192 one Person instance to the Policy and Claim instances as an alternate method to
4193 nesting information about Person within Policy and Claim.

4194

4195 In general, when the level of reuse of an instance ABIE in a message is significant it
4196 becomes adequate to use XML referencing for the purpose of removing redundancy
4197 from the message and increasing information integrity.

4198

4199 **Example I-1: XML Instance of ClaimNotify using nesting**

```

4200 <ClaimNotify>
4201 .....
4202 <Claim>
4203   <ClaimantParty>
4204     <Name>John Smith</Name>
4205   </ClaimantParty>
4206 </Claim>
4207 .....
4208 <Policy>
4209   <InsuredParty>
4210     <Name>John Smith</Name>
4211   </InsuredParty>
4212 </Policy>
4213 </ClaimNotify>

```

4214 **Example I-2: XML Instance of ClaimNotify using referencing**

```

4215 <ClaimNotify>
4216 .....
4217 <Party key="P1">
4218   <Name>John Smith</Name>
4219 </Party>
4220 <Claim>
4221   <ClaimantParty partyReference="P1"/>
4222 </Claim>
4223 .....
4224 <Policy>
4225   <InsuredParty partyReference="P1"/>
4226 </Policy>
4227 .....
4228 </ClaimNotify>

```

4229 **1.1.2.2 Other Usages of XML Referencing**

4230 Another requirement for XML element referencing is *Dynamic Referencing*.

4231 The requirement is that any element composing a message is potentially the target
 4232 of a reference for the purpose of building dynamic relationships between elements
 4233 within the message. An important use case is identification of faulty elements for
 4234 error reporting.

4235 **1.1.2.3 Schema Validation Requirements for XML References**4236 **1.1.2.3.1 Structural References between Aggregated ABIEs**

4237 For structural references between ABIEs, the level of validation performed by the
 4238 XML Schema definition of a message should be as strong as if the referenced
 4239 element would have been defined as a nested child of the element that references it.
 4240 Thus, the schema must strictly enforce identity constraints, i.e.:

- 4241 1. Check uniqueness of the identifiers of the referenced elements
- 4242 2. Check that the references match the identifiers of the corresponding
 4243 referenced elements.

4244 Due to its more robust identity constraints, this specification mandates **key/keyRef**
 4245 as the XML referencing technique to be used instead of **Id/IdRef**. See sections
 4246 [7.1.5 Constraints on Schema Construction](#), [I.2.1.1 Constraints on Schema](#)
 4247 [Construction](#) and [I.3.1.1 Declaration of the Referencing Constraints](#).

4248 Referencing between ABIEs occur in the boundaries of a particular 'scope element'
4249 in the XML document. The scope element is the container of all the elements that
4250 can be involved in the identity constraints. These identity constraints act as follows:

- 4251 • The uniqueness (xsd:unique) or key (xsd:key) constraints define the keys and
4252 enforce that a value is unique within the scope element.

4253 The key reference (xsd:keyRef) constraints define the key references and enforce
4254 that a value corresponds to a value represented by a uniqueness (xsd:unique) or key
4255 (xsd:key) constraint.

4256 Most often the scope element will be the message root element but it can also be
4257 another element lower in the hierarchy. The XML Schema language requires that the
4258 key-keyref constraints be defined within a scope element.

4259 **I.1.2.3.2 Dynamic References**

4260 For dynamic references schema validation is not required. Since dynamic
4261 referencing is only used for ancillary purposes, it is not deemed essential to enforce
4262 uniqueness of identifiers in the schema when they are not involved in structural
4263 referencing. Uniqueness of such identifiers should be granted by use of adequate
4264 algorithms for the generation of the identifiers. This will avoid unnecessary
4265 complexity of the identity constraints.

4266 **I.2 General XML Schema Language Conventions**

4267 **I.2.1 Overall XML Schema Structure and Rules**

4268 **I.2.1.1 Constraints on Schema Construction**

4269 The XML Schema **xsd:key**, **xsd:keyref** or **xsd:unique** identity constraints
4270 have the following characteristics that make them preferable to the
4271 **xsd:ID/xsd:IDREF** technique.

- 4272 • The keys and relationships between objects are strongly typed. They are
4273 declared explicitly in the schema. Each relationship is distinctly defined and
4274 specifies exactly which object has a key, what is the key, which other objects
4275 can link to this object and through which element or attribute. You can prevent
4276 an object to point to an arbitrary object that has an identifier attribute, as it is
4277 the case with the ID/IDREF method.

- 4278 • The scope of key uniqueness is precisely defined among one or several
4279 objects within a particular instance of an XML element. It is not more
4280 necessary to ensure uniqueness of id attributes across the whole XML
4281 document.

- 4282 • The elements or attributes used as keys or key references can be of any data
4283 type, not only ID or IDRef (implying the NMTOKEN format). This allows any
4284 element or attribute to be used for linking.

4285 The following principles are taken into account for the implementation of schema
4286 identity constraints:

- 4287 1. Identifiers and references used in schema identity constraints will be
 4288 attributes. This has the advantage that the data element content of the XML
 4289 complex types derived from ABIEs is kept unchanged
- 4290 2. For maximum element and type reuse and to stay away from forward
 4291 compatibility problems, attributes used as identifiers or references will be
 4292 optional. This means that no key (`xsd:key`) constraints should be defined on
 4293 identifiers, which would make the identifiers mandatory in the context of a
 4294 message; only uniqueness (`xsd:unique`) constraints must be used.
- 4295 3. Only the ABIEs that are part of a logical aggregation implemented by XML
 4296 referencing will be subject to explicit schema identity constraints. For all other
 4297 ABIEs - which may only be involved in dynamic references - uniqueness of
 4298 identifiers should be granted by use of adequate algorithms for the generation
 4299 of the identifiers.

[R 8E89]	Schema identity constraints MUST be used to implement references between elements when they represent ABIE's that are linked by an association, whose AggregationKind property is 'shared'.	1
[R 8103]	The uniqueness (<code>xsd:unique</code>) constraint MUST be used rather than the key (<code>xsd:key</code>) constraint to define the keys and enforce that their values are unique within their scope of application.	1

4300 **I.2.2 Attribute and Element Declarations**

4301 **I.2.2.1 Attributes**

4302 Attributes are only used in two cases:

- 4303
- To convey the supplementary components of BDTs;
 - To serve as identifiers and references when two elements need to be related to one another via schema identify constraints (`xsd:key/xsd:keyref`).
 - To serve as identifiers for dynamic referencing.
- 4304
- 4305
- 4306

[R 8EE7]	Identifiers used in schema identify constraints or for dynamic referencing MUST be declared as attributes.	1
[R 991C]	User defined attributes MUST only be used for Supplementary Components or to serve as identifiers in identity constraints. Modification to Rule [R AFEE].	1

4307 **I.2.2.2 Elements**

[R A577]	Empty elements MUST NOT be used, except when their definition includes an identifier attribute that serves to reference another element via schema identity constraints. Modification to Rule [R B8B6].	1
----------	---	---

4308 **I.3 XML Schema Files**4309 **I.3.1 Root XML Schema Files**4310 **I.3.1.1 Declaration of the Referencing Constraints**

4311 Referencing between ABIEs occurs within the limits defined by a particular 'scope'
4312 element in the XML document tree.

4313

4314 The scope element is the container of all the elements that can be involved in the
4315 identity constraints. The schema language requires that the identity constraints be
4316 contained in the schema declaration of the scope element.

4317 Most often the scope element will be the message root element, but it can also be
4318 another element lower in the hierarchy.

4319 The identifier attribute of each ABIE that is part of a logical aggregation implemented
4320 by XML referencing will be subject to a uniqueness (**xsd:unique**) constraint
4321 defined in the scope element. The name of the **xsd:unique** constraint must be
4322 unique in the schema.

4323 The uniqueness (**xsd:unique**) constraints define the keys and enforce that a value
4324 is unique within the scope element.

4325 The key reference (**xsd:keyRef**) constraints define the key references and enforce
4326 that a value corresponds to a value represented by a uniqueness (**xsd:unique**)
4327 constraint.

[R BA43]	Each ABIE element that is a scope element of a set of XML Schema identity constraints MUST contain one or more xsd:unique constraint declarations.	1
[R 88DB]	Each ABIE that is the target of a reference under a scope element MUST be the object of a xsd:unique constraint declaration via a xsd:selector/@xpath component.	1
[R B40C]	The name of an xsd:unique constraint MUST be constructed as follows: " <Scope element><Referenced Element>Key " Where: <ul style="list-style-type: none"> • Scope element – is the name of the scope element. • Referenced Element – is the element name being referenced within the scope element. 	1

4328 This declaration will guarantee uniqueness of the identifier attribute values across all
4329 referenced elements of the same name, in the given scope.

4330

[Note:]

4331

The value of `xsd:selector/@xpath` identifies instances of one element in one namespace (by default the namespace of the XML Schema File in which the `xsd:selector` is declared.).

4332

4333

4334

In Example I-3 the declaration under the message root element will guarantee uniqueness of the `@key` attribute values across all `bie:Party` elements, in the scope of the `rsm:ClaimNotify` message.

4335

4336

4337

Example I-3: Unique Declaration

4338

4339

4340

4341

```
<xsd:unique name="ClaimNotifyPartyKey">
  <xsd:selector xpath="bie:Party"/>
  <xsd:field xpath="@key"/>
</xsd:unique>
```

4342

For each referenced ABIE used in a given scope, corresponding key reference (`xsd:keyRef`) declarations must be made. Naming conventions used for key reference attributes, as exposed in I.3.2.2, are such that only one key reference (`xsd:keyRef`) declaration is needed for all the elements where the key reference attribute appears.

4343

4344

4345

4346

[R AC2D]	For each referenced element in a given scope one <code>xsd:keyref</code> constraint involving the reference attribute that point to the referenced element MUST be declared in the XML Schema, under the scope element.	1
[R 9BE8]	The <code>xsd:keyref/xsd:selector/@xpath</code> component must be such that it selects all the elements where the key reference attribute may occur.	1
[R 858D]	<p>The name of an <code>xsd:keyref</code> constraint MUST be constructed as follows: “<Scope Element ><Referenced Element>Reference”</p> <p>Where:</p> <ul style="list-style-type: none"> • Scope Element – is the name of the scope element. • Referenced Element – is the element name being referenced within the scope element. 	1

4347

In Example I-4 the declaration under the message root element will enforce referencing between all the elements that have the `@PartyReference` attribute and instances of `bie:Party`, in the scope of the `rsm:ClaimNotify` message.

4348

4349

4350

Example I-4: Key Reference Declaration

4351

4352

4353

4354

```
<xsd:keyref name="ClaimNotifyPartyReference" refer="ClaimNotifyPartyKey">
  <xsd:selector xpath=".*"/>
  <xsd:field xpath="@partyReference"/>
</xsd:keyref>
```


4355 [Note:]
 4356 The value of `xsd:selector/@xpath` allows for any element in any namespace to
 4357 be the parent element of the reference attribute in the `xsd:keyref` constraint.

4358 Dynamic referencing does not require the schema to enforce uniqueness of `@key`
 4359 attributes when they are not involved in structural referencing. This will avoid
 4360 unnecessary complexity of the identity constraints.

[R 886A]	Uniqueness of <code>@key</code> attributes that are not involved in structural referencing MUST NOT be enforced by the schema via identity constraints. Uniqueness of <code>@key</code> attributes should be assured by use of adequate algorithms for the generation of the identifiers (e.g. UUIDs).	1
----------	--	---

4361 **I.3.2 Business Information Entities XML Schema Files**

4362 **I.3.2.1 Type Definitions**

4363 Every aggregate business information entity (ABIE) `xsd:complexType` definition
 4364 will include an optional identifier attribute that may be used for both dynamic and
 4365 structural referencing. It will be defined as a local attribute named “key” to avoid any
 4366 confusion with legacy XML ID attributes.

[R 8EA2]	Every aggregate business information entity (ABIE) <code>xsd:complexType</code> definition MUST contain an optional, locally defined, “key” attribute that MAY be used as the complex element identifier in the XML document where it appears.	1
[R 92C0]	“key” MUST be a reserved attribute name.	1
[R 8A37]	Every “key” local attribute declaration MUST be of the type <code>xsd:token</code> .	1

4367 **I.3.2.2 Element Declarations and References**

4368 **I.3.2.2.1 ASBIE Elements**

4369 For each ASBIE who’s `ccts:AggregationKind` value=`Shared`, there are two
 4370 mutually exclusive cases, one of which needs to be selected on the base of the
 4371 applicable Message Assembly definition.

- 4372 • The globally declared element for the associated ABIE is included in the
 4373 content model of the parent ABIE as a nested complex property.
- 4374 • An equivalent referencing element pointing to the associated ABIE is included
 4375 in the content model of the parent ABIE.

4376 See section [5.4 Reusability Schema](#) and [I.1.1 Message Assembly Considerations](#)
 4377 earlier this specification.

[R B78E]	Every ASBIE whose <code>ccts:AggregationKind</code> value= <code>Shared</code> , and where the association must be implemented as a referenced property, an equivalent referencing element pointing to the associated ABIE MUST be locally declared.	1
[R B173]	For each equivalent referencing element an <code>xsd:complexType</code> MUST be declared. Its structure will be an empty element with a local attribute.	1
[R AEDD]	The equivalent referencing element MUST have a name composed of the ASBIE property term and property qualifier term(s)) and the object term and qualifier term(s) of the associated ABIE.	1
[R B3E5]	When there is no ASBIE property term the generic property term "Referred" followed by the name of the associated ABIE MUST be used as a naming convention to distinguish this element from the ABIE element.	1
[R B523]	The name of the local attribute that is part of the empty element MUST be composed of the object class term and object qualifier term(s) of the ABIE being referenced, followed by the suffix 'Reference'.	1
[R 8B0E]	The name of the <code>xsd:complexType</code> representing the equivalent referencing element MUST be composed of the object class term and object qualifier term(s) of the ABIE being referenced, followed by the suffix 'ReferenceType'.	1
[R B7D6]	Each equivalent referencing element MUST be declared using the <code>xsd:complexType</code> that relates to the ABIE being referenced.	1

4378

4379 Example I-5 shows the schema definition of an ASBIE specified as a referencing
4380 element.

4381 **Example I-5: Element and type definition of an ASBIE, specified as a referencing element**

4382
4383
4384
4385

```
<xsd:complexType name="PartyReferenceType">
  <xsd:attribute name="partyReference" type="xsd:token"/>
</xsd:complexType>
<xsd:element name="ClaimantParty" type="PartyReferenceType"/>
```

4386

Appendix J. Naming and Design Rules List

[R B998]	<p>Conformance SHALL be determined through adherence to the content of the normative sections and rules. Furthermore each rule is categorized to indicate the intended audience for the rule by the following:</p> <table border="1" data-bbox="430 436 1315 1522"> <thead> <tr> <th colspan="2" data-bbox="430 436 1315 499">Rule Categorization</th> </tr> <tr> <th data-bbox="430 499 462 573">ID</th> <th data-bbox="462 499 1315 573">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="430 573 462 716">1</td> <td data-bbox="462 573 1315 716">Rules which must not be violated by individual organizations else conformance and interoperability is lost – such as named types.</td> </tr> <tr> <td data-bbox="430 716 462 858">2</td> <td data-bbox="462 716 1315 858">Rules which may be modified by individual organizations while still conformant to the NDR structure – such as namespace string contents and namespace tokens.</td> </tr> <tr> <td data-bbox="430 858 462 1031">3</td> <td data-bbox="462 858 1315 1031">Rules which may be modified by individual organizations while still conformant to agreed upon data models – such as the use of global or local element declarations. (Changes to the XML Schema Architecture.)</td> </tr> <tr> <td data-bbox="430 1031 462 1182">4</td> <td data-bbox="462 1031 1315 1182">Rules that if violated lose conformance with the UN/CEFACT data/process model – such as <code>xsd:redefine</code>, <code>xsd:any</code>, and <code>xsd:substitutionGroups</code>.</td> </tr> <tr> <td data-bbox="430 1182 462 1318">5</td> <td data-bbox="462 1182 1315 1318">Rules that relate to extension that are not used by UN/CEFACT and have specific restrictions on their use by other than UN/CEFACT organizations.</td> </tr> <tr> <td data-bbox="430 1318 462 1423">6</td> <td data-bbox="462 1318 1315 1423">Rules that relate to extension that are determined by specific organizations.</td> </tr> <tr> <td data-bbox="430 1423 462 1522">7</td> <td data-bbox="462 1423 1315 1522">Rules that can be modified while not changing instance validation capability.</td> </tr> </tbody> </table>	Rule Categorization		ID	Description	1	Rules which must not be violated by individual organizations else conformance and interoperability is lost – such as named types.	2	Rules which may be modified by individual organizations while still conformant to the NDR structure – such as namespace string contents and namespace tokens.	3	Rules which may be modified by individual organizations while still conformant to agreed upon data models – such as the use of global or local element declarations. (Changes to the XML Schema Architecture.)	4	Rules that if violated lose conformance with the UN/CEFACT data/process model – such as <code>xsd:redefine</code> , <code>xsd:any</code> , and <code>xsd:substitutionGroups</code> .	5	Rules that relate to extension that are not used by UN/CEFACT and have specific restrictions on their use by other than UN/CEFACT organizations.	6	Rules that relate to extension that are determined by specific organizations.	7	Rules that can be modified while not changing instance validation capability.	1
Rule Categorization																				
ID	Description																			
1	Rules which must not be violated by individual organizations else conformance and interoperability is lost – such as named types.																			
2	Rules which may be modified by individual organizations while still conformant to the NDR structure – such as namespace string contents and namespace tokens.																			
3	Rules which may be modified by individual organizations while still conformant to agreed upon data models – such as the use of global or local element declarations. (Changes to the XML Schema Architecture.)																			
4	Rules that if violated lose conformance with the UN/CEFACT data/process model – such as <code>xsd:redefine</code> , <code>xsd:any</code> , and <code>xsd:substitutionGroups</code> .																			
5	Rules that relate to extension that are not used by UN/CEFACT and have specific restrictions on their use by other than UN/CEFACT organizations.																			
6	Rules that relate to extension that are determined by specific organizations.																			
7	Rules that can be modified while not changing instance validation capability.																			
[R 8059]	All XML Schema design rules MUST be based on the W3C XML Schema Recommendations: XML Schema Part 1: Structures Second Edition and XML Schema Part 2: Datatypes Second Edition .	1																		
[R 935C]	All conformant XML instance documents MUST be based on the W3C suite of technical specifications holding recommendation status.	1																		

[R 9224]	XML Schema MUST follow the standard structure defined in Appendix B of this document.	1				
[R A9E2]	Each element or attribute XML name MUST have one and only one fully qualified XPath (FQXP).	1				
[R AA92]	Element, attribute and type names MUST be composed of words in the English language, using the primary English spellings provided in the Oxford English Dictionary.	1				
[R 9956]	LowerCamelCase (LCC) MUST be used for naming attributes.	1				
[R A781]	UpperCamelCase (UCC) MUST be used for naming elements and types.	1				
[R 8D9F]	Element, attribute and type names MUST be in singular form unless the concept itself is plural.	1				
[R AB19]	XML element, attribute and type names constructed from dictionary entry names MUST NOT include periods, spaces, or other separators; or characters not allowed by W3C XML 1.0 for XML names.	1				
[R 9009]	XML element, attribute and type names MUST NOT use acronyms, abbreviations, or other word truncations, except those included in the defining organizations list of approved acronyms and abbreviations.	1				
[R BFA9]	The acronyms and abbreviations listed by the defining organization MUST always be used in place of the word or phrase they represent.	1				
[R 9100]	Acronyms MUST appear in all upper case except for when the acronym is the first set of characters of an attribute in which case they will be all lower case.	1				
[R 984C]	Each organization's XML Schema components MUST be assigned to a namespace for that organization.	1				
[R 8E2D]	<p>The XML Schema namespaces MUST use the following pattern:</p> <table border="1"> <tr> <td>URN:</td> <td><code>urn:<organization>:<org hierarchy>[:<org hierarchy level>]*:<schematype>:<context category>:<major>:<status></code></td> </tr> <tr> <td>URL:</td> <td><code>http://<organization>/<org hierarchy>[/<org hierarchy level>]*/<schematype>/context category/<major>/<status></code></td> </tr> </table>	URN:	<code>urn:<organization>:<org hierarchy>[:<org hierarchy level>]*:<schematype>:<context category>:<major>:<status></code>	URL:	<code>http://<organization>/<org hierarchy>[/<org hierarchy level>]*/<schematype>/context category/<major>/<status></code>	3
URN:	<code>urn:<organization>:<org hierarchy>[:<org hierarchy level>]*:<schematype>:<context category>:<major>:<status></code>					
URL:	<code>http://<organization>/<org hierarchy>[/<org hierarchy level>]*/<schematype>/context category/<major>/<status></code>					

	<p>Where:</p> <ul style="list-style-type: none"> • organization – An identifier of the organization providing the standard. • org hierarchy – The first level of the hierarchy within the organization providing the standard. • org hierarchy level – Zero to n level hierarchy of the organization providing the standard. • schematype – A token identifying the type of schema module: data odelist documentation. • context category – The context category [business process] for UN/CEFACT from the UN/CEFACT catalogue of common business processes. Other values may be used by the other organizations. • major – The major version number. • status – The status of the schema as: draft standard. 	
[R 8CED]	UN/CEFACT namespaces MUST be defined as Uniform Resource Names.	3
[R B56B]	Published namespace content MUST only be changed by the publishing organization of the namespace or its successor.	1
[R 92B8]	The XML Schema File name for files other than code lists and identifier schemes MUST be of the form <code><SchemaModuleName>_<Version>.xsd</code> , with periods, spaces, or other separators and the words ' XML Schema File ' removed.	3
[R 8D58]	When representing versioning schemes in file names, the period MUST be represented by a lowercase p .	3
[R B387]	Every XML Schema File MUST have a namespace declared, using the <code>xsd:targetNamespace</code> attribute.	1
[R 9354]	A Root XML Schema File MUST be created for each unique business information payload.	1
[R B3E4]	Each Root XML Schema File MUST be named after the <code><BusinessInformationPayload></code> that is expressed in the XML Schema File by using the value of <code><BusinessInformationPayload></code> followed by the words ' XML Schema File ' as the name and placing the name in the Header documentation section of the file.	1
[R 9961]	A Root XML Schema File MUST NOT replicate reusable constructs available in XML Schema Files that can be referenced	1

	through <code>xsd:include</code> .	
[R 8238]	A BIE XML Schema File MUST be created within each namespace that is defined for the primary context category value.	1
[R 8252]	The BIE XML Schema Files MUST be named 'Business Information Entity XML Schema File' by placing the name within the Header documentation section of the file.	1
[R A2F0]	An unqualified BDT XML Schema File MUST be created in the documentation common namespace to represent the set of unrestricted BDTs.	1
[R AA56]	A BDT XML Schema File MUST be created within each namespace that is defined for the primary context category value.	1
[R 847C]	The BDT XML Schema Files MUST be named 'Business Data Type XML Schema File' by placing the name within the header documentation section of the file.	1
[R 8A68]	A Code List XML Schema File MUST be created to convey code list enumerations for each code list being used.	1
[R B0AD]	<p>The name of each Code List XML Schema File as defined in the comment within the XML Schema File MUST be of the form:</p> <p><Code List Agency Identifier Code List Agency Name><Code List Identification Identifier Code List Name>” - Code List XML Schema File”</p> <p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Identifier – Identifies the agency that maintains the code list. • Code List Agency Name – Agency that maintains the code list. • Code List Identification Identifier – Identifies a list of the respective corresponding codes. • Code List Name – The name of the code list as assigned by the agency that maintains the code list. 	1
[R 942D]	Each CCL XML Schema File MUST contain enumeration values for both the actual codes and the code values.	1
[R A8A6]	<p>Each BCL XML Schema File MUST contain enumeration values for both the actual codes and the code values, through one of the following:</p> <ul style="list-style-type: none"> • The restriction of an imported CCL. • The extension of a CCL where the codes and values of the CCL are included and the new extensions are added. 	1

	<ul style="list-style-type: none"> The creation of a new Code List that is used within the context category value namespace. 	
[R AB90]	An Identifier Scheme XML Schema File MUST be created to convey identifier scheme metadata for each scheme being used.	1
[R A154]	<p>The name of each Identifier Scheme XML Schema File as defined in the comment within the XML Schema File MUST be of the form:</p> <pre><Identifier Scheme Agency Identifier Identifier Scheme Agency Name><Identifier Scheme Identification Identifier Identifier Scheme Name>” Identifier Scheme XML Schema File”</pre> <p>Where:</p> <ul style="list-style-type: none"> Identifier Scheme Agency Identifier – Identifies the agency that maintains the identifier scheme. Identifier Scheme Agency Name – Agency that maintains the identifier scheme. Identifier Scheme Identification Identifier – Identifies the scheme. Identifier Scheme Name – The name of the identifier scheme as assigned by the agency that maintains the identifier scheme. 	1
[R BD2F]	A Business Identifier Scheme XML Schema File MUST be created for each Business Scheme used by a BDT.	1
[R AFEB]	Each Business Identifier Scheme XML Schema File MUST contain metadata that describes the scheme or points to the scheme.	1
[R B564]	Imported XML Schema Files MUST be fully conformant to category 1, 2, 3, 4 and 7 rules as defined in rule [R B998] .	4
[R 9733]	Imported XML Schema File components MUST be derived using these NDR rules from artifacts that are fully conformant to the latest version of the UN/CEFACT Core Components Technical Specification.	4
[R 8F8D]	Each <code>xsd:schemaLocation</code> attribute declaration within an XML Schema File MUST contain a resolvable relative path URL.	2
[R BF17]	The <code>xsd:schema</code> version attribute MUST always be declared.	1
[R 84BE]	<p>The <code>xsd:schema</code> version attribute MUST use the following template:</p> <pre><xsd:schema ... version=“<major>”p”<minor>[”p”<revision>]”></pre>	2

	Where: <ul style="list-style-type: none"> • <major> - sequential number of the major version. • <minor> - sequential number of the minor version • <revision> - optional sequential number of the revision. 	
[R 9049]	Every XML Schema File major version number MUST be a sequentially assigned incremental integer greater than zero.	1
[R A735]	Minor versioning MUST be limited to declaring new optional XML content, extending existing XML content, or refinements of an optional nature.	1
[R AFA8]	Minor versions MUST NOT rename existing XML Schema defined artifacts.	1
[R BBD5]	Changes in minor versions MUST NOT break semantic compatibility with prior versions having the same major version number.	1
[R 998B]	XML Schema Files for a minor version XML Schema MUST incorporate all XML Schema components from the immediately preceding version of the XML Schema File.	1
[R 88E2]	Every UN/CEFACT XML Schema File MUST use UTF-8 encoding.	1
[R ABD2]	Every XML Schema File MUST contain a comment that identifies its name immediately following the XML declaration using the format defined in Appendix B-2 .	1
[R BD41]	Every XML Schema File MUST contain a comment that identifies its owning agency, version and date immediately following the schema name comment using the format defined in Appendix B-2 .	1
[R A0E5]	The <code>xsd:elementFormDefault</code> attribute MUST be declared and its value set to qualified.	1
[R A9C5]	The <code>xsd:attributeFormDefault</code> attribute MUST be declared and its value set to unqualified.	1
[R 9B18]	The <code>xsd</code> prefix MUST be used in all cases when referring to the namespace <code>http://www.w3.org/2001/XMLSchema</code> as follows: <code>xmlns:xsd=http://www.w3.org/2001/XMLSchema</code> .	1
[R 90F1]	All required CCTS metadata for ABIEs, BBIEs, ASBIEs, and BDTs must be defined in an XML Schema File.	1
[R 9623]	The name of the CCTS Metadata XML Schema file will be "Core Components Technical Specification Schema File" and will be defined within the header comment within the XML Schema File.	1

[R 9443]	The CCTS Metadata XML Schema File MUST reside in its own namespace and be defined in accordance with rule [R 8E2D] and assigned the prefix <code>ccts</code> .	1
[R AD26]	<code>xsd:notation</code> MUST NOT be used.	1
[R ABFF]	The <code>xsd:any</code> element MUST NOT be used.	4 6
[R AEBB]	The <code>xsd:any</code> attribute MUST NOT be used.	4 6
[R 9859]	Mixed content MUST NOT be used.	1
[R B20F]	<code>xsd:redefine</code> MUST NOT be used.	4 6
[R 926D]	<code>xsd:substitutionGroup</code> MUST NOT be used.	4 6
[R 8A83]	<code>xsd:ID/xsd:IDREF</code> MUST NOT be used.	1
[R B221]	Supplementary Components MUST be declared as Attributes.	1
[R AFEE]	User defined attributes MUST only be used for Supplementary Components.	3
[R 9FEC]	An <code>xsd:attribute</code> that represents a Supplementary Component with variable information MUST be based on an appropriate XML Schema built-in simpleType.	1
[R B2E8]	A <code>xsd:attribute</code> that represents a Supplementary Component which uses codes MUST be based on the <code>xsd:simpleType</code> of the appropriate code list.	1
[R 84A6]	A <code>xsd:attribute</code> that represents a Supplementary Component which uses identifiers MUST be based on the <code>xsd:simpleType</code> of the appropriate identifier scheme.	1
[R B8B6]	Empty elements MUST NOT be used.	3
[R 8337]	The <code>xsd:nillable</code> attribute MUST NOT be used.	1
[R 8608]	Anonymous types MUST NOT be used.	1
[R A4CE]	An <code>xsd:complexType</code> MUST be defined for each CCTS ABIE.	1
[R BC3C]	An <code>xsd:complexType</code> MUST be defined for each CCTS BDT that cannot be fully expressed using an <code>xsd:simpleType</code> .	1

[R A010]	The xsd:all element MUST NOT be used.	1
[R AB3F]	xsd:extension MUST only be used in the BDT XML Schema File.	4 6
[R 9D6E]	xsd:extension MUST only be used for declaring xsd:attributes to accommodate relevant supplementary components.	4 6
[R 9947]	xsd:restriction MUST only be used in BDT XML Schema Files.	1
[R 8AF7]	When xsd:restriction is applied to a data type the resulting type MUST be uniquely named.	1
[R 847A]	Each defined or declared construct MUST use the xsd:annotation element for required CCTS documentation and application information to communicate context.	1
[R A9EB]	Each defined or declared construct MUST use an xsd:annotation and xsd:documentation element for required CCTS documentation.	3
[R 9B07]	Each of the resulting XML Schema Components (xsd:element , xsd:complexType and xsd:simpleType) MUST have an xsd:annotation xsd:appInfo declared that includes zero or more ccts:UsageRule elements and one or more ccts:BusinessContext elements.	1
[R 88DE]	Usage rules MUST be expressed within an xsd:appInfo ccts:UsageRule element.	1
[R B851]	The structure of the ccts:UsageRule element MUST be: <ul style="list-style-type: none"> • ccts:UniqueID [1..1] – A unique identifier for the UsageRule. • ccts:Constraint [1..1] – The actual constraint expression. • ccts:ConstraintType [1..1] – The type of constraint E.g. unstructured, OCL. • ccts:ConditionType [1..1] – The type of condition. Allowed values are pre-condition, post-condition, and invariant. 	1
[R A1CF]	A ccts:ConstraintType code list XML Schema File will be created.	1
[R A538]	Each defined or declared XML Schema artifact MUST use an xsd:annotation and xsd:appInfo element to	1

	communicate the context of the artifact.	
[R B96F]	Each Root, BIE, BDT and BCL XML Schema File MUST be assigned to a unique namespace that represents the primary context category value of its contents.	1
[R B698]	The Root XML Schema File MUST include the BIE and BDT XML Schema Files that reside in its namespace.	1
[R BD9F]	A global element known as the root element, representing the business information payload, MUST be declared in the Root XML Schema File using the XML Schema Component xsd:element .	1
[R A466]	The name of the root element MUST be the same as the name of the business information payload data dictionary name, with separators and spaces removed.	1
[R 8062]	The root element declaration MUST be defined using an xsd:complexType that represents the message content contained within the business information payload.	1
[R 8837]	Each Root XML Schema File MUST define a single xsd:complexType that fully describes the business information payload.	1
[R 9119]	The name of the root schema xsd:complexType MUST be the name of the root element with the word ' Type ' appended.	1
[R 8010]	<p>The Root XML Schema File root element declaration MUST have a structured set of annotations documentation (xsd:annotation xsd:documentation) present in that includes:</p> <ul style="list-style-type: none"> • UniqueID (mandatory): The identifier that uniquely identifies the business information payload, the root element. • VersionID (mandatory): The unique identifier that identifies the version of the business information payload, the root element. • ObjectClassQualifierName (zero or more): Is a word or words which help define and differentiate an ABIE from its associated CC and other BIEs. It enhances the semantic meaning of the DEN to reflect a restriction of the concept, conceptual domain, content model or data value. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the Object class. It is the basis for the DEN. • DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the business information payload. • Definition (mandatory): The semantic meaning of the root 	1

	<p>element.</p> <ul style="list-style-type: none"> BusinessTermName (optional, repeating): A synonym term under which the payload object is known by in industry. 	
[R 8FE2]	The BIE XML Schema File MUST contain an xsd:include statement for the BDT XML Schema File that resides in the same namespace.	1
[R AF95]	For every object class (ABIE) identified in a primary context category, a named xsd:complexType MUST be defined in its corresponding BIE XML Schema File.	1
[R 9D83]	The name of the ABIE xsd:complexType MUST be the ccts:DictionaryEntryName with the spaces and separators removed, with approved abbreviations and acronyms applied and with the 'Details' suffix replaced with 'Type'.	1
[R 90F9]	The cardinality and sequencing of the elements within an ABIE xsd:complexType MUST be as defined by the corresponding ABIE values in the syntax neutral model.	1
[R 9C70]	Every aggregate business information entity (ABIE) xsd:complexType definition content model MUST use zero or more xsd:sequence and/or zero or more xsd:choice elements to reflect each property (BBIE or ASBIE) of its class.	1
[R 81F0]	Repeating series of only xsd:sequence MUST NOT occur.	1
[R 8FA2]	Repeating series of only xsd:choice MUST NOT occur.	1
[R A21A]	Every BBIE within the containing ABIE MUST have a named xsd:simpleType (If the BBIE BDT includes only the content component) or xsd:complexType (If the BBIE BDT includes one or more supplementary components).	1
[R 8B85]	Every BBIE type MUST be named the property term and qualifiers and the representation term of the basic business information entity (BBIE) it represents with the word 'Type' appended.	1
[R 9DA0]	For each ABIE, a named xsd:element MUST be globally declared.	1
[R 9A25]	The name of the ABIE xsd:element MUST be the ccts:DictionaryEntryName with the separators and 'Details' suffix removed and approved abbreviations and acronyms applied.	1
[R B27B]	Every ABIE global element declaration MUST be of the xsd:complexType that represents the ABIE.	1

[R 89A6]	For every BBIE identified in an ABIE, a named xsd:element MUST be locally declared within the xsd:complexType representing that ABIE.	1
[R AEFE]	Each BBIE element name declaration MUST be the property term and qualifiers and the representation term of the BBIE.	1
[R 96D9]	For each BBIE element name declaration where the word 'Identification' is the final word of the property term and the representation term is 'Identifier', the term 'Identification' MUST be removed.	1
[R 9A40]	For each BBIE element name declaration where the word 'Indication' is the final word of the property term and the representation term is 'Indicator', the term 'Indication' MUST be removed from the property term.	1
[R A34A]	If the representation term of a BBIE is 'Text', 'Text' MUST be removed from the name of the element or type definition.	1
[R BCD6]	Every BBIE element declaration MUST be of the BusinessDataType that represents the source basic business information entity (BBIE) data type.	1
[R 9025]	For every ASBIE whose ccts:AggregationKind value = composite , a local element for the associated ABIE MUST be declared in the associating ABIE xsd:complexType content model.	1
[R 9241]	For every ASBIE whose ccts:AggregationKind value = shared , a global element MUST be declared.	1
[R A08A]	Each ASBIE element name MUST be the ASBIE property term and qualifier term(s) and the object class term and qualifier term(s) of the associated ABIE.	1
[R B27C]	Each ASBIE element declaration MUST use the xsd:complexType that represents its associated ABIE.	1
[R ACB9]	For every ABIE xsd:complexType definition a structured set of annotations MUST be present in the following pattern: <ul style="list-style-type: none"> • UniqueID (mandatory): The unique identifier that identifies an ABIE instance in a unique and unambiguous way. • VersionID (mandatory): An unique identifier that identifies the version of an ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the 	1

	<p>associated ABIE from its CC.</p> <ul style="list-style-type: none"> ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE. DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the ABIE. Definition (mandatory): The semantic meaning of the ABIE. BusinessTermName (optional, repeating): A synonym term in which the ABIE is commonly known. 	
[R B0BA]	For every ABIE xsd:complexType definition a structured set of xsd:annotation xsd:appInfo elements MUST be present that fully declare its context.	1
[R BCE9]	For every ABIE usage rule, the ABIE xsd:complexType definition MUST contain a structured set of xsd:annotation xsd:appInfo elements in the following pattern: <ul style="list-style-type: none"> ccts:UniqueID ccts:Constraint ccts:ConstraintType ccts:ConditionType. 	1
[R 88B6]	For every ABIE xsd:element declaration definition, a structured set of annotations MUST be present in the following pattern: <ul style="list-style-type: none"> UniqueID (mandatory): The unique identifier that identifies an ABIE instance in a unique and unambiguous way. VersionID (mandatory): An unique identifier that identifies the version of an ABIE. ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE. DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the ABIE. Definition (mandatory): The semantic meaning of the ABIE. BusinessTermName (optional, repeating): A synonym term in which the ABIE is commonly known. 	1
[R B8BE]	For every BBIE xsd:element declaration a structured set of xsd:annotation xsd:documentation elements MUST be present in the following pattern:	1

	<ul style="list-style-type: none"> • Cardinality (mandatory): Indicates the cardinality of the BBIE within the containing ABIE. • SequencingKey (mandatory): Indicates the sequence of the BBIE within the containing ABIE. • DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the BBIE. • Definition (mandatory): The semantic meaning of the associated BBIE. • BusinessTermName (optional, repeating): A synonym term in which the BBIE is commonly known. • PropertyTermName (mandatory): Represents a distinguishing characteristic of the BBIE. • PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the BBIE. • RepresentationTermName (mandatory): An element of the component name that describes the form in which the BBIE is represented. 	
[R 95EB]	For every BBIE <code>xsd:element</code> declaration a structured set of <code>xsd:annotation</code> <code>xsd:appInfo</code> elements MUST be present that fully declare its context.	1
[R 8BF6]	For every BBIE usage rule, the BBIE <code>xsd:element</code> declaration MUST contain a structured set of <code>xsd:annotation</code> <code>xsd:appInfo</code> elements in the following pattern: <ul style="list-style-type: none"> • <code>ccts:UniqueID</code> • <code>ccts:Constraint</code> • <code>ccts:ConstraintType</code> • <code>ccts:ConditionType</code>. 	1
[R 8D3E]	Every ASBIE global element declaration MUST have a structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements in the following pattern: <ul style="list-style-type: none"> • AssociationKind (mandatory): Indicates the UML AssociationKind value of <code>shared</code> or <code>composite</code> of the associated ABIE. • PropertyTermName (mandatory): Represents a distinguishing characteristic of the ASBIE. • PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the ASBIE. • AssociatedObjectClassName (Mandatory): The name of the 	1

	<p>associated object class.</p> <ul style="list-style-type: none"> AssociatedObjectClassQualifierName (optional, repeating): a name or names that qualify the associated object class. 	
[R 926A]	<p>Every ASBIE xsd:element declaration or xsd:ref occurrence MUST have a structured set of xsd:annotation xsd:documentation elements present in the following pattern:</p> <ul style="list-style-type: none"> Cardinality (mandatory): Indicates the cardinality of the ASBIE within the containing ABIE. SequencingKey (mandatory): Indicates the sequence of the ASBIE within the containing ABIE. DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the ASBIE. Definition (mandatory): The semantic meaning of the ASBIE. BusinessTermName (optional, repeating): A synonym term in which the ASBIE is commonly known. AssociationKind (mandatory): Indicates the UML AssociationKind value of shared or composite of the associated ABIE. PropertyTermName (mandatory): Represents a distinguishing characteristic of the ASBIE. PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the ASBIE. AssociatedObjectClassName (Mandatory): The name of the associated object class. AssociatedObjectClassQualifierName (optional, repeating): a name or names that qualify the associated object class. 	1
[R 9D87]	<p>Every ASBIE xsd:element declaration or ASBIE xsd:ref to an ABIE global element declaration MUST contain a structured set of xsd:annotation xsd:appInfo elements that fully declare its context.</p>	1
[R A76D]	<p>Every ASBIE usage rule xsd:element declaration or ASBIE xsd:ref to an ABIE global element declaration MUST contain a structured set of xsd:annotation xsd:appInfo elements in the following pattern:</p> <ul style="list-style-type: none"> ccts:UniqueID ccts:Constraint ccts:ConstraintType 	1

	<ul style="list-style-type: none"> • ccts:ConditionType. 	
[R 8E0D]	The BDT XML Schema File MUST include (xsd:include) the BCL XML Schema Files and BIS XML Schema Files that are defined in the same namespace.	1
[R B4C0]	The BDT XML Schema File MUST import (xsd:import) the CCL XML Schema Files and CIS XML Schema Files that are used by a BDT contained within the file.	1
[R AE00]	Each CCTS BDT artifact within the UN/CEFACT Data Type Catalogue used by the Root XML Schema Files and the BIE XML Schema File within a given namespace MUST be defined as an xsd:simpleType or xsd:complexType in the BDT XML Schema File with the given namespace.	1
[R 9908]	For every BDT whose content component BVD is defined by a primitive whose facets map directly to the facets of an XSD built-in data type, the BDT MUST be defined as a named xsd:simpleType .	1
[R B91F]	Every BDT whose content component BVD is defined by a primitive whose facets map directly to the facets of an xsd:simpleType MUST contain one xsd:restriction element.	1
[R 9910]	The xsd:restriction element used in a BDT content component BVD defined by a primitive MUST include an xsd:base attribute that defines the specific XSD built-in data type required for the content component.	1
[R A7B8]	The name of a BDT that uses a primitive to define its content component BVD MUST be the BDT ccts:DataTypeQualifier(s) if any, plus the ccts:DataTypeTerm , plus the primitive type name, followed by the word ' Type ' with the separators removed and approved abbreviations and acronyms applied.	1
[R AA60]	A BDT whose content component BVD is defined as an xsd:simpleType whose base is a single code list MUST contain an xsd:restriction element with the xsd:base attribute set to the code lists defined xsd:simpleType .	1
[R 8DB1]	The name of A BDT that uses a single code list to define its content component BVD MUST be its ccts:DataTypeQualifier(s) if any, plus the ccts:DataTypeTerm , plus the code list suffix, followed by the	1

	<p>word 'Type' with the separators removed and approved abbreviations and acronyms applied.</p> <p>The code list suffix MUST be the following: (Any repeated words are eliminated.)</p> <ul style="list-style-type: none"> • <Code List Agency Identifier Code List Agency Name><Code List Identification Identifier Code List Name> <p>Where.</p> <ul style="list-style-type: none"> • Code List Agency Identifier – is the identifier for the agency that code list is from. • Code List Agency Name – is the name of the agency that maintains the code list. • Code List Identification Identifier – is the identifier for the given code list. • Code List Name – is the name for the code list. 	
[R AAD1]	<p>A BDT whose content component BVD is defined by a choice of two or more code lists MUST be defined as an xsd:simpleType that contains an xsd:union element whose xsd:memberType attribute includes the xsd:simpleType definitions of the code lists to be included.</p>	1
[R 973C]	<p>The name of a BDT that uses multiple code lists MUST be its ccts:DataTypeQualifier(s) if any, plus the ccts:DataTypeTerm, plus the code list suffix, followed by the word 'Type' with the separators removed and approved abbreviations and acronyms applied.</p> <p>The suffix MUST be the following: (Any repeated words are eliminated)</p> <ul style="list-style-type: none"> • <Code List Agency Identifier Code List Agency Name><Code List Identification Identifier Code List Name> <p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Identifier – is the identifier for the agency that code list is from. • Code List Agency Name – is the name of the agency that maintains the code list. • Code List Identification Identifier – is the identifier for the given code list. • Code List Name – is the name for the code list. 	1

[R A861]	If a BDT content component BVD is defined as an xsd:simpleType whose base is an identifier scheme, it MUST contain an xsd:restriction element with the xsd:base attribute set to the identifier scheme defined xsd:simpleType .	1
[R 8F96]	<p>The name of A BDT that uses an identifier scheme to define its content component BVD MUST be its ccts:DataTypeQualifier(s) if any, plus the ccts:DataTypeTerm, plus the identifier scheme suffix, followed by the word 'Type' with the separators removed and approved abbreviations and acronyms applied.. The code list suffix MUST be the following: (Any repeated words are eliminated.)</p> <ul style="list-style-type: none"> • <Identifier Scheme Agency Identifier Identifier Scheme Agency Name><Identifier Scheme Identification Identifier Identifier Scheme Name> <p>Where.</p> <ul style="list-style-type: none"> • Identifier Scheme Agency Identifier – is the identifier for the agency that code list is from. • Identifier Scheme Agency Name – is the name for the Agency that owns the identifier scheme. • Identifier Scheme Identification Identifier – is the identifier for the given identifier scheme. • Identifier Scheme Name – is the name for the identifier scheme. 	1
[R AB05]	Every BDT that includes one or more Supplementary Components MUST be defined as an xsd:complexType	1
[R AAA5]	Every BDT xsd:complexType definition MUST have an xsd:simpleContent expression whose xsd:extension base attribute is set to the primitive type or scheme or list that defines its Content Component Business Value Domain.	1
[R 890A]	Every BDT xsd:complexType definition MUST include an xsd:attribute declaration for each Supplementary Component.	1
[R ABC1]	The name of the Supplementary Component xsd:attribute must be the DEN of the Supplementary Component with periods, spaces, and other separators removed.	1
[R 90FB]	The name of a BDT that includes one or more Supplementary Components MUST be:	1

	<ul style="list-style-type: none"> • The BDT ccts:DataTypeQualifier(s) if any, plus • The ccts:DataTypeTerm, plus • The suffix of the Content Component Business Value Domain where: <ul style="list-style-type: none"> ○ The suffix is the primitive type name, the code list token, the series of code list tokens, or the identifier scheme token. <p>plus</p> <ul style="list-style-type: none"> • The ccts:DictionaryEntryName for each Supplementary Component present following the order defined in the Data Type Catalogue, plus • The suffix that represents the Supplementary Component BVD where the suffix is the primitive type name, the code list token, the series of code list tokens, or the identifier scheme token, plus • The word 'Type'. • With all separators removed and approved abbreviations and acronyms applied. 	
[R 80FD]	Every restricted BDT XML Schema Component xsd:type definition MUST be derived from its base type using xsd:restriction unless a non-standard variation from the base type is required.	1
[R A9F6]	Every restricted BDT XML Schema Component xsd:type definition requiring a non-standard variation from its base type MUST be derived from a custom type.	1
[R 8B3D]	Global xsd:element declarations MUST NOT occur in the BDT XML Schema File.	1
[R B340]	Global xsd:attribute declarations MUST NOT occur in the BDT XML Schema File.	1
[R ACA7]	In the BDT XML Schema File, local xsd:attribute declarations MUST only represent CCTS Supplementary Components for the BDT for which they are declared.	1
[R BFE5]	Every BDT definition MUST contain a structured set of annotation documentation in the following sequence and pattern: <ul style="list-style-type: none"> • UniqueID (mandatory): The unique identifier that identifies the BDT in a unique and unambiguous way. • VersionID (mandatory): An unique identifier that identifies 	1

	<p>the version of the BDT.</p> <ul style="list-style-type: none"> • DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the BDT. • Definition (mandatory): The semantic meaning of the BDT. • BusinessTermName (optional, repeating): A synonym term in which the BDT is commonly known. • PropertyTermName (mandatory): Represents a distinguishing characteristic of the BDT and shall occur naturally in the definition. • DataTypeName (mandatory): The name of the DataType. The possible values for the DataType are defined in the Data Type Catalogue. • DataTypeQualifierName (mandatory): Is a word or words which help define and differentiate a Data Type. It further enhances the semantic meaning of the DataType. • DefaultIndicator (mandatory): Indicates that the specific Code List Value is the default for the Code List. • DefaultValue (optional): Is the default value. • DefaultValueSource (optional): Indicates the source for the default value. • SchemeOrListID (optional): The unique identifier assigned to the scheme or list that uniquely identifies it. • SchemeOrListAgencyID (optional): The unique identifier assigned to the Agency that owns or is responsible for the Scheme or Code List being referenced. • SchemeOrListAgencyName (optional): The name of the Agency that owns or is responsible for the Scheme or Code List being referenced. • SchemeOrListModificationAllowedIndicator (optional): Indicates whether the values being validated can be outside the enumerations specified by the Scheme or Code List. • SchemeOrListName (optional): Name of the Scheme or Code List. • SchemeOrListBusinessTermName (optional, repeating): A synonym term under which the Scheme or Code List is commonly known and used in business. (BusinessTerm) 	
[R 9C95]	<p>Every supplementary component xsd:attribute declaration MUST contain a structured set of annotation documentation MUST in the following pattern:</p> <ul style="list-style-type: none"> • Cardinality (mandatory): Indicates the cardinality of the SC within the containing BDT. 	1

	<ul style="list-style-type: none"> • PropertyTermName (mandatory): Represents a distinguishing characteristic of the SC and shall occur naturally in the definition. • RepresentationTermName (mandatory): An element of the component name that describes the form in which the SC is represented. • PrimitiveTypeName (mandatory): The name of the SC PrimitiveType. • DataTypeName (mandatory): The name of the DataType. The possible values for the DataType are defined in the Data Type Catalogue. • DataTypeQualifierName (mandatory): A word or words which help define and differentiate a Data Type. It further enhances the semantic meaning of the DataType. • DefaultIndicator (mandatory): Indicates that the specific Code List Value is the default for the Code List or identifier scheme. • DefaultValue (optional): Is the default value. • DefaultValueSource (optional): Indicates the source for the default value. • SchemeOrListID (optional): The unique identifier assigned to the scheme or list that uniquely identifies it. • SchemeOrListAgencyID (optional): The unique identifier assigned to the Agency that owns or is responsible for the identifier scheme or code list being referenced. • SchemeOrListAgencyName (optional): The name of the Agency that owns or is responsible for the identifier scheme or code list being referenced. • SchemeOrListModificationAllowedIndicator (optional): Indicates whether the values being validated can be outside the enumerations specified by the identifier scheme or code list. • SchemeOrListName (optional): Name of the identifier scheme or code list. • SchemeOrListBusinessTermName (optional, repeating): A synonym term under which the identifier scheme or code list is commonly known and used in business. (BusinessTerm) 	
[R 9E40]	Each code list used by a BDT or BBIE MUST be defined in its own XML Schema File.	2

[R 849E]	<p>Code List XML Schema File names MUST be of the form:</p> <p><Agency Identifier Agency Name>_<List Identification Identifier List Name>_<Version Identifier>.xsd</p> <p>All periods, spaces, or other separators are removed except for the "." before xsd and the "_" between the names.</p> <p>Where:</p> <ul style="list-style-type: none"> • Agency Identifier – identifies the agency that manages the list. The default agencies used are those from DE 3055 but roles defined in DE 3055 cannot be used. • Agency Name – the name of the agency that maintains the list. • List Identification Identifier – identifies a list of the respective corresponding codes or ids. • List Name – the name of a list of codes. • Version Identifier – identifies the version. 	2
[R 8D1D]	Each Code List XML Schema File MUST declare a single global element.	3
[R BE84]	The Code List XML Schema File global element MUST be of the xsd:simpleType that is defined in the Code List XML Schema File.	3
[R A8EF]	Each Code List XML Schema File MUST define one, and only one, named xsd:simpleType for the content component.	1
[R 92DA]	The Code List XML Schema File xsd:simpleType name MUST be the name of the code list root element with the word 'ContentType' appended.	1
[R 962C]	Each code in a Code List XML Schema File MUST be expressed as xsd:enumeration , where the xsd:value for the enumeration is the actual code value.	1
[R A142]	<p>Every Code List MUST contain a structured set of annotation documentation in the following sequence and pattern:</p> <ul style="list-style-type: none"> • SchemeOrListID (mandatory): The unique identifier assigned to the code list. • SchemeOrListAgencyID (optional): The unique identifier assigned to the Agency that owns or is responsible for the code list being referenced. • SchemeOrListAgencyName (optional): The name of the 	1

	<p>Agency that owns or is responsible for the code list being referenced.</p> <ul style="list-style-type: none"> • SchemeOrListModificationAllowedIndicator (optional): Indicates whether the values being validated can be outside the enumerations specified by the code list. • SchemeOrListName (optional): Name of the code list. • SchemeOrListBusinessTermName (optional, repeating): A synonym term under which the code list is commonly known and used in business. (BusinessTerm) 					
[R A814]	<p>Each code list xsd:enumeration MUST contain a structured set of annotations in the following sequence and pattern:</p> <ul style="list-style-type: none"> • Name (mandatory): The name of the code. • Description (optional): Descriptive information concerning the code. 	1				
[R 992A]	<p>Code list XML Schema File namespaces MUST use the following pattern:</p> <table border="1" data-bbox="436 894 1302 1230"> <tr> <td>URN:</td> <td><code>urn:<organization>:<org hierarchy> *[:<org hierarchy level n>]:codelist:common:<major>:<status>:<name></code></td> </tr> <tr> <td>URL:</td> <td><code><a href="http://<organization>/<org hierarchy>*/<org hierarchy level n>]/codelist/common/<major>/<status>/<name>">http://<organization>/<org hierarchy>*/<org hierarchy level n>]/codelist/common/<major>/<status>/<name></code></td> </tr> </table> <p>Where:</p> <ul style="list-style-type: none"> • organization – Identifier of the organization providing the standard. • org hierarchy – The first level of the hierarchy within the organization providing the standard. • org hierarchy level – Zero to n level hierarchy of the organization providing the standard. • codelist – A fixed value token for common codelists. • common – A fixed value token for common codelists. • major – The Major version number of the codelist. • status – The status of the schema as: draft standard • name – The name of the XML Schema File (using upper camel case) with periods, spaces, or other separators and the words 'schema module' removed. <ul style="list-style-type: none"> ○ Code list names are further defined as: <Code List 	URN:	<code>urn:<organization>:<org hierarchy> *[:<org hierarchy level n>]:codelist:common:<major>:<status>:<name></code>	URL:	<code><a href="http://<organization>/<org hierarchy>*/<org hierarchy level n>]/codelist/common/<major>/<status>/<name>">http://<organization>/<org hierarchy>*/<org hierarchy level n>]/codelist/common/<major>/<status>/<name></code>	1
URN:	<code>urn:<organization>:<org hierarchy> *[:<org hierarchy level n>]:codelist:common:<major>:<status>:<name></code>					
URL:	<code><a href="http://<organization>/<org hierarchy>*/<org hierarchy level n>]/codelist/common/<major>/<status>/<name>">http://<organization>/<org hierarchy>*/<org hierarchy level n>]/codelist/common/<major>/<status>/<name></code>					

	<p>Agency Identifier Code List Agency Name> ><divider><Code List Identification Identifier Code List Name></p> <p>Where:</p> <ul style="list-style-type: none"> ▪ Code List Agency Identifier – is the identifier for the agency that code list is from. ▪ Code List Agency Name – is the name of the agency that maintains the code list. ▪ Divider – the divider character for URN is ‘:’ the divider character for URL is ‘/’. ▪ Code List Identification Identifier – is the identifier for the given code list. ▪ Code List Name – is the name for the code list. 	
[R 9FD1]	<p>Each UN/CEFACT maintained CCL XML Schema File MUST be represented by a unique token constructed as follows:</p> <p>clm<Code List Agency Identifier Code List Agency Name><Code List Identification Identifier Code List Name></p> <p>Such that any repeated words are eliminated.</p> <p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Identifier – is the identifier for the agency that code list is from. • Code List Agency Name – is the name of the agency that maintains the code list. • Code List Identification Identifier – is the identifier for the given code list. • Code List Name – is the name for the code list. 	2
[R 86C8]	<p>CCL XML Schema Files MUST NOT import or include any other XML Schema Files.</p>	1
[R B40B]	<p>Each CCL XML Schema File xsd:simpleType MUST use an xsd:restriction element whose base attribute is xsd:token.</p>	1
[R 8F2D]	<p>BCL XML Schema file MUST be used to</p> <ul style="list-style-type: none"> • Extend existing CCL or • Define a codelist where one does not exist or • Restrict the value of a CCL for a context category 	1

[R 87A9]	BCL XML Schema Files MUST import only CCL XML Schema Files it uses directly.	1
[R 882D]	In each BCL XML Schema File the xsd:restriction element base attribute value MUST be set to xsd:token . or the 'Content Type ' from the CCL that is being used.	1
[R A1EE]	Each identifier scheme used by a BDT or BBIE MUST be defined in its own XML Schema file.	2
[R A50B]	<p>Identifier Scheme XML Schema File names MUST be of the form: <Agency Identifier Agency Name>_<Scheme Identification Identifier Scheme Name>_<Version Identifier>.xsd</p> <p>All periods, spaces, or other separators are removed except for the "." before xsd and the "_" between the names.</p> <p>Where:</p> <ul style="list-style-type: none"> • Agency Identifier – identifies the agency that manages the identifier scheme. The default agencies used are those from DE 3055 but roles defined in DE 3055 cannot be used. • Agency Name – the name of the agency that maintains the scheme. • Scheme Identification Identifier – identifies the identifier scheme. • Scheme Name – the name of the identifier scheme. • Version Identifier – identifies the version of the scheme. 	2
[R BFEB]	Each Identifier Scheme XML Schema File MUST declare a single global element.	3
[R B236]	The Identifier Scheme XML Schema File root element MUST be of the xsd:simpleType that is defined in the Identifier Scheme XML Schema File.	3
[R 9451]	Each Identifier Scheme XML Schema File MUST define one, and only one, named xsd:simpleType for the content component.	1
[R 8CD3]	The Identifier Scheme XML Schema File xsd:simpleType name MUST be the name of the identifier scheme root element with the word 'Content Type ' appended.	1

[R B30A]	<p>Every Identifier Scheme MUST contain a structured set of annotation documentation in the following sequence and pattern:</p> <ul style="list-style-type: none"> • SchemeOrListID (mandatory): The unique identifier assigned to the Identifier Scheme. • SchemeOrListAgencyID (optional): The unique identifier assigned to the Agency that owns or is responsible for the identifier scheme being referenced. • SchemeOrListAgencyName (optional): The name of the Agency that owns or is responsible for the identifier scheme being referenced. • SchemeOrListModificationAllowedIndicator (optional): Indicates whether the values being validated can be outside the pattern specified by the scheme. • SchemeOrListName (optional): Name of the identifier scheme. • SchemeOrListBusinessTermName (optional, repeating): A synonym term under which the identifier scheme is commonly known and used in business. (BusinessTerm) 	1				
[R 9CCF]	<p>Identifier scheme XML Schema File namespaces MUST use the following pattern:</p> <table border="1" data-bbox="440 1035 1304 1367"> <tr> <td data-bbox="440 1035 548 1199">URN:</td> <td data-bbox="548 1035 1304 1199">urn:<organization>:<org hierarchy>*[:<org hierarchy level n>]:identifierscheme:common:<major>:<status>:<name></td> </tr> <tr> <td data-bbox="440 1199 548 1367">URL:</td> <td data-bbox="548 1199 1304 1367"><a href="http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/identifierscheme/common/<major>/<status>/<name>">http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/identifierscheme/common/<major>/<status>/<name></td> </tr> </table> <p>Where:</p> <ul style="list-style-type: none"> • organization – Identifier of the organization providing the standard. • org hierarchy – The first level of the hierarchy within the organization providing the standard. • org hierarchy level – Zero to n level hierarchy of the organization providing the standard. • identifierscheme – A fixed value token for common identifier schemes. • common – A fixed value token for common identifier schemes. • major – The Major version number of the identifier scheme. 	URN:	urn:<organization>:<org hierarchy>*[:<org hierarchy level n>]:identifierscheme:common:<major>:<status>:<name>	URL:	<a href="http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/identifierscheme/common/<major>/<status>/<name>">http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/identifierscheme/common/<major>/<status>/<name>	1
URN:	urn:<organization>:<org hierarchy>*[:<org hierarchy level n>]:identifierscheme:common:<major>:<status>:<name>					
URL:	<a href="http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/identifierscheme/common/<major>/<status>/<name>">http://<organization>/<org hierarchy>*[/<org hierarchy level n>]/identifierscheme/common/<major>/<status>/<name>					

	<ul style="list-style-type: none"> • status – The status of the schema as: draft standard • name – The name of the XML Schema File (using upper camel case) with periods, spaces, or other separators and the words ‘schema module’ removed. <ul style="list-style-type: none"> ○ Identifier scheme names are further defined as: <Identifier Scheme Agency Identifier Identifier Scheme Agency Name><divider><Identifier Scheme Identification Identifier Identifier Scheme Name> <p>Where:</p> <ul style="list-style-type: none"> ▪ Identifier Scheme Agency Identifier – is the identifier for the agency that identifier scheme is from. ▪ Identifier Scheme Agency Name – is the name of the agency that maintains the identifier scheme. ▪ Divider – the divider character for URN is ‘:’ the divider character for URL is ‘/’. ▪ Identifier Scheme Identification Identifier – is the identifier for the given identifier scheme. ▪ Identifier Scheme Name – is the name for the identifier scheme. 	
[R B2BC]	<p>Each UN/CEFACT maintained CIS XML Schema File MUST be represented by a unique token constructed as follows:</p> <pre>clm<Identifier Scheme Agency Identifier Identifier Scheme Agency Name><Identifier Scheme Identification Identifier Identifier Scheme Name></pre> <p>Such that any repeated words are eliminated.</p> <p>Where:</p> <ul style="list-style-type: none"> • Identifier Scheme Agency Identifier – is the identifier for the agency that the identifier scheme is from. • Identifier Scheme Agency Name – is the name of the agency that maintains the identifier scheme. • Identifier Scheme Identification Identifier – is the identifier for the given identifier scheme. • Identifier Scheme Name – is the name for the identifier scheme. 	2
[R A6C0]	CIS XML Schema Files MUST NOT import or include any other XML Schema Files.	1

[R 9DDA]	Each CIS XML Schema File <code>xsd:simpleType</code> MUST use an <code>xsd:restriction</code> element whose base attribute value = <code>xsd:token</code> .	1
[R A1E3]	BIS XML Schema file MUST be used to <ul style="list-style-type: none"> Define an identifier scheme where one does not exist or Redefine an existing CIS 	1
[R A4BF]	BIS XML Schema Files MUST NOT use <code>xsd:import</code> or <code>xsd:include</code> .	1
[R 96B0]	Each CIS XML Schema File <code>xsd:simpleType</code> MUST use an <code>xsd:restriction</code> element whose base attribute value is <code>xsd:token</code> .	1
[R ACE9]	All XML MUST be instantiated using UTF. UTF-8 should be used if possible, if not UTF-16 should be used.	1
[R A1B9]	The <code>xsi</code> namespace prefix MUST be used to reference the " <code>http://www.w3.org/2001/XMLSchema-instance</code> " namespace and anything defined by the W3C XMLSchema-instance namespace.	1
[R 9277]	The <code>xsi:nil</code> attribute MUST NOT appear in any conforming instance.	1
[R 8250]	The <code>xsi:type</code> attribute MUST NOT be used within an XML Instance.	1
[R A884]	The attributes for scheme or list supplementary components SHOULD NOT be used within an XML Instance.	1

4387

Naming and Design Rules for the Alternative Business Message Syntax in Appendix I

4388

[R 8E89]	Schema identity constraints MUST be used to implement references between elements when they represent ABIE's that are linked by an association, whose <code>AggregationKind</code> property is 'shared'.	1
[R 8103]	The uniqueness (<code>xsd:unique</code>) constraint MUST be used rather than the key (<code>xsd:key</code>) constraint to define the keys and enforce that their values are unique within their scope of application.	1
[R 8EE7]	Identifiers used in schema identify constraints or for dynamic	1

	referencing MUST be declared as attributes.	
[R 991C]	User defined attributes MUST only be used for Supplementary Components or to serve as identifiers in identity constraints. Modification to Rule [R AFEE].	1
[R A577]	Empty elements MUST NOT be used, except when their definition includes an identifier attribute that serves to reference another element via schema identity constraints. Modification to Rule [R B8B6].	1
[R BA43]	Each ABIE element that is a scope element of a set of XML Schema identity constraints MUST contain one or more <code>xsd:unique</code> constraint declarations.	1
[R 88DB]	Each ABIE that is the target of a reference under a scope element MUST be the object of a <code>xsd:unique</code> constraint declaration via a <code>xsd:selector/@xpath</code> component.	1
[R B40C]	The name of an <code>xsd:unique</code> constraint MUST be constructed as follows: “<Scope element><Referenced Element>Key” Where: <ul style="list-style-type: none"> • Scope element – is the name of the scope element. • Referenced Element – is the element name being referenced within the scope element. 	1
[R AC2D]	For each referenced element in a given scope one <code>xsd:keyref</code> constraint involving the reference attribute that point to the referenced element MUST be declared in the XML Schema, under the scope element.	1
[R 9BE8]	The <code>xsd:keyref/xsd:selector/@xpath</code> component must be such that it selects all the elements where the key reference attribute may occur.	1
[R 858D]	The name of an <code>xsd:keyref</code> constraint MUST be constructed as follows: “<Scope Element ><Referenced Element>Reference” Where: <ul style="list-style-type: none"> • Scope Element – is the name of the scope element. • Referenced Element – is the element name being referenced within the scope element. 	1
[R 886A]	Uniqueness of <code>@key</code> attributes that are not involved in structural referencing MUST NOT be enforced by the schema via identity constraints. Uniqueness of <code>@key</code> attributes should be assured by use of adequate algorithms for the generation of the identifiers	1

	(e.g. UUIDs).	
[R 8EA2]	Every aggregate business information entity (ABIE) xsd:complexType definition MUST contain an optional, locally defined, “key” attribute that MAY be used as the complex element identifier in the XML document where it appears.	1
R 92C0]	“key” MUST be a reserved attribute name.	1
[R 8A37]	Every “key” local attribute declaration MUST be of the type xsd:token .	1
[R B78E]	Every ASBIE whose ccts:AggregationKind value= Shared , and where the association must be implemented as a referenced property, an equivalent referencing element pointing to the associated ABIE MUST be locally declared.	1
[R B173]	For each equivalent referencing element an xsd:complexType MUST be declared. Its structure will be an empty element with a local attribute.	1
[R AEDD]	The equivalent referencing element MUST have a name composed of the ASBIE property term and property qualifier term(s)) and the object term and qualifier term(s) of the associated ABIE.	1
[R B3E5]	When there is no ASBIE property term the generic property term “Referred” followed by the name of the associated ABIE MUST be used as a naming convention to distinguish this element from the ABIE element.	1
[R B523]	The name of the local attribute that is part of the empty element MUST be composed of the object class term and object qualifier term(s) of the ABIE being referenced, followed by the suffix ‘ Reference ’.	1
[R 8B0E]	The name of the xsd:complexType representing the equivalent referencing element MUST be composed of the object class term and object qualifier term(s) of the ABIE being referenced, followed by the suffix ‘ ReferenceType ’.	1
[R B7D6]	Each equivalent referencing element MUST be declared using the xsd:complexType that relates to the ABIE being referenced.	1

4389

4390 **Appendix K. Glossary**

4391 **Aggregate Business Information Entity (ABIE)** – A collection of related pieces of
4392 business information that together convey a distinct business meaning in a specific
4393 business context. Expressed in modelling terms, it is the representation of an object
4394 class, in a specific business context.

4395 **Aggregate Core Component (ACC)** – A collection of related pieces of business
4396 information that together convey a distinct business meaning, independent of any
4397 specific business context. Expressed in modelling terms, it is the representation of
4398 an object class, independent of any specific business context.

4399 **Aggregation** – An Aggregation is a special form of Association that specifies a
4400 whole-part relationship between the aggregate (whole) and a component part.

4401 **Artefact** – A piece of information that is produced, modified, or used by a process.
4402 An artefact can be a model, a model element, or a document. A document can
4403 include other documents. CCTS artefacts include all registry classes as specified in
4404 Section 9 of the *CCTS Technical Specification* and all subordinate named constructs
4405 of a CCTS registry class.

4406 **Assembly Rules** – Assembly Rules group sets of unrefined business information
4407 entities into larger artefacts suitable for expressing complete business information
4408 exchange concepts.

4409 **Association Business Information Entity (ASBIE)** – A business information entity
4410 that represents a complex business characteristic of a specific object class in a
4411 specific business context. It has a unique business semantic definition. An
4412 Association Business Information Entity represents an Association Business
4413 Information Entity property and is therefore associated to an Aggregate Business
4414 Information Entity, which describes its structure. An Association Business
4415 Information Entity is derived from an Association Core Component.

4416 **Association Business Information Entity Property** – A business information entity
4417 property for which the permissible values are expressed as a complex structure,
4418 represented by an Aggregate Business Information Entity.

4419 **Association Core Component (ASCC)** – A core component which constitutes a
4420 complex business characteristic of a specific Aggregate Core Component that
4421 represents an object class. It has a unique business semantic definition. An
4422 Association Core Component represents an Association Core Component Property
4423 and is associated to an Aggregate Core Component, which describes its structure.

4424 **Association Core Component Property** – A core component property for which the
4425 permissible values are expressed as a complex structure, represented by an
4426 Aggregate Core Component.

4427 **Attribute** – A named value or relationship that exists for some or all instances of
4428 some entity and is directly associated with that instance.

4429 **Backward Compatibility** – Any XML instance that is valid against one schema
4430 version will also validate against the previous schema version.

4431 **Basic Business Information Entity (BBIE)** – A business information entity that
4432 represents a singular business characteristic of a specific object class in a specific

- 4433 business context. It has a unique business semantic definition. A Basic Business
4434 Information Entity represents a Basic Business Information Entity property and is
4435 therefore linked to a data type, which describes its values. A Basic Business
4436 Information Entity is derived from a Basic Core Component.
- 4437 **Basic Business Information Entity Property** – A business information entity
4438 property for which the permissible values are expressed by simple values,
4439 represented by a data type.
- 4440 **Basic Core Component (BCC)** – A core component which constitutes a singular
4441 business characteristic of a specific Aggregate Core component that represents a
4442 object class. It has a unique business semantic definition. A Basic Core Component
4443 represents a Basic Core Component property and is therefore of a data type, which
4444 defines its set of values. Basic core components function as the properties of
4445 Aggregate Core components.
- 4446 **Basic Core Component (BCC) Property** – A core component property for which
4447 the permissible values are expressed by simple values, represented by a data type.
- 4448 **Business Context** – The formal description of a specific business circumstance as
4449 identified by the values of a set of context categories, allowing different business
4450 circumstances to be uniquely distinguished.
- 4451 **Business Data Type** – A business data type is a data type, which consists of one
4452 and only one BDT content component, that carries the actual content plus one or
4453 more BDT supplementary component giving an essential extra definition to the CDT
4454 content component. BDTs do not have business semantics.
- 4455 **Business Data Type Content Component** – Defines the primitive type used to
4456 express the content of a core data type.
- 4457 **Business Data Type Content Component Restriction** – The formal definition of a
4458 format restriction that applies to the possible values of a core data type content
4459 component.
- 4460 **Business Data Type Supplementary Component** – Gives additional meaning to
4461 the business data type content component.
- 4462 **Business Data Type Supplementary Component Restrictions** – The formal
4463 definition of a format restriction that applies to the possible values of a business data
4464 type Supplementary Component.
- 4465 **Business Information Entity (BIE)** – A piece of business data or a group of pieces
4466 of business data with a unique business semantic definition. A business information
4467 entity can be a Basic Business Information Entity (BBIE), an Association Business
4468 Information Entity (ASBIE), or an Aggregate Business Information Entity (ABIE).
- 4469 **Business Information Entity (BIE) Property** – A business characteristic belonging
4470 to the Object Class in its specific business context that is represented by an
4471 Aggregate Business Information Entity.
- 4472 **Business Libraries** – A collection of approved process models specific to a line of
4473 business (e.g., shipping, insurance).
- 4474 **Business Process** – The business process as described using the UN/CEFACT
4475 Catalogue of Common business processes.

- 4476 **Business Process Context** – The business process name(s) as described using
4477 the *UN/CEFACT Catalogue of Common Business Processes* as extended by the
4478 user.
- 4479 **Business Process Role Context** – The actors conducting a particular business
4480 process, as identified in the *UN/CEFACT Catalogue of Common Business*
4481 *Processes*.
- 4482 **Business Semantic(s)** – A precise meaning of words from a business perspective.
- 4483 **Business Term** – This is a synonym of the dictionary entry name under which the
4484 artefact is commonly known and used in business. A CCTS artefact may have
4485 several business terms or synonyms.
- 4486 **Cardinality** – An indication of the minimum and maximum occurrences for a
4487 characteristic: not applicable (0..0), optional (0..1), optional repetitive (0..*)
4488 mandatory (1..1), mandatory repetitive (1..*), fixed (n..n) where n is a non-zero
4489 positive integer.
- 4490 **Catalogue of Business Information Entities** – This represents the approved set of
4491 Business Information Entities from which to choose when applying the Core
4492 Component discovery process
- 4493 **Classification Scheme** – This is an officially supported scheme to describe a given
4494 context category.
- 4495 **Composition** – A form of aggregation which requires that a part instance be
4496 included in at most one composite at a time, and that the composite object is
4497 responsible for the creation and destruction of the parts. Composition may be
4498 recursive.
- 4499 **Context** – Defines the circumstances in which a business process may be used.
4500 This is specified by a set of context categories known as business context.
- 4501 **Context Category** – A group of one or more related values used to express a
4502 characteristic of a business circumstance.
- 4503 **Controlled Vocabulary** – A supplemental vocabulary used to uniquely define
4504 potentially ambiguous words or business terms. This ensures that every word within
4505 any of the core component names and definitions is used consistently,
4506 unambiguously and accurately.
- 4507 **Core Component (CC)** – A building block for the creation of a semantically correct
4508 and meaningful information exchange package. It contains only the information
4509 pieces necessary to describe a specific concept.
- 4510 **Core Component Library (CCL)** – The Core Component Library is the part of the
4511 registry/repository in which Core Components shall be stored as registry classes.
4512 The Core Component Library will contain all the registry classes.
- 4513 **Core Component Property** – A business characteristic belonging to the object class
4514 represented by an Basic Core Component property or an Association Core
4515 Component property.
- 4516 **Core Component Type (CCT)** –
- 4517 **Core Data Type (CDT)** – The Core Data Type is the data type that constitutes the
4518 value space for the allowed values for a property.

- 4519 **Definition** – This is the unique semantic meaning of a core component, business
4520 information entity, business context or data type.
- 4521 **Dictionary Entry Name** – This is the official name of a CCTS-conformant artefact.
- 4522 **Facet** – A facet is a constraining value that represents a component restriction of a
4523 Business Data Type content or supplementary component so as to define its allowed
4524 value space.
- 4525 **Geopolitical Context** – Geographic factors that influence business semantics (e.g.,
4526 the structure of an address).
- 4527 **Industry Classification Context** – Semantic influences related to the industry or
4528 industries of the trading partners (e.g., product identification schemes used in
4529 different industries).
- 4530 **Information Entity** – A reusable semantic building block for the exchange of
4531 business-related information.
- 4532 **LowerCamelCase (LCC)** – LowerCamelCase is a lexical representation of
4533 compound words or phrases in which the words are joined without spaces and all but
4534 the first word are capitalized within the resulting compound.
- 4535 **Message Assembly** – The process whereby Business Information Entities are
4536 assembled into a usable message for exchanging business information.
- 4537 **Naming Convention** – The set of rules that together comprise how the dictionary
4538 entry name for CCTS artefacts are constructed.
- 4539 **Object Class** – The logical data grouping (in a logical data model) to which a data
4540 element belongs (ISO11179). The object class is the part of a core component or
4541 business information entity dictionary entry name that represents an activity or
4542 object.
- 4543 **Object Class Term** – A component of the name of a core component or business
4544 information entity which represents the object class to which it belongs.
- 4545 **Official Constraints Context** – Legal and governmental influences on semantics
4546 (e.g. hazardous materials information required by law when shipping goods).
- 4547 **Primitive Type** – A primitive type, also known as a base type or built-in type, is the
4548 basic building block for the representation of a value as expressed by more complex
4549 data types.
- 4550 **Product Classification Context** – Factors influencing semantics that are the result
4551 of the goods or services being exchanged, handled, or paid for, etc. (e.g. the buying
4552 of consulting services as opposed to materials).
- 4553 **Property Term** – A semantically meaningful name for the characteristic of the Object
4554 Class that is represented by the core component property. It shall serve as basis for
4555 the DEN of the basic and Association Core Components that represents this core
4556 component property.
- 4557 **Qualified Business Data Type** – A qualified business data type contains restrictions
4558 on a business data type content or business data type supplementary component(s).
- 4559 **Qualifier Term** – A word or group of words that help define and differentiate an item
4560 (e.g. a business information entity or a business data type) from its associated items

- 4561 (e.g. from a core component, a core data type, another business information entity or
4562 another business data type).
- 4563 **Registry** – An information system that manages and references artifacts that are
4564 stored in a repository. The term registry implies a combination of registry/repository.
- 4565 **Registry Class** – The formal definition of all the common information necessary to
4566 be recorded in the registry by a registry artefact – core component, a business
4567 information entity, a data type or a business context.
- 4568 **Repository** – an information system that stores artifacts.
- 4569 **Representation Term** – The type of valid values for a Basic Core Component or
4570 Basic Business Information Entity.
- 4571 **Scope element** – (for identity constraints) – The element whose schema declaration
4572 contains the identity constraints.
- 4573 **Supporting Role Context** – Semantic influences related to non-partner roles (e.g.,
4574 data required by a third-party shipper in an order response going from seller to
4575 buyer.).
- 4576 **Syntax Binding** – The process of expressing a Business Information Entity in a
4577 specific syntax.
- 4578 **System Capabilities Context** – This context category exists to capture the
4579 limitations of systems (e.g. an existing back office can only support an address in a
4580 certain form).
- 4581 **UMM Information Entity** – A UMM information entity realizes structured business
4582 information that is exchanged by partner roles performing activities in a business
4583 transaction. Information entities include or reference other information entities
4584 through associations.”
- 4585 **Unique Identifier** – The identifier that references a registry class instance in a
4586 universally unique and unambiguous way.
- 4587 **UpperCamelCase (UCC)** – UpperCamelCase is a lexical representation of
4588 compound words or phrases in which the words are joined without spaces and are
4589 capitalized within the resulting compound.
- 4590 **Usage Rules** – Usage rules describe a constraint that describes specific conditions
4591 that are applicable to a component in the model.
- 4592 **User Community** – A user community is a group of practitioners, with a publicized
4593 contact address, who may define Context profiles relevant to their area of business.
4594 Users within the community do not create, define or manage their individual context
4595 needs but conform to the community’s standard. Such a community should liaise
4596 closely with other communities and with general standards-making bodies to avoid
4597 overlapping work. A community may be as small as two consenting organizations.
- 4598 **Version** – An indication of the evolution over time of an instance of a core
4599 component, data type, business context, or business information entity.
- 4600 **XML Schema** – A generic term used to identify the family of grammar based XML
4601 document structure validation languages to include the more formal W3C XML
4602 Schema Definition Language, ISO 8601 Document Type Definition, or Schematron.
4603 An XML Schema is a collection of schema components.

4604 **XML Schema Definition Language Component** –The 13 building blocks that
4605 comprise the abstract data model of the schema, consisting of simple type
4606 definitions, complex type definitions, attribute declarations, element declarations,
4607 attribute group definitions, identity-constraint definitions, model group definitions,
4608 notation declarations, annotations, model groups, particles, wildcards, and attribute
4609 uses.

4610 **XML Schema Definition Language** – The World Wide Web Consortiums official
4611 recommendation for describing the structure and constraining the contents of XML
4612 documents.

4613 **XML Schema Document** – An XML conformant document expression of an XML
4614 schema.

4615

4616

4617

4618

Disclaimer

4619 The views and specification expressed in this document are those of the authors and
4620 are not necessarily those of their employers. The authors and their employers
4621 specifically disclaim responsibility for any problems arising from correct or incorrect
4622 implementation or use of this design.

4623 Copyright Statement

4624

4625 Copyright © UN/CEFACT 2008. All Rights Reserved.

4626

4627 This document and translations of it may be copied and furnished to others, and
4628 derivative works that comment on or otherwise explain it or assist in its
4629 implementation may be prepared, copied, published and distributed, in whole or in
4630 part, without restriction of any kind, provided that the above copyright notice and this
4631 paragraph are included on all such copies and derivative works. However, this
4632 document itself may not be modified in any way, such as by removing the copyright
4633 notice or references to UN/CEFACT except as required to translate it into languages
4634 other than English.

4635 The limited permissions granted above are perpetual and will not be revoked by
4636 UN/CEFACT or its successors or assigns.

4637 This document and the information contained herein is provided on an "AS IS" basis
4638 and UN/CEFACT DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED,
4639 INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE
4640 INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED
4641 WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR
4642 PURPOSE.