



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

UN/CEFACT
XML Naming and Design Rules Technical Specification
Version 3.0

17 December 2009

18 Abstract

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

27

28 **Table of Contents**

29	Abstract	2
30	Table of Contents	3
31	1 Status of This Document	8
32	2 XML Naming and Design Rules Project Team Participants	9
33	2.1 Acknowledgements	9
34	2.2 Disclaimer	10
35	2.3 Contact Information	10
36	3 Introduction	11
37	3.1 Summary of Contents of Document	11
38	3.1.1 Notation	12
39	3.2 Audience	12
40	4 Objectives	13
41	4.1 Goals of the Technical Specification	13
42	4.2 Requirements	13
43	4.3 Conformance	13
44	4.4 Caveats and Assumptions	14
45	4.5 Guiding Principles	15
46	5 XML Schema Architecture	16
47	5.1 Overall XML Schema Structure	16
48	5.2 Relationship to CCTS	17
49	5.2.1 CCTS	18
50	5.2.2 The XML Schema Components	18
51	5.2.3 Context Categories	20
52	5.3 Business Message Syntax Binding	20
53	5.3.1 Nesting Alternative	22
54	5.4 Naming and Modeling Constraints	22
55	5.5 Reusability Scheme	25
56	5.6 Namespace Scheme	28
57	5.6.1 Namespace Uniform Resource Identifiers	29
58	5.6.2 Namespace Tokens	31
59	5.7 XML Schema Files	31
60	5.7.1 Root XML Schema Files	34
61	5.7.2 Business Information Entity XML Schema Files	35
62	5.7.3 Business Data Type XML Schema Files	35

63	5.7.4	XML Schema Built-in Type Extension XML Schema File.....	36
64	5.7.5	Code List XML Schema Files.....	36
65	5.7.6	Identifier Schemes	38
66	5.7.7	Other Standard Bodies BIE XML Schema Files.....	40
67	5.8	Schema Location	40
68	5.9	Versioning Scheme	42
69	5.9.1	Major Versions	42
70	5.9.2	Minor Versions.....	43
71	6	Application of Context	44
72	7	General XML Schema Definition Language Conventions	45
73	7.1	Overall XML Schema Structure and Rules.....	45
74	7.1.1	XML Schema Declaration	45
75	7.1.2	XML Schema File Identification and Copyright Information	45
76	7.1.3	Schema Declaration.....	46
77	7.1.4	CCTS Artefact Metadata.....	46
78	7.1.5	Constraints on Schema Construction.....	47
79	7.2	Attribute and Element Declarations.....	47
80	7.2.1	Attributes.....	47
81	7.2.2	Elements.....	48
82	7.3	Type Definitions	48
83	7.3.1	Simple Type Definitions	49
84	7.3.2	Complex Type Definitions	49
85	7.4	Use of Extension and Restriction	50
86	7.4.1	Extension	50
87	7.4.2	Restriction.....	50
88	7.5	Annotation	51
89	7.5.1	Documentation.....	51
90	7.5.2	Application Information (AppInfo).....	56
91	8	XML Schema Files	60
92	8.1	XML Schema Files, Context and Namespaces	60
93	8.2	Root XML Schema Files.....	62
94	8.2.1	XML Schema Structure.....	62
95	8.2.2	Imports and Includes.....	63
96	8.2.3	Element Declarations.....	64
97	8.2.4	Type Definitions	65

98	8.2.5	Annotations.....	66
99	8.3	Business Information Entity XML Schema Files.....	67
100	8.3.1	Schema Structure.....	68
101	8.3.2	Imports and Includes.....	68
102	8.3.3	Type Definitions.....	69
103	8.3.4	Element Declarations and References.....	72
104	8.3.5	Annotation.....	75
105	8.4	Business Data Type XML Schema Files.....	83
106	8.4.1	Use of Business Data Type XML Schema Files.....	83
107	8.4.2	XML Schema Structure.....	83
108	8.4.3	Imports and Includes.....	84
109	8.4.4	Type Definitions.....	84
110	8.4.5	BDT Attribute and Element Declarations.....	92
111	8.4.6	BDT Annotations.....	93
112	8.5	XML Schema Built-in Type Extension XML Schema File.....	98
113	8.5.1	XML Schema Structure.....	98
114	8.5.2	Type Definitions.....	99
115	8.6	Code List XML Schema Files.....	99
116	8.6.1	General Code List XML Schema Components.....	100
117	8.6.2	Common Code List XML Schema Components.....	104
118	8.6.3	Business Code List XML Schema Components.....	108
119	8.7	Identifier Scheme XML Schema Files.....	110
120	8.7.1	General Identifier Scheme XML Schema Components.....	111
121	8.7.2	Common Identifier Scheme XML Schema Components.....	114
122	8.7.3	Business Identifier Scheme XML Schema Components.....	118
123	9	XML Instance Documents.....	120
124	9.1	Character Encoding.....	120
125	9.2	xsi:schemaLocation.....	120
126	9.3	Empty Content.....	120
127	9.4	xsi:type.....	121
128	9.5	Supplementary Components.....	121
129		Appendix A. Related Documents.....	122
130		Appendix B. Overall Structure.....	123
131	B.1	XML Declaration.....	123
132	B.2	Schema Module Identification and Copyright Information.....	123

133	B.3 Schema Start-Tag.....	124
134	B.4 Includes	125
135	B.5 Imports	126
136	B.6 Elements.....	127
137	B.7 Root element.....	127
138	B.8 Type Definitions	128
139	Appendix C. ATG Approved Acronyms and Abbreviations.....	133
140	Appendix D. Core Component XML Schema File	134
141	Appendix E. Business Data Type XML Schema File.....	135
142	Appendix F. Annotation Templates	136
143	F.1 Annotation Documentation.....	137
144	F.2 Annotation Application Information.....	139
145	Appendix G. UN/CEFACT Data Type Catalogue	143
146	Appendix H. Use Cases for Code Lists	144
147	H.1 Referencing a Common Code List as a Supplementary	
148	Component in a Business Data Type.....	145
149	H.2 Referencing any code list using BDT CodeType	146
150	H.3 Referencing a Common Code List in a BDT	147
151	H.4 Choosing or Combining Values from Several Code Lists	147
152	H.5 Restricting the Allowed Code Values	148
153	Appendix I. Alternative Business Message Syntax Binding.....	150
154	I.1 XML Schema Architecture.....	150
155	I.1.1 Message Assembly Considerations	150
156	I.1.2. Requirements for XML Element Referencing	150
157	I.1.2.1 Implementation of Aggregations – Nesting or Referencing	150
158	I.1.2.2 Other Usages of XML Referencing.....	151
159	I.1.2.3 Schema Validation Requirements for XML References	151
160	I.2 General XML Schema Language Conventions	152
161	I.2.1 Overall XML Schema Structure and Rules.....	152
162	I.2.2 Attribute and Element Declarations.....	153
163	I.3 XML Schema Files	154
164	I.3.1 Root XML Schema Files.....	154
165	I.3.2 Business Information Entities XML Schema Files	156
166	Appendix J. Date. Type, DateTime. Type and Time. Type Data Type	
167	Representations and Their Translation to XML Schema	
168	Types	158

169 Appendix K. Naming and Design Rules List.....172
170 K.1 Naming and Design Rules for the Alternative Business Message
171 Syntax in Appendix I207
172 Appendix L. Glossary210
173 Copyright Statement.....217
174

175 **1 Status of This Document**

176 This UN/CEFACT technical specification has been developed in accordance with the
177 UN/CEFACT/TRADE/R.650/Rev.4/Add.1/Rev.1 Open Development Process (ODP)
178 for technical specifications. The UN/CEFACT Applied Technology Group (ATG) has
179 approved it for distribution.

180 This technical specification contains information to guide in interpretation or
181 implementation.

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

183 Distribution of this document is unlimited.

184 This version: UN/CEFACT XML Naming and Design Rules, Version 3.0 of 17
185 December, 2009

186 Previous version: UN/CEFACT XML Naming and Design Rules, Version 3.0 Draft of
187 November 16, 2009.

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

190 Copyright © 2009 UN/CEFACT, All Rights Reserved. UN liability, trademark and
191 document use rules apply.

192

193 2 XML Naming and Design Rules Project Team 194 Participants

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

198 **ATG2 Chair**

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

199 **Project Team Leader**

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

200 **Lead Editor**

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

201 **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	UK Payments Administration
Kevin Smith	Independent
Gunther Stuhec	SAP AG
Jim Wilson	KCX/CIDX

202 2.1 Acknowledgements

203 This version of UN/CEFACT - *XML Naming and Design Rules* Technical
204 Specification has been created to foster convergence among Standards
205 Development Organizations (SDOs). It has been developed in close coordination
206 with these organizations.

- 207 • ACORD

- 208 • CIDX
- 209 • GS1
- 210 • HR-XML
- 211 • OASIS Universal Business Language (UBL) Technical Committee
- 212 • Open Application Group (OAGi)

213 **2.2 Disclaimer**

214 The views and specification expressed in this technical specification are those of the
215 authors and are not necessarily those of their employers. The authors and their
216 employers specifically disclaim responsibility for any problems arising from correct or
217 incorrect implementation or use of this technical specification.

218 **2.3 Contact Information**

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

222 **3 Introduction**223 **3.1 Summary of Contents of Document**

224 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: Core Data Type Catalogue	Informative
Appendix H: Common Use Cases for Code Lists	Informative
Appendix I: Alternate Message Assembly	Informative
Appendix J: Date, Type, DateTime, Type and Time, Type Data Type Representations and Their Translation to XML Schema Types	Informative
Appendix K: Naming and Design Rules List	Normative
Appendix L: Glossary	Normative

225 3.1.1 Notation

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

230 Wherever **xsd:** appears in this specification it refers to a construct taken from one of
231 the W3C XML Schema recommendations. Wherever **cts:** appears it refers to a
232 construct taken from the *UN/CEFACT Core Components Technical Specification*.

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

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

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

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

241 When defining rules, the following annotations are used:

242 [] = optional

243 < > = variable

244 | = choice

245 3.2 Audience

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

- 248 • Members of the UN/CEFACT Applied Technologies Group who are
249 responsible for development and maintenance of UN/CEFACT XML
250 Schema
- 251 • The wider membership of the other UN/CEFACT Groups who participate
252 in the process of creating and maintaining UN/CEFACT XML Schema
253 definitions
- 254 • Designers of tools who need to specify the conversion of user input into
255 XML Schema definitions adhering to the rules defined in this document.
- 256 • Designers of XML Schema definitions outside of the UN/CEFACT Forum
257 community. These include designers from other standards organizations
258 and companies that have found these rules suitable for their own
259 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>

260 **4 Objectives**

261 **4.1 Goals of the Technical Specification**

262 This technical specification has been developed to provide for XML standards based
263 expressions of semantic data models representing business information exchanges.
264 It can be employed wherever business information is being shared in an open
265 environment using XML Schema to define the structure of business content. It
266 describes and specifies the rules and guidelines UN/CEFACT will use for developing
267 XML schema and schema documents based on Core Component Technical
268 Specification (CCTS) conformant artefacts and information models developed in
269 accordance with the UN/CEFACT Core Components Technical Specification Version
270 3.0.

271 **4.2 Requirements**

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

274 **4.3 Conformance**

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

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

284 Rules in categories 1, 4 and 5 cannot be modified. Rules in categories 2, 3, 6, and 7
285 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.		

286 4.4 Caveats and Assumptions

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

292 Although this specification defines schema components as expressions of CCTS
 293 artefacts, it can also be used by non-CCTS developers for other class based
 294 expressions of logical data models and information exchanges.

295 This specification does not address transformations via scripts or any other means. It
296 does not address any other representation of CCTS artefacts – such as OWL, Relax
297 NG, and XMI which are clearly outside the scope of this document.

298 **4.5 Guiding Principles**

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

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

337 5 XML Schema Architecture

338 This section defines general XML Schema construction including:

- 339 • Overall XML Schema Structure
- 340 • Relationship to CCTS
- 341 • Business Message Syntax Binding
- 342 • Naming and Modeling Constraints
- 343 • Reusability Scheme
- 344 • Namespace Scheme
- 345 • XML Schema Files
- 346 • Schema Location
- 347 • Versioning Scheme

348 5.1 Overall XML Schema Structure

349 UN/CEFACT has determined that the World Wide Web Consortium (W3C) XML
 350 Schema Recommendation is the schema definition language with the broadest
 351 adoption and tool support. Accordingly, all UN/CEFACT XML Schema definitions will
 352 be expressed in XML Schema. All references to W3C XML Schema will be as XML
 353 Schema. References to XML Schema defined by UN/CEFACT will be as
 354 UN/CEFACT XML Schema.

[R 8059]	All XML Schema design rules MUST be based on the W3C XML Schema 1.0 Recommendation: XML Schema Part 1: Structures Second Edition and XML Schema Part 2: Datatypes Second Edition .	1
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

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

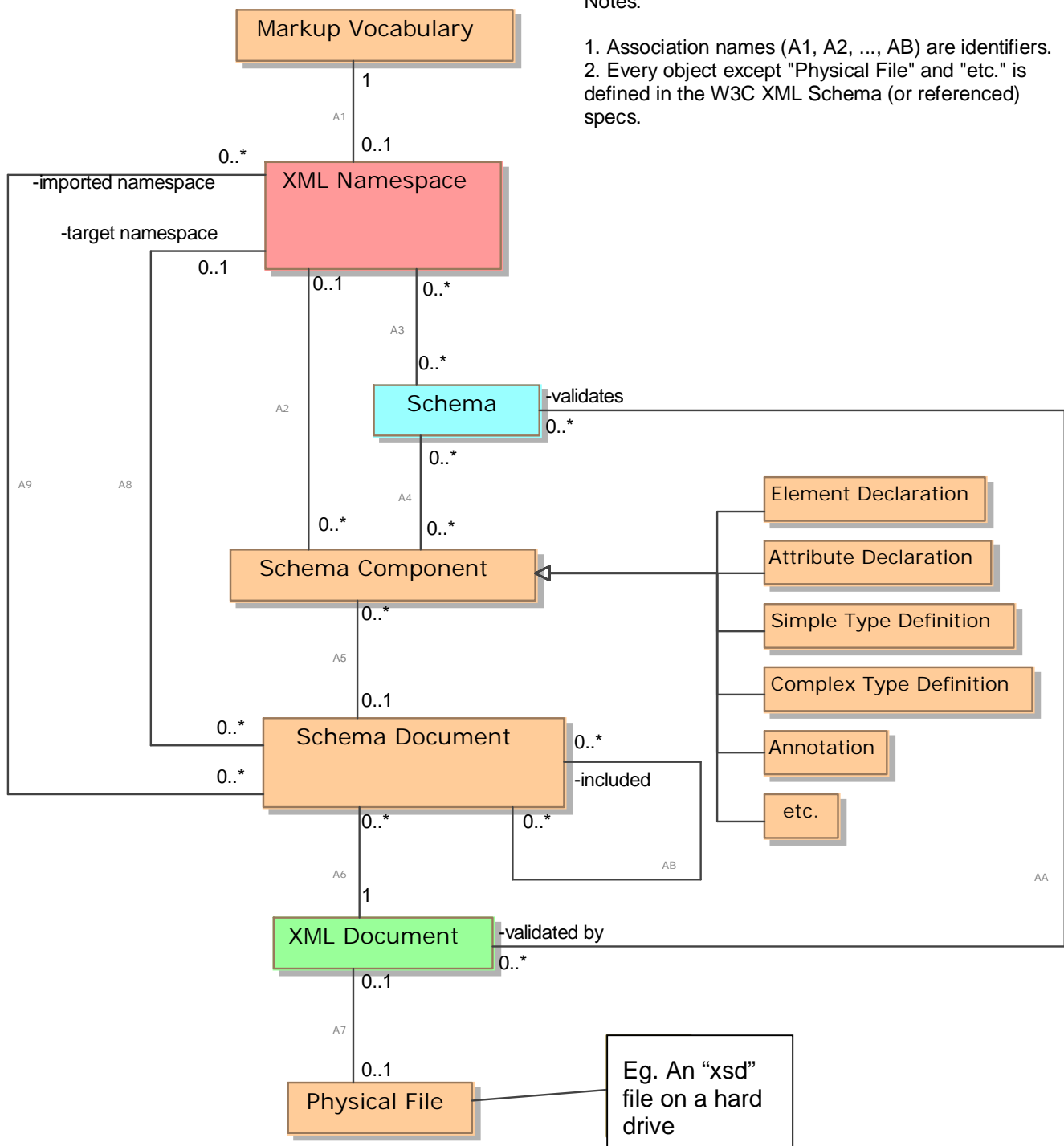
[R 935C]	All conformant XML instance documents MUST be based on the W3C suite of technical specifications holding recommendation status.	1
----------	---------------------------------------------------------------------------------------------------------------------------------	---

358 To maintain consistency in lexical form, all XML Schema need to use a standard
 359 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
----------	-------------------------------------------------------------------------------------------------------	---

360 The W3C XML Schema specification uses specific terms to define the various
 361 aspects of a W3C XML Schema. These terms and concepts are used without
 362 change in this NDR specification.

363 Figure 5-1, shows these terms and concepts and their relationship as defined by the
 364 W3C.



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

366 **5.2 Relationship to CCTS**

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

369 **5.2.1 CCTS**

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

372 The foundational concepts of CCTS are Core Components (CC) and Business
373 Information Entities (BIE). CCs are building blocks that can be used for all aspects of
374 data modeling, information modelling and information exchange. CCs are conceptual
375 models that are used to define Business Information Entities (BIEs).

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

- 380 • Be qualified to provide a unique business semantic,
- 381 • Specify a restriction of the underlying CC.

382 CCs include Aggregate Core Components (ACCs), Basic Core Components (BCCs)
383 and Association Core Components (ASCCs). BIEs include Aggregate Business
384 Information Entities (ABIEs), Basic Business Information Entities (BBIEs) and
385 Association Business Information Entities (ASBIEs).

386 The CCTS model for BIEs includes:

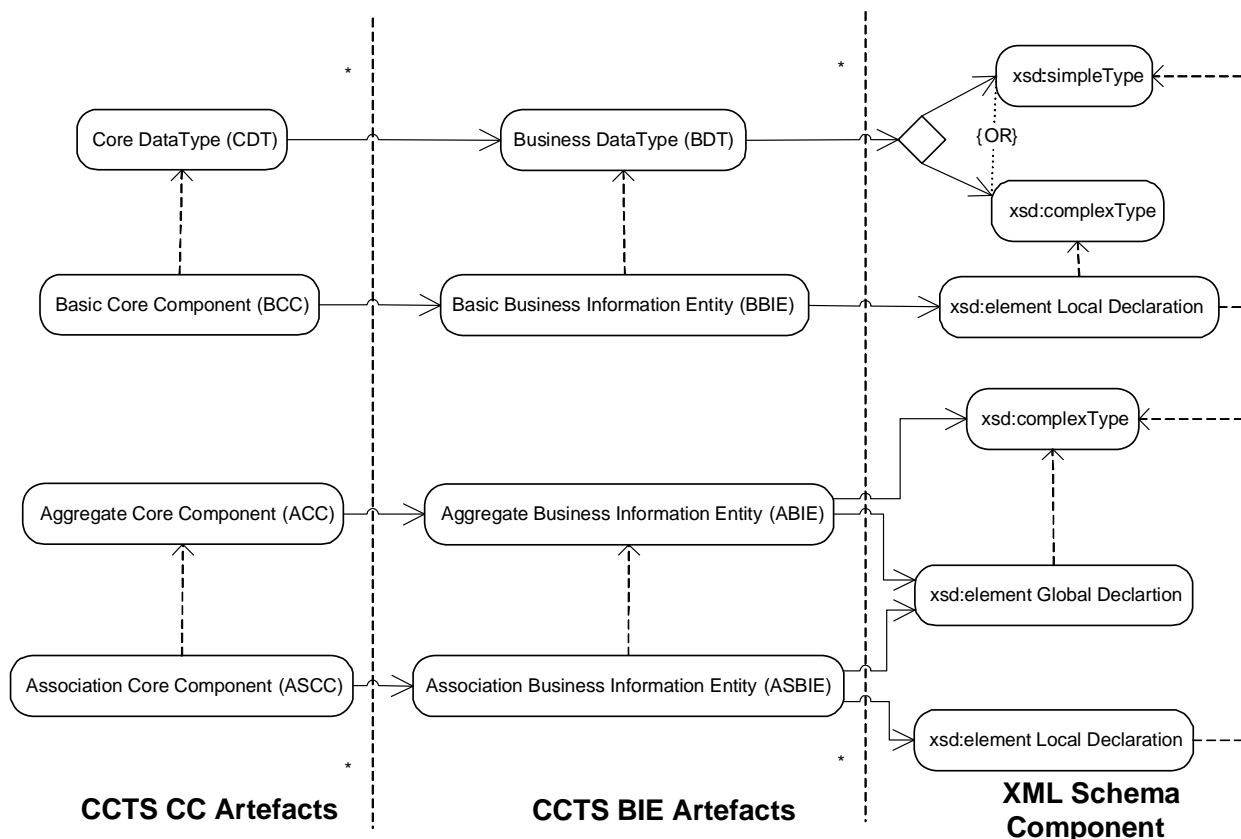
- 387 • Common Information – Information that is expressed in the annotation
388 documentation in the XML Schema.
- 389 • Localized Information – Information that while expressed in the model is not
390 expressed in the XML Schema.
- 391 • Usage Rules – Information that is expressed in the annotation application
392 information in the XML Schema.

393 **5.2.2 The XML Schema Components**

394 UN/CEFACT XML Schema design rules are closely coupled with CCTS. Thus,
395 UN/CEFACT XML Schema will be developed from fully conformant BIEs that are
396 based on fully conformant CCs. Figure 5-2 shows the relationship between relevant
397 CCTS CC artefacts, BIE artefacts and XML Schema components. The dotted arrows
398 within the CCTS CC Artefact column indicates that the given CC artefact makes use
399 of the CC artefact pointed to by the arrow. The solid arrows flowing from the CC to
400 the BIE column show the direct mapping of the artefacts from CC to BIEs as defined
401 by CCTS.

402 [Note:]

403 CCTS specifies Data Types, CCs and BIEs. The columns in Figure 5-2 represent the
404 conceptual CC model view, the logical BIE model view, and how these are
405 transformed into XML Schema binding.



406 **Figure 5-2 Transitions between CCTS artefacts and XML Schema Components**

407 The dotted arrows within the CCTS BIE Artefact column indicates that the given BIE
 408 artefact makes use of the BIE artefact pointed to by the arrow. The solid arrow
 409 flowing between the BIE column and the XML Schema component column show the
 410 direct mapping from the BIE to the XML Schema component used to represent it.

411 The dotted arrows within the XML Schema component column indicate that the given
 412 element makes use of the artefact type pointed to by the arrow.

413 **5.2.2.1 Aggregate Business Information Entity**

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

418 **5.2.2.2 Association Business Information Entity**

419 An ASBIE represents an association between the associating (parent) ABIE and the
 420 associated (child) ABIE. An ASBIE is represented as either a local or global element,
 421 depending upon the type of association (UML association
 422 `AggregationKind=shared` or `AggregationKind=composite`) specified in the
 423 model. An ASBIE will be declared as follows:

- 424 • If the ASBIE is a `composite` association (`AggregationKind=composite`),
- 425 the associated ASBIE is declared as a local element (`xsd:element`) within

426 the type (`xsd:complexType`) representing the associating ABIE. This local
427 element (`xsd:element`) is of the type (`xsd:complexType`) of the
428 associated ABIE.

- 429 • If the ASBIE is a `shared` association (`AggregationKind=shared`), the
430 ASBIE is referenced as a global element (`xsd:element`) within the type
431 representing the associating ABIE. The global element (`xsd:element`) is
432 declared in the same namespace as the associating ABIE and is of the type
433 (`xsd:complexType`) of the associated ABIE.

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

435 5.2.2.3 Basic Business Information Entity

436 A BBIE is declared as a local element within the `xsd:complexType` definition
437 representing the parent ABIE. The BBIE is of the `xsd:simpleType` or
438 `xsd:complexType` of its BDT. See section [8.3 Business Information Entities XML
439 Schema Files](#).

440 5.2.2.4 Business Data Type

441 A BDT represents the value domain of a BBIE. A BDT is defined as either an
442 `xsd:complexType` or `xsd:simpleType`. If the BDT value domain can be
443 expressed by the facets of an XML Schema built-in data type, then the BDT will be
444 defined as an `xsd:simpleType` whose base type is the XML Schema built-in type.

445 If the BDT requires a more robust expression, then the BDT will be defined as an
446 `xsd:complexType` whose content model fully defines its value domain.

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

448 5.2.3 Context Categories

449 CCTS identifies a set of context categories such as business process, geopolitical,
450 system capabilities, or business process role. The defined values of these categories
451 collectively express the context of specific BIEs. This NDR specification expresses
452 the context through the use of an annotation application information element
453 (`<xsd:annotation> <xsd:appInfo>`) accompanying each element declaration.

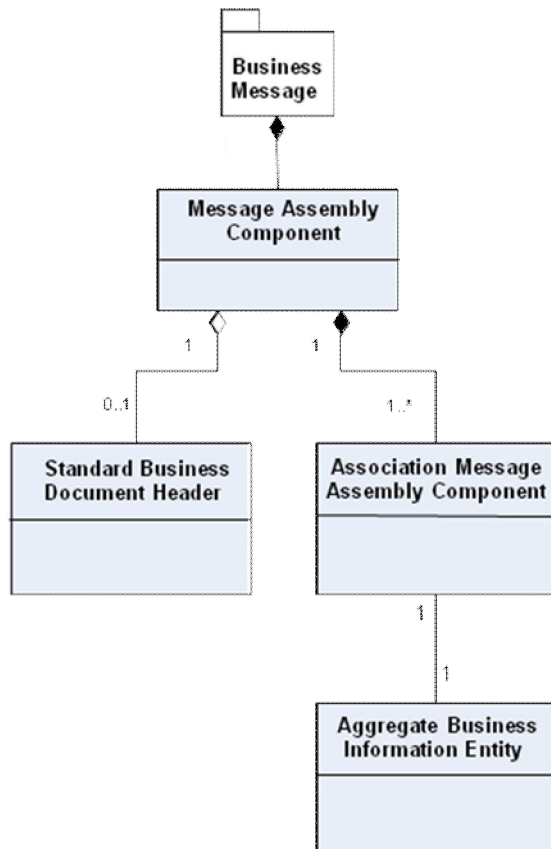
454 See section [6 Application of Context](#) and section [7.5.2 Application Information
455 \(AppInfo\)](#) for more information.

456 5.3 Business Message Syntax Binding

457 UN/CEFACT will create the XML syntax binding of its CCTS conformant BIE data
458 models directly from the associations and hierarchies expressed in the Business
459 Message Template (BMT) defined for each business message exchange. This
460 transformation approach is based on nesting of all components of the data model in
461 a Schema Modularity Model.

462 As shown in Figure 5-3, UN/CEFACT business message structures are defined
463 using the Business Message Template metamodel. The business message structure
464 consists of a single Message Assembly (MA) component representing the business

465 message. The MA consists of 1 or more Association Message Assembly
 466 Components and zero or one Standard Business Document Header (SBDH)
 467 Components. Each ASMA Component is a proxy for a first level ABIE in a given
 468 business message. The optional SBDH Component contains application specific
 469 information unique to the instance.



470

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

472 The MA component is defined as a named `xsd:complexType` and declared as the
 473 sole global element in a Root XML Schema File. The MA content model consists of a
 474 set of ASMAs that represent the first level ABIEs in a message. Each ASMA is
 475 manifested as an `xsd:element` and is either:

- 476 • Declared as a local element whose type is of an `xsd:complexType` defined
 477 in a BIE XML Schema File if the ASMA `aggregationKind=composite`.
- 478 • Referenced to a global element that exists in a BIE XML Schema File if its
 479 `aggregationKind=shared`.

480 UN/CEFACT will treat all ASMAs as `aggregationKind=composite`. See section
 481 [5.5 Reusability Scheme](#).

[R 8EC9]	UN/CEFACT MA <code>xsd:complexType</code> definitions MUST locally declare all ASMAs.	3
----------	---------------------------------------------------------------------------------------	---

482 The MA may also contain an optional Standard Business Document Header (SBDH)
 483 component. The SBDH component is manifested in the MA as an `xsd:element`
 484 reference to the root element declared in the SBDH Schema. The header of the
 485 SBDH schema, including XML declaration, copyright information, and XML Schema
 486 component is defined in accordance with the applicable rules in this NDR, and it is
 487 assigned to a unique namespace that is declared using the applicable rules in this
 488 specification for namespaces. The actual element declarations and type definitions in
 489 the SBDH component are in accordance with the names defined in the SBDH
 490 Technical Specification.

491 See section [8.2 Root XML Schema Files](#) and SBDH Technical Specification.

492 **5.3.1 Nesting Alternative**

493 The W3C XML Schema Specification also supports an alternative to nesting. This
 494 alternative – using schema identity constraints (`xsd:key/xsd:keyRef` – enables
 495 referencing and reuse of a given XML element in instance documents. UN/CEFACT
 496 is currently evaluating this alternative for future use; to include a method for
 497 application at the data model level. In anticipation that the data model issues will be
 498 resolved, UN/CEFACT has already developed a set of rules for this implementation.
 499 These rules and the supporting narrative can be found in [Appendix I Alternative](#)
 500 [Business Message Syntax Binding](#). Organizations using this alternative method will
 501 still be considered conformant to this specification, if they adhere to all other
 502 conformance requirements and use the rules defined in the [Appendix I Alternative](#)
 503 [Business Message Syntax Binding](#).

504 **5.4 Naming and Modeling Constraints**

505 UN/CEFACT XML Schemas are derived from components created through the
 506 application of CCTS. These schema contain XML Schema Components that follow
 507 the naming and design rules in this specification.

508 These naming and design rules take advantage of the features of the W3C XML
 509 Schema specification. In many cases this approach results in the truncation of the
 510 CCTS Dictionary Entry Names (DENs). However, the fully conformant CCTS DENs
 511 of the underlying CCTS artefacts are preserved as part of the annotation
 512 documentation (`<xsd:annotation> <xsd:documentation>`) element
 513 accompanying each element declaration.

514 The CCTS DEN can be reconstructed by using XPath expressions. The Fully
 515 Qualified XPath (FQXP) ties the information to its standardized CCTS semantics,
 516 while the XML element or attribute name is a truncation that reflects the hierarchy of
 517 the XML construct.

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

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

524 Example 5-1 shows the FQXP for BIEs Address. Latitude_ Coordinate. Measure and
 525 Organization. Location. Name.

526 **Example 5-1: Fully Qualified XPath**

```
527 Address/LatitudeCoordinate/Measure
528 Organisation/Location/Name
```

529 The official language for UN/CEFACT is English. All official XML constructs
 530 published by UN/CEFACT will be in English. XML and XML Schema development
 531 work may very well occur in other languages, however official submissions for
 532 inclusion in the UN/CEFACT XML Schema library must be in English. Other
 533 language translations of UN/CEFACT published XML Instances and XML Schema
 534 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
----------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

535 LowerCamelCase (LCC) is used for naming XML Schema attributes and
 536 UpperCamelCase (UCC) is used for naming XML Schema elements and types.
 537 LowerCamelCase capitalizes the first character of each word except the first word
 538 and compounds the name. UpperCamelCase capitalizes the first character of each
 539 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

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

541 **Example 5-2: Attribute**

542 Allowed

```
543 <xsd:attribute name="timeZoneCode" .../>
```

544 **Example 5-3: Element**

545 Allowed

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

547

548 **Example 5-4: Type**

549 Allowed

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

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

552 Allowed - Singular:

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

554 Not Allowed - Plural:

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

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

557 Not Allowed

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

559 While CCTS allows for the use of periods, spaces and underscores in the dictionary
560 entry name. XML best practice is to not include these characters in an XML tag
561 name.

[R AB19]	XML element, attribute and type names constructed from dictionary entry names MUST only use lowercase alphabetic characters [a-z], uppercase alphabetic characters [A-Z], digit characters [0-9] or the underscore character [_] as allowed by W3C XML 1.0 for XML names.	1
----------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

562 Additionally, XML 1.0 specifically prohibits the use of certain reserved characters in
563 XML tag names.

[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
----------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

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

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

566 Not Allowed

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

568

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

570 Allowed – ID is an approved abbreviation

571

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

572 Not Allowed – Cd is not an approved abbreviation, if it was an approved abbreviation
573 it must appear in all upper case574

```
<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

575 **5.5 Reusability Scheme**576 UN/CEFACT is committed to an object based approach for its process, data, and
577 information models.

578 UN/CEFACT considered adopting an XML Schema type based approach which uses
579 named types, a type and element based approach, or an element based approach. A
580 type based approach for XML management provides the closest alignment with the
581 process modelling methodology described in UMM. Type information is beginning to
582 be accessible when processing XML instance documents. Post schema-validation
583 infoSet (PSVI) capabilities are beginning to emerge that support this approach, such
584 as *data-binding* software that compiles schema into ready-to-use object classes and
585 is capable of manipulating XML data based on their types.

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

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

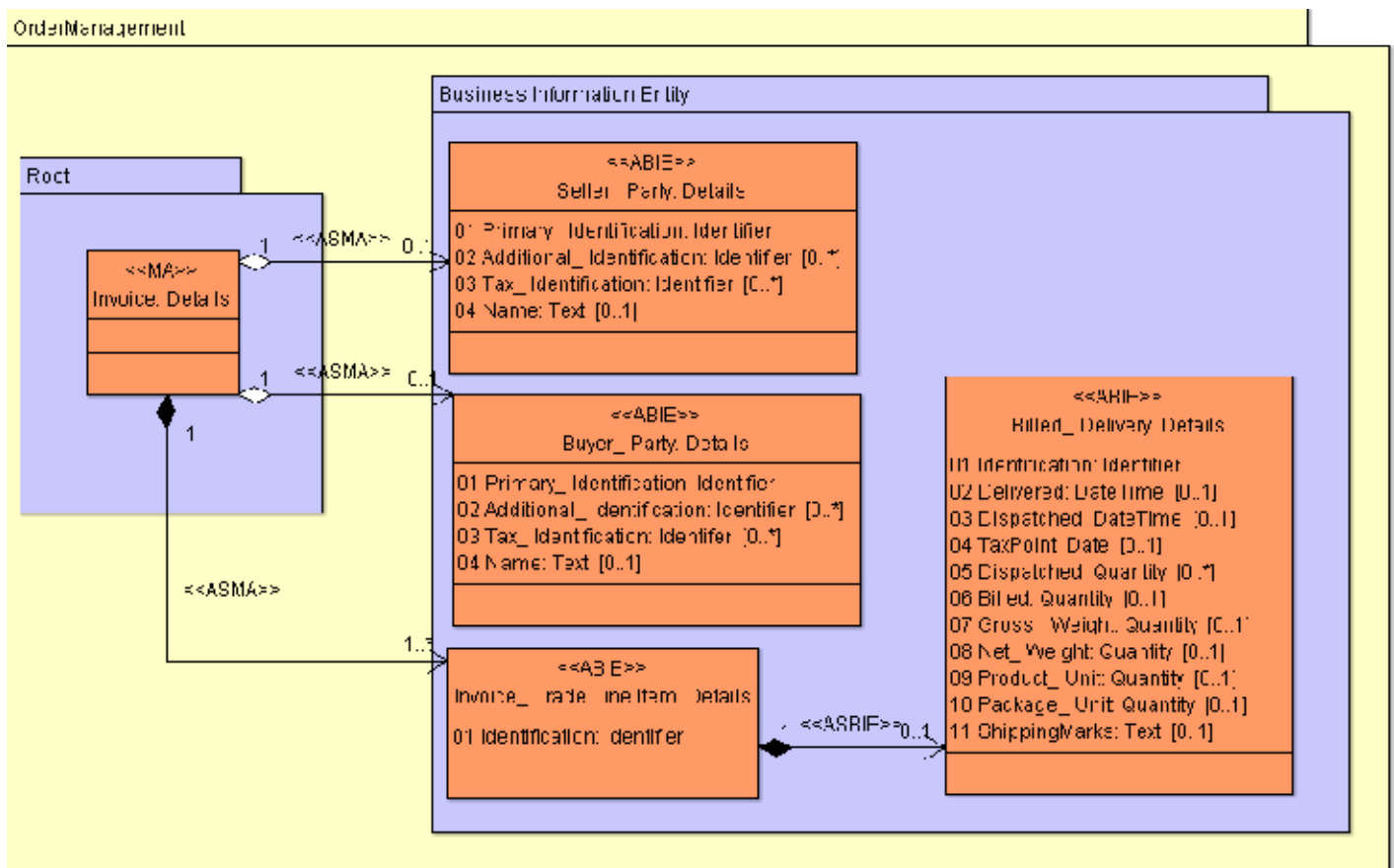
594 A purely type based approach does, however, limit the ability to reuse elements,
595 especially in technologies such as Web Services Description Language (WSDL).

596 For these reasons, UN/CEFACT implements a *hybrid approach* that provides
597 benefits over a pure type based approach. Most significantly it increases reusability
598 of library content both at the modelling and XML Schema level.

599 The key principles of the *hybrid approach* are:

- 600 • All classes (Invoice, Seller_Party, Buyer_Party, Invoice_Trade.Line.Item and
601 Billed_Delivery in Figure 5-4) are declared as an `xsd:complexType`.

- 602 • All simple attributes of a class (BBIEs) are declared as a local `xsd:element`
 603 within the corresponding named `xsd:complexType`.
- 604 • ASBIEs whose UML `aggregationKind=composite` are declared as a
 605 local `xsd:element` with a named `xsd:complexType` (e.g.
 606 Invoice_Trade.Line.Item and Billed_Delivery in Figure 5-4). A composite
 607 aggregation ASBIE represents a relationship wherein if the associating ABIE
 608 ceases to exist the associated ABIE ceases to exist.
- 609 • ASBIEs whose UML `aggregationKind=shared` are declared as a global
 610 `xsd:element` with a named `xsd:complexType` (e.g. Invoice. Buyer.
 611 Buyer_Party and Invoice. Seller. Seller Party in Figure 5-4). A shared
 612 aggregation ASBIE represents a relationship wherein if the associating ABIE
 613 ceases to exist, the associated ABIE continues to exist.
- 614 The rules pertaining to the *hybrid approach* are contained in sections [8.3.3 Type](#)
 615 [Definitions](#) and [8.3.4 Element Declarations and References](#).



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

617 Figure 5-4 shows an example UML model. Example 5-9 shows the resulting XML
 618 Schema declaration (devoid of `<xsd:annotation>` and `<xsd:comments>`) that
 619 results directly from the translation of the UML to XML Schema following the rules
 620 defined in this specification.

621 [Note] - Tokens

622 The tokens rsm, bie, bdt, xbt, bcl, ccl, bis, and cis are used throughout this document
 623 to generically represent Root XML Schema Files, BIE XML Schema Files, BDT XML
 624 Schema Files, XML Schema Business Type XML Schema File, Business Code List
 625 XML Schema Files, Common Code List XML Schema Files, Business Identifier
 626 Schema XML Schema Files and Common Identifier Schema XML Schema Files.
 627 The actual tokens are developed using the rules stated elsewhere in this
 628 specification.

629 **Example 5-9: XML Schema declarations representing Figure 5-4.**630 **Invoice - Root XML Schema File**

```
631 <xsd:schema targetNamespace="urn:un:unece:uncefact:data:invoice:l:draft">
632
633   <xsd:include "BusinessInformationEntity.xsd"
634
635   <xsd:element name="Invoice" type="rsm:InvoiceType"/>
636   <xsd:complexType name="InvoiceType">
637     <xsd:sequence>
638       <xsd:element ref="bie:SellerParty"/>
639       <xsd:element ref="bie:BuyerParty"/>
640       <xsd:element name="InvoiceTradeLineItem" type="bie:InvoiceTradeLineItemType"
641         maxOccurs="unbounded" />
642     </xsd:sequence>
643   </xsd:complexType>
644 </xsd:schema>
```

645 **Business Information Entity XML Schema File**

```
646 <xsd:schema targetNamespace="urn:un:unece:uncefact:data:invoice:l:draft">
647
648   <xsd:element name="BuyerParty" type="BuyerPartyType"/>
649   <xsd:element name="SellerParty" type="SellerPartyType"/>
650   <xsd:element name="InvoiceTradeLineItem" type="InvoiceTradeLineItemType"/>
651   <xsd:element name="BilledDelivery" type="BilledDeliveryType"/>
652
653   <xsd:complexType name="BuyerPartyType">
654     <xsd:sequence>
655       <xsd:element name="ID" type="IDType"/>
656       <xsd:element name="Name" type="NameType"/>
657     </xsd:sequence>
658   </xsd:complexType>
659
660   <xsd:complexType name="SellerPartyType">
661     <xsd:sequence>
662       <xsd:element name="ID" type="IDType"/>
663       <xsd:element name="GivenName" type="NameType"/>
664       <xsd:element name="Surname" type="NameType"/>
665     </xsd:sequence>
666   </xsd:complexType>
667
668   <xsd:complexType name="InvoiceTradeLineItemType">
669     <xsd:sequence>
670       <xsd:element name="ID" type="IDType"/>
671       <xsd:element name="BilledDelivery" type="bie:BilledDeliveryType"/>
672     </xsd:sequence>
673   </xsd:complexType>
674
675   <xsd:complexType name="BilledDeliveryType">
676     <xsd:sequence>
677       <xsd:element name="ID" type="IDType"/>
678       <xsd:element name="Name" type="NameType"/>
679     </xsd:sequence>
680   </xsd:complexType>
681 </xsd:schema>
```

683 **5.6 Namespace Scheme**

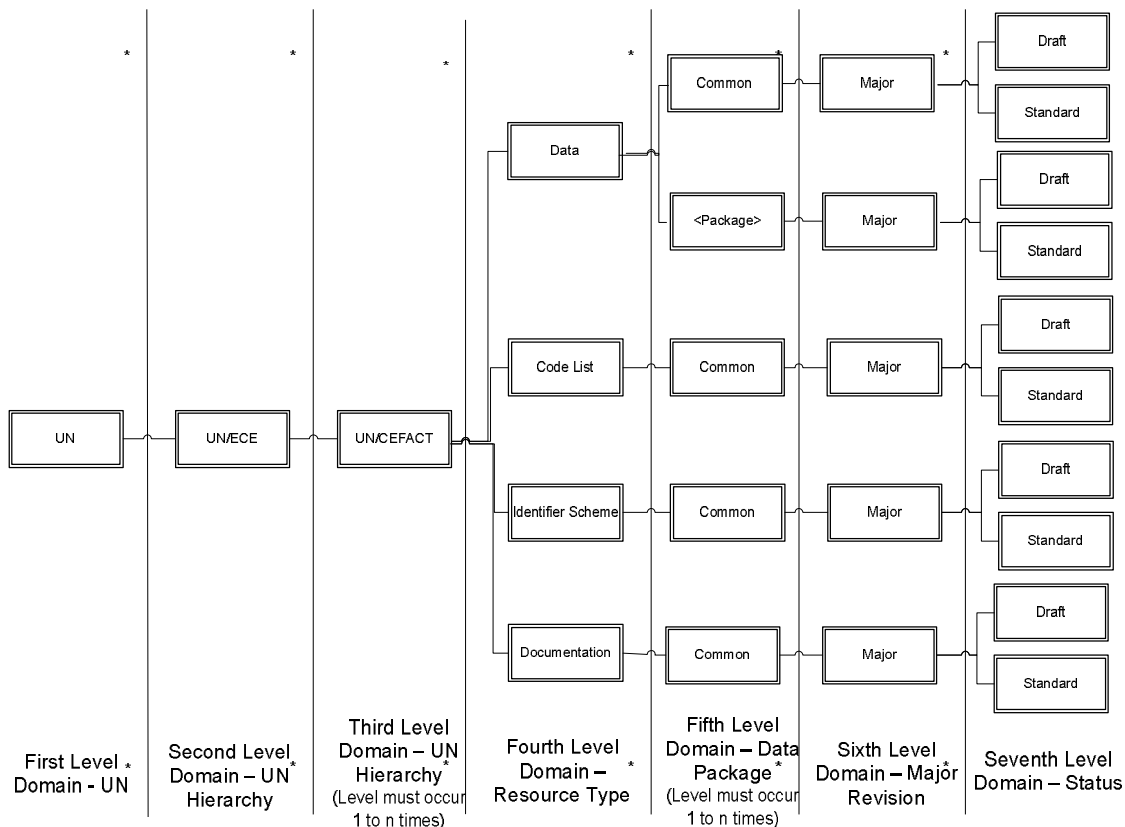
684 A namespace is an abstract container for a collection of elements, attributes and
 685 types that serve to uniquely identify one such collection from all other collections.

686 “An XML namespace is identified by a URI reference [RFC3986]; element and
 687 attribute names may be placed in an XML namespace...”²

688 UNCEFACT assigns XML artefacts to UNCEFACT namespaces following the
 689 namespace scheme shown in Figure 5-5.

690 Each organization that intends to adhere to this specification will assign their XML
 691 Schema defined content in a namespace that follows a scheme similar to the
 692 UN/CEFACT namespace scheme shown in Figure 5-5. This scheme will reflect the
 693 hierarchy of the organization and the package structure of the CCTS compliant
 694 model from which the XML Schema is derived.

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



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

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

696 [Note:]
 697 Both the organizational hierarchy and the package structure of a CCTS v3.0
 698 compliant model is reflected in the namespace in the resulting set of XML Schema
 699 Files.
 700 [Note:]
 701 The third level organizational hierarchy level may occur zero to n times.
 702 [Note:]
 703 The fifth level package level may occur one to n times to reflect the structure of the
 704 package hierarchy.

705 5.6.1 Namespace Uniform Resource Identifiers

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

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

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

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

719 The URN format will be:

```
720 urn:<organization>:<organization
721 hierarchy>[:<organization hierarchy level>]*:<schema
722 type>>[:<package>]+:<major>:<status>
```

723 The URL namespace structure will follow the provisions of Internet Engineering Task
 724 Force (IETF) Request for Comments (RFC) 1738 – Uniform Resource Locators
 725 (URL).

726 The URL format will be:

```
727 http://<organization>/<organization
728 hierarchy>[/<organization hierarchy level>]*/<schema
729 type>[/<package>]+/<major> /<status>
```

730 Where:

- 731 • organization – An identifier of the organization providing the standard.
- 732 • organization hierarchy – The first level of the hierarchy within the organization
733 providing the standard.

- 734 • organization hierarchy level – Zero to n level hierarchy of the organization
- 735 providing the standard.
- 736 • schema type – A token identifying the type of schema module:
- 737 **data | codelist | identifierscheme | documentation.**
- 738 • package – One to n level of the packages expressed in the associated CCTS
- 739 v3.0 complaint model in which the XML Schema Files expressed. Additionally,
- 740 a **common** location is used by each of the schema types for common content.
- 741 • major – The major version number.
- 742 • status – The status of the schema as: **draft | standard.**

[R 8E2D]	<p>The XML Schema namespaces MUST use the following pattern:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 5px;">URN:</td> <td style="padding: 5px;"><code>urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:<schema type>[:<package>]+:<major>:<status></code></td> </tr> <tr> <td style="padding: 5px;">URL:</td> <td style="padding: 5px;"><code>http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/<schema type>[/<package>]+/<major>/<status></code></td> </tr> </table>	URN:	<code>urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:<schema type>[:<package>]+:<major>:<status></code>	URL:	<code>http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/<schema type>[/<package>]+/<major>/<status></code>	3
	URN:	<code>urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:<schema type>[:<package>]+:<major>:<status></code>				
URL:	<code>http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/<schema type>[/<package>]+/<major>/<status></code>					
<p>Where:</p> <ul style="list-style-type: none"> • organization – An identifier of the organization providing the standard. • organization hierarchy – The first level of the hierarchy within the organization providing the standard. • organization hierarchy level – Zero to n level hierarchy of the organization providing the standard. • schematype – A token identifying the type of schema module: data codelist identifierscheme documentation. • package – One to n level of the packages expressed in the associated CCTS v3.0 complaint model in which the XML Schema Files expressed. Additionally, a common location is used by each of the schema types for common content. • major – The major version number. • status – The status of the schema as: draft standard. 						

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

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

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

747

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

749 `"urn:un:unece:unefact:data:odermanagement:l:standard"`

750 UN/CEFACT namespace names include a major version identifier, therefore once a
 751 namespace’s content is published; any change that breaks backward compatibility
 752 requires a new namespace. See the section on [5.9.1 Major Versions](#). Only the
 753 publisher of a namespace may change the content defined within the namespace.
 754 The publisher may only make changes that adhere to the rules defined for minor
 755 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
----------	--------------------------------------------------------------------------------------------------------------------	---

756 **5.6.2 Namespace Tokens**

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

760 Namespace tokens representing the namespace will be created using three
 761 character representations for each unique namespace.

762 Additionally, XML Schema Files that are defined for Common Code List will use a
 763 token that is prefixed with `c1m` to indicate that they are Common Code List XML
 764 Schema Files.

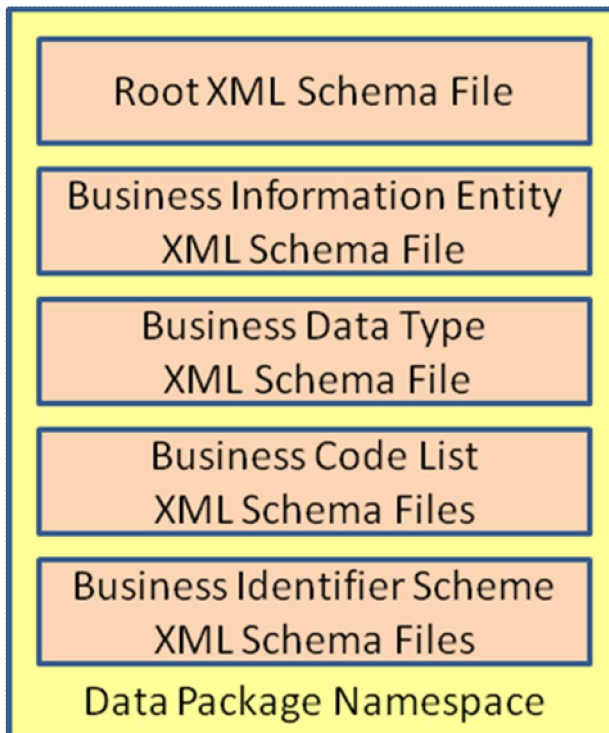
765 **5.7 XML Schema Files**

766 An XML Schema File is a schema document realized as a physical file. As defined
 767 by the W3C, a schema document represents relevant instantiations of the thirteen
 768 defined W3C XML Schema Components that collectively comprise an abstract data
 769 model. For consistency, XML Schema File names will adhere to a specific pattern.

[R 92B8]	<p>The XML Schema File name for files other than code lists and identifier schemes MUST be of the form: <code><Schema Module Name>_<Version Identifier>.xsd</code>, with periods, spaces, other separators and the words <code>XML Schema File</code> removed.</p> <p>Where:</p> <ul style="list-style-type: none"> • Schema Module Name – Is the name of the Schema Module. • Version Identifier – Is the major and minor version identifier. 	3
[R 8D58]	When representing versioning schemes in file names, the period MUST be represented by a lowercase <code>p</code> .	3

770 XML Schema Files can be either unique in their functionality, or represent splitting of
 771 larger XML Schema Files for performance or manageability enhancement. A well
 772 thought out approach to the layout provides an efficient and effective mechanism for
 773 providing components as needed rather than dealing with complex, multi-focused

774 XML Schema Files. XML Schema Files created from this specification represent
 775 abstract data models for messages, CCTS conformant BIEs BDTs, Business Code
 776 Lists (BCL), Business Identifier Schemes (BIS), references to Common Code Lists
 777 (CCL), Common Identifier Schemes (CIS) and to Common XML Schema Built-in
 778 Type Extensions (XBTs).



779

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

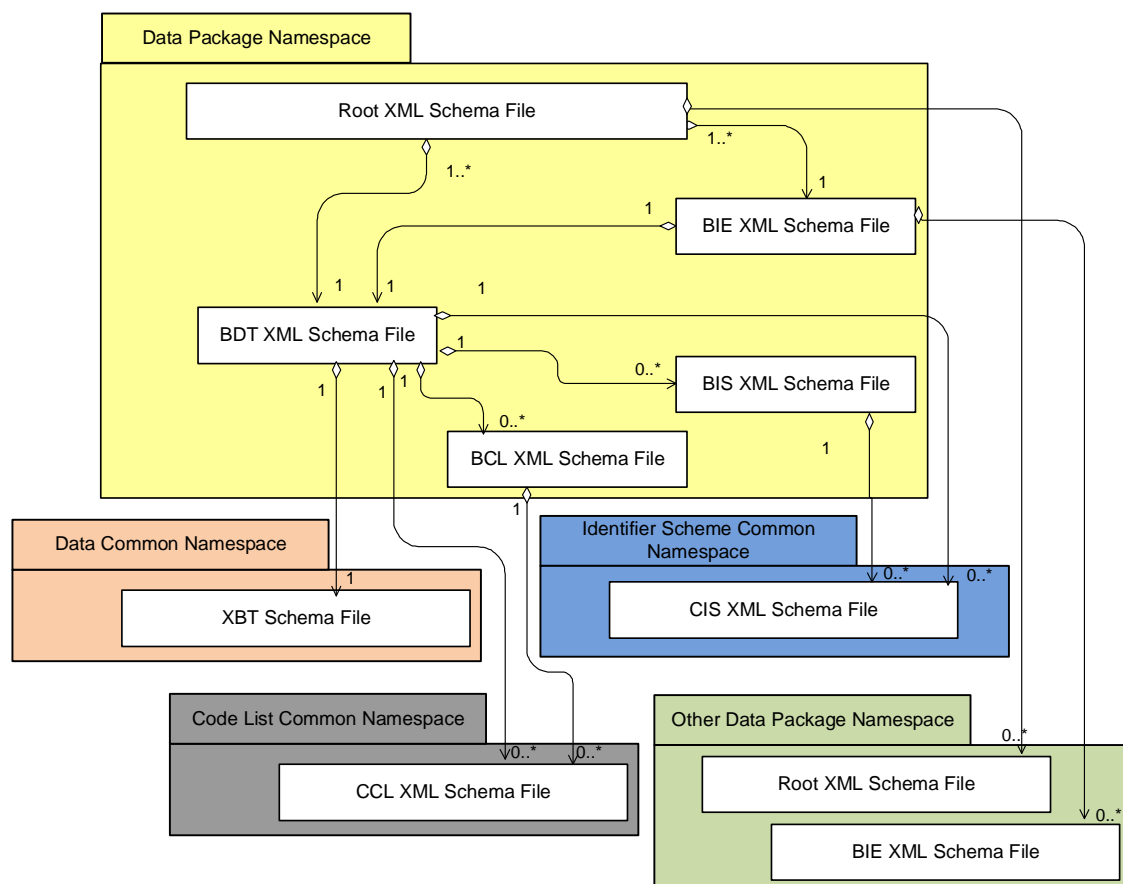
781 Figure 5-6 shows the XML Schema Files that are collected into relevant namespaces
 782 representing business processes/information messages. Figure 5-6 does not show
 783 the common XML Schema Files CCL, CIS and XBT; each of which are defined in
 784 different namespaces. Dependencies exist among the various files as shown in
 785 Figure 5-7.

786 See section [8 XML Schema Files](#) and the corresponding sub-sections.

787 [Note:]

788 By sharing common components between packages the model developer must be
 789 aware that changes reflected in one package or context are reflected in every
 790 package or context that use the shared component, whether intended or not.

791 Each of the Root XML Schema Files defined within the given package namespace
 792 hierarchy always contains `xsd:include` statements for both the BIE XML Schema
 793 File and the BDT XML Schema File assigned to the same namespace. Each Root
 794 XML Schema File may also contain zero or more `xsd:import` statements for Root
 795 XML Schema Files that are imported from other package namespaces that contain
 796 components that are reused.



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

798 The BIE XML Schema File always contains an `xsd:include` statement for the BDT
 799 XML Schema File that are assigned to the same namespace. The BIE XML Schema
 800 File may also contain zero or more `xsd:import` statements for BIE XML Schema
 801 Files that are imported from other package namespaces that contain components
 802 that are reused.

803 The BDT XML Schema File may have zero or more `xsd:include` statements for
 804 BCL XML Schema Files and BIS XML Schema Files that are assigned to the same
 805 namespace. The BDT XML Schema File also always has an `xsd:import` statement
 806 for the one XML Schema Built-in Type Extension XML Schema File. The BDT XML
 807 Schema File may also have zero or more `xsd:import` statements for each BDT
 808 required CCL XML Schema File and CIS XML Schema File.

809 The BCL XML Schema Files may contain an `xsd:import` statement for a CCL XML
 810 Schema File if it restricts the list of allowed common codes .

811 Each `xsd:schema` element used to define an XML Schema Document within an
 812 XML Schema File will have the namespace declared using
 813 `xsd:targetNamespace`.

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

814 The contents of the set of XML Schema within a given namespace are so
 815 interrelated that proper management dictates that versioning of all members of the
 816 set be synchronized, so that incompatible definitions are avoided. All schemas of the
 817 set, which are already assigned a single namespace version, are additionally
 818 assigned to a single file version number.

[R 9C85]	Every XML Schema File within a single namespace version MUST also be assigned to a single file version number.	1
----------	----------------------------------------------------------------------------------------------------------------	---

819 **5.7.1 Root XML Schema Files**

820 As expressed in section [5.6 Namespace Scheme](#), Root XML Schema Files are
 821 assigned to a namespace that reflect the data package value of the schema as
 822 shown in Figure 5-5. The determination of the data package value is at the discretion
 823 of the originating organization. The data package can be hierarchical within a set of
 824 related data packages in one or more business processes. However, the hierarchy
 825 should be kept as simple as possible. It is better to have multiple data packages at
 826 the same level in a hierarchy than to create deeply nested package hierarchies. This
 827 approach enables the use of individual package focused Root XML Schema Files
 828 that only use a tight set of related components without importing the entire library.
 829 Each Root XML Schema File will define its own dependencies.

830 The XML Schema File modularity scheme also calls for a set of XML Schema Files
 831 that support a Root XML Schema File. This set of XML Schema Files is also
 832 assigned to the same data package namespace.

833 There maybe a number of UN/CEFACT Root XML Schema Files, each of which
 834 expresses a separate business information payload. The Root XML Schema Files
 835 include the recognized business transactions for the data package namespace.

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

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

[R B3E4]	Each Root XML Schema File MUST be named in the Header comment of the file after the <code><BusinessInformationPayload></code> that is expressed in the XML Schema File by using the value of the <code><BusinessInformationPayload></code> followed by the words XML Schema File .	1
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

840 As defined in Section 5.3, each root XML Schema File will only contain a single MA
 841 Component consisting of one or more ASMA components and an optional SBDH
 842 Component. The Root XML Schema File will not duplicate reusable XML constructs
 843 available in the other XML Schema Files in the same namespace. Instead, the root
 844 XML Schema File uses the `xsd:include` and `xsd:import` features of XML
 845 Schema to access these constructs.

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

846 5.7.2 Business Information Entity XML Schema Files

847 A BIE XML Schema File will be created to define all reusable BIEs within a package
848 namespace.

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

[R 8238]	A BIE XML Schema File MUST be created within each namespace that is defined for a package.	1
[R 8252]	The BIE XML Schema Files MUST be named <i>Business Information Entity XML Schema File</i> by placing the name within the Header documentation section of the file.	1

851 Where desired, these BIE XML Schema Files may be further compressed for runtime
852 performance considerations, if necessary. This is accomplished through the creation
853 of a runtime version that only:

- 854 1) Contains those ABIEs necessary to support the including Root XML Schema
855 File, and
- 856 2) Reducing the `xsd:annotation` `xsd:documentation` elements.

857 5.7.3 Business Data Type XML Schema Files

858 The CCTS BDT value domain defines the value domain for a BBIE. The BDT value
859 domain is defined by selecting from one of the allowed primitives or scheme or list
860 and providing additional restrictions if desired through the use of supplementary
861 components which themselves have a primitive or a business scheme or list.

862 For reference purposes, UN/CEFACT publishes a Reference BDT XML Schema File
863 that consists of BDTs derived from CDTs using default value domains. This schema
864 file resides in the data common namespace and is used for reference purposes or as
865 a template for users desiring to create BDTs.

[R A2F0]	A Reference BDT XML Schema File MUST be created in the data common namespace to represent the set of unrestricted BDTs using default value domains.	1
----------	-----------------------------------------------------------------------------------------------------------------------------------------------------	---

866 An additional BDT XML Schema File that contains only the BDTs used in a package
867 namespace will also be published as part of the schema set for each package
868 namespace.

[R AA56]	A BDT XML Schema File MUST be created within each package namespace.	1
----------	----------------------------------------------------------------------	---

[R 847C]	The BDT XML Schema Files MUST be named <i>Business Data Type XML Schema File</i> by placing the name within the header documentation section of the file.	1
----------	-----------------------------------------------------------------------------------------------------------------------------------------------------------	---

869 5.7.4 XML Schema Built-in Type Extension XML Schema File

870 Not all CDTs defined in the *UN/CEFACT Data Type Catalogue Version 3.0* can be
 871 expressed in XML using the built-in types from XSD part 2. Some additional type
 872 definitions are necessary. The XML Schema Built-in Type Extension (XBT) XML
 873 Schema File defines these additional `xsd:simpleTypes` and `xsd:complexType`s. This
 874 XML Schema File resides in the data common namespace. It is included by the
 875 Reference BDT XML Schema File. It is imported by each data package specific BDT
 876 XML Schema File.

[R 9CDD]	An XBT XML Schema File MUST be created in the data common namespace to represent the additional types not defined by XML Schema that are needed to implement the BDT equivalents of the CDTs defined in the <i>UN/CEFACT Data Type Catalogue Version 3.0</i> .	1
[R 96ED]	The XBT XML Schema Files MUST be named <i>CCTS XML Builtin Types XML Schema File</i> by placing the name within the header documentation section of the file.	1

877 5.7.5 Code List XML Schema Files

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

- 881 • Unrestricted by an implementation packaging scheme, defined outside of any
 882 implementation packaging scheme. These are expressed as a CCL XML
 883 Schema File.
- 884 • Defined by an implementation packaging scheme. These are expressed as a
 885 BCL XML Schema File.

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

[R 8A68]	A Code List XML Schema File MUST be created to convey code list enumerations for each code list being used.	1
----------	-------------------------------------------------------------------------------------------------------------	---

[R B443]	<p>A Code List XML Schema File MUST be given a file name that represents the name of the code list and is unique within the namespace to which it belongs using the form:</p> <p><Code List Agency Identifier>_<Code List Identifier>_<Code List Version Identifier>.xsd</p> <p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Identifier – Identifies the agency that maintains the code list. • Code List Identifier – Identifies a list of the respective corresponding codes. • Code List Version Identifier – Identifies the version of the code list. 	1
[R B0AD]	<p>The semantic name of each Code List XML Schema File as defined in the comment section within the XML Schema File MUST be of the form:</p> <p><Code List Agency Name> <Code List Name> - Code List XML Schema File</p> <p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Name – Agency that maintains the code list. • Code List Name – The name of the code list as assigned by the agency that maintains the code list. 	1

891 Example 5-12 shows an example of the CCL file name.

892 **Example 5-12: File Name of UN/CEFACT Character Set Encoding Code XML Schema File Name**

893 `6_0133_40106.xsd`
 894 Where:
 895 6 = Code list Agency Identifier for UN/CEFACT from UN/CEFACT DE 3055 Codes for the
 896 Identification of Agencies
 897 0133= Code list identifier
 898 40106 = Code list version Identifier

899 Example 5-13 shows an example of the CCL semantic name.

900 **Example 5-13: Semantic Name of UN/CEFACT Security Type Code List XML Schema File**

901 `UN/CEFACT Security Initiative Document Security Code - Code List XML Schema File`

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

904 **5.7.5.1 Common Code List XML Schema Files**

905 A code list is considered common if it is published by a recognized standards
 906 organization for use across a broad spectrum of contexts. UN/CEFACT will prepare

907 a CCL XML Schema File for each common code list used by a BDT. Each CCL XML
 908 Schema File will 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
----------	---------------------------------------------------------------------------------------------------------	---

909 **5.7.5.2 Business Code List XML Schema Files**

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

916 Each BCL XML Schema File contains enumerated values for codes and their code
 917 values. These enumerated values may be a part of a restriction of a CCL, as a new
 918 code list for the given business context, 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 only used within the package namespace. 	1
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

919 **5.7.6 Identifier Schemes**

920 Identifier schemes are different than code lists in both concept and functionality.
 921 Whereas a code has a value, an identifier is a pointer that is typically devoid of any
 922 specific value. Code lists are enumerated lists. Identifier schemes are the equivalent
 923 of a pattern facet and are not enumerated. A scheme formally expresses a pattern
 924 facet.

925 Identifier schemes will be defined as simple types without enumeration in an
 926 Identifier Scheme XML Schema File following the same approach as is used for
 927 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 AD8C]	<p>An Identifier Scheme XML Schema File MUST be given a file name that represents the name of the Identifier Scheme and is unique within the namespace to which it belongs using the form:</p> <p><Identifier Scheme Agency Identifier>_<Identifier Scheme Identifier>_<Identifier Scheme Version Identifier>.xsd</p> <p>Where:</p> <ul style="list-style-type: none"> • Identifier Scheme Agency Identifier – Identifies the agency that maintains the identifier scheme. • Identifier Scheme Identifier – Identifies the scheme. • Identifier Scheme Version Identifier – Identifier of the version of the identifier scheme. 	1
[R A154]	<p>The semantic name of each Identifier Scheme XML Schema File as defined in the comment section within the XML Schema File MUST be of the form:</p> <p><Identifier Scheme Agency Name> < Identifier Scheme Name> - Identifier Scheme XML Schema File</p> <p>Where:</p> <ul style="list-style-type: none"> • Identifier Scheme Agency Name – Agency that maintains the identifier scheme. • Identifier Scheme Name – The name of the identifier scheme as assigned by the agency that maintains the identifier scheme. 	1

928 Example 5-14 shows an example of the United States Social Security Number
929 Identifier expressed as the file name of the Identifier Scheme XML Schema File.

930 **Example 5-14: File Name of United States Social Security Number Identifier Scheme XML**
931 **Schema File Name**

932 `244_SSN_1933.xsd`
933 Where:
934 `244 = Code List Agency Identifier for the US Department of Health and Human`
935 `Services which manages the US Social Security Number Identifier Scheme`
936 `SSN = Identifier Scheme Unique ID`
937 `1933 = Identifier Scheme version Identifier`

938 Example 5-15 shows an example of using the CIS Name to name the Identifier
939 Scheme XML Schema File.

940 **Example 5-15: Semantic Name of GS1 Global Trade Item Number Identifier Scheme XML**
941 **Schema File Name**

942 `GS1 Global Trade Item Number - Identifier Scheme XML Schema File`
943 where:
944 `GS1 = Agency Name`
945 `Global Trade Item Number = Identifier Scheme Name for GTIN (Global Trade Item`
946 `Number)`

947 **5.7.6.1 Common Identifier Scheme**

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

951 **5.7.6.2 Business Identifier Scheme**

952 A business scheme may be defined for a BDT. In cases where some identifiers
 953 allowed by the source CIS are not needed in the business process, the BIS will be a
 954 restriction to the CIS. All BISs are expressed as individual XML Schema Files and
 955 are assigned to the same namespace as the XML Schema Files that make use of
 956 them. If a BDT that references a BIS is used in different namespaces, then a BDT
 957 will be defined and a BIS 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
----------	------------------------------------------------------------------------------------------------------	---

958 Each Business Identifier Scheme XML Schema File contains metadata regarding the
 959 scheme. If a business scheme is a restriction on a common scheme, the nature of
 960 the restriction will be included in the metadata as a business rule in an
 961 `xsd:annotation 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
----------	--------------------------------------------------------------------------------------------------------------------------	---

962 **5.7.7 Other Standard Bodies BIE XML Schema Files**

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

969 In order to achieve interoperability it is critical that these components are consistently
 970 represented regardless of in 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 artefacts that are fully conformant to the latest version of the UN/CEFACT Core Components Technical Specification.	4

971 **5.8 Schema Location**

972 Schema locations:

- 973
- Are required to be in the form of a URI scheme.

- 974 • Are associated to the namespace of the file being accessed.
- 975 • Are typically defined as URLs because of resolvability limitations of URNs.
- 976 • Can be defined as absolute path or relative paths.

977 According to the W3C XML Schema specification, part 0, the `schemaLocation`
978 attribute

979 "... provides hints from the author to a processor regarding the location of a
980 schema document. The author warrants that these schema documents are
981 relevant to checking the validity of the document content, on a namespace by
982 namespace basis."³

983 The value provided in the `xsi:schemaLocation` attribute is:

984 "...only a hint and some processors and applications will have reasons to not
985 use it."⁴

986 Thus the presence of these hints does not require the processor to obtain or use the
987 cited schema documents, and the processor is free to use other schemas obtained
988 by any suitable means, or to use no schema at all.

989 In practical implementations XML tools attempt to acquire resources using the
990 schema location attribute. The implication of the `xsi:schemaLocation` attribute
991 pointing to an absolute path (e.g., hard-drive location; URL) is that when tools
992 attempt to acquire the resources and they are not available at the specified location,
993 the tool may raise errors. In the case of URL-formatted `xsi:schemaLocation`
994 values, this might occur after a seemingly lengthy timeout period – a period in which
995 other work cannot be done. On the other hand, relative paths increase the likelihood
996 that resources will be readily available to tools (assuming well organized schema
997 files). Thus using an absolute path approach with URL-formatted
998 `xsi:schemaLocation` values often represents a challenge in practical
999 implementations as it requires open internet connections at run-time (due to tool
1000 implementations) and is seen as a security issue by a number of implementers.

1001 Providing the `schemaLocation` value as a relative path provides an overall
1002 improvement in user productivity, including off-line use. It is important to note that
1003 this approach doesn't prohibit making resources available on-line (much in the same
1004 way that HTML documents frequently provided references to relative locations for
1005 images).

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

1006 Example 5-16 shows an example of using a relative path from the containing XML
1007 Schema File to an imported BusinessDataType XML Schema File.

1008 **Example 5-16: Relative path `schemaLocation`.**

1009

```
<xsd:import namespace="urn:un:unece:uncefact:ordermanagementdata:draft:1"  
1010 schemaLocation="../../data/draft/BusinessDataType_1p0.xsd"/>
```

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

⁴ <http://www.w3.org/TR/xmlschema-0/#schemaLocation>

1011 **5.9 Versioning Scheme**

1012 The UN/CEFACT versioning scheme consists of the following values:

- 1013 • Status of the XML Schema File.
- 1014 • A major version number.
- 1015 • A minor version number.
- 1016 • A revision number.

1017 These values are declared in the version attribute in the `xsd:schema` element.1018 The major version number is also reflected in the namespace declaration for
1019 each XML Schema File rule. See 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

1020 **5.9.1 Major Versions**

1021 A major version of a UN/CEFACT XML Schema File constitutes significant non-
1022 backwards compatible changes. If any XML instance based on an older major
1023 version of UN/CEFACT XML Schema attempts validation against a newer version, it
1024 may experience validation errors. A new major version will be produced whenever
1025 non-backward compatible changes occur. This would include the following changes:

- 1026 • Removing or changing values in enumerations
- 1027 • Changing of element names, type names and attribute names
- 1028 • Changing the structures so as to break polymorphic processing capabilities
- 1029 • Deleting or adding mandatory elements or attributes
- 1030 • Changing cardinality from optional to mandatory

1031 Major version numbers will be based on logical progressions to ensure semantic
1032 understanding of the approach and guarantee consistency in representation. Non-
1033 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
----------	-------------------------------------------------------------------------------------------------------------------	---

1034 **5.9.2 Minor Versions**

1035 Within a major version iteration of a UN/CEFACT XML Schema File there could
 1036 potentially be a series of minor, or backward compatible, changes. Each minor
 1037 version will be compatible with both preceding and subsequent minor versions within
 1038 the same major version. The minor versioning scheme thus helps to identify
 1039 backward and forward compatibility. Minor versions will only be increased when
 1040 compatible changes occur, i.e.

1041 • Adding values to enumerations.

1042 • Optional extensions.

1043 • 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
----------	------------------------------------------------------------------------------------------------------------------------------------------------------	---

1044 Minor versions will be declared using the **xsd:version** attribute in the
 1045 **xsd:schema** element. It is only necessary to declare the minor version in the
 1046 schema version attribute since instance documents with different minor versions are
 1047 compatible with the major version held in the same namespace. By using the version
 1048 attribute in each document instance, the application can provide the appropriate logic
 1049 switch for different compatible versions without having knowledge of the schema
 1050 version which the document instance was delivered.

1051 Compatibility includes consistency in naming of the schema constructs to include
 1052 elements, attributes, and types. UN/CEFACT minor version changes will not include
 1053 renaming XML Schema constructs.

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

1054 For a particular namespace, the major version and subsequent minor versions and
 1055 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
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1056 6 Application of Context

1057 The intent of this NDR is to express everything that is necessary in a UN/CEFACT
1058 XML Schema to enable integration of business information within an XML Schema
1059 conformant XML instance message. To accomplish this, the XML Schema will
1060 address all aspects of the business information to include:

- 1061 • Business semantics – The meaning of business information in
1062 communication.
 - 1063 ○ Meaning can vary between different individuals depending on the
1064 context of the sender and the receiver of the information.
 - 1065 ○ Meaning can be the same between different individuals depending on
1066 the context of the sender and the receiver of the information.
- 1067 • Business context – The circumstances that determine the meaning of
1068 business information. The business context may change the semantic
1069 meaning for the sender and or the receiver of the information.

1070 In CCTS, BIEs represent context specific artefacts for a message. CCTS defines
1071 different context categories that capture context category values. BIE artefacts may
1072 be defined within any number of combinations of context categories and context
1073 category values within a category. BIEs may have the same name with different
1074 context values and different content models. As identified in Section 5.6, the
1075 namespace mechanism uses the packaging structure of a CCTS compliant model
1076 that enables the reuse of common components through a hieracrchy relationship.

1077 [Note:]

1078 While the package hierarchy relationship allows the reuse of specific BIEs, it also
1079 means that any changes to a BIE within one package affects the BIEs use in other
1080 packages that reuse it. Model developers must consider this in their design. For this
1081 reason, linking of BIEs across packages is strongly discouraged.

1082 [Note:]

1083 It is possible to extend the namespace described in section [5.6 Namespace Scheme](#)
1084 for an implementation set of schemas to include a business context identifier on the
1085 end of the namespace to express the full business context of the reduced set of XML
1086 Schemas. While this business context identifier is out side the scope of this technical
1087 specification, it is recommended that when used, this identifier be a Universally
1088 Unique Identifier (UUID).

1089 The full business context for every BIE is fully expressed within the XML Schema
1090 definition for each XML Schema Component as an `xsd:appInfo` declaration
1091 following the structure defined in section [7.5.2 Application Information \(AppInfo\)](#).

1092 **7 General XML Schema Definition Language Conventions**

1093 The XML Schema Definition (XSD) specification defines many constructs that can be
 1094 used to express a model. The purpose of this section is to provide a profile and set
 1095 of rules based on general best practices for those XSD constructs that can be used
 1096 and to identify those constructs that should not be used to include:

- 1097 • Overall XML Schema Structure and Rules
- 1098 • Attribute and Element Declarations
- 1099 • Type Definitions
- 1100 • Use of Extension and Restriction
- 1101 • Annotation

1102 **7.1 Overall XML Schema Structure and Rules**

1103 **7.1.1 XML Schema Declaration**

1104 An XML Schema File is an XML Document. As defined in XML, the first line in an
 1105 XML Document should be an XML declaration which specifies the xml version being
 1106 used. The XML declaration may also specify the encoding being used. All
 1107 UN/CEFACT XML Schema will:

- 1108 • Contain an XML declaration.
- 1109 • Use XML version 1.0.
- 1110 • Use UTF-8 encoding.

[R 9EDA]	Every UN/CEFACT XML Schema File must start with an XML declaration that specifies the xml version and encoding being used.	1
[R AABF]	Every UN/CEFACT XML Schema File must use XML version 1.0.	1
[R 88E2]	Every UN/CEFACT XML Schema File MUST use UTF-8 encoding.	1

1111 Example 7-1 shows the XML declaration for the XML Schema File.

1112 **Example 7-1: XML Schema File Line 1 setting the XML version and encoding**

1113

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

1114 **7.1.2 XML Schema File Identification and Copyright Information**

1115 After the XML declaration, information about the schema is provided in the form of
 1116 comment lines. The template 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
----------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1117 7.1.3 Schema Declaration

1118 Following the XML declaration and comments, the `xsd:schema` element is
 1119 declared. The `xsd:schema` element is a component for the schema that includes
 1120 attributes that affect how the rest of the document behaves and how XML parsers
 1121 and other tools treat it. The XML Schema Component will have:

- 1122 • `elementFormDefault` set to qualified
- 1123 • `attributeFormDefault` set to unqualified
- 1124 • 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

1125 Example 7-2 shows a XML Schema snippet declaring the `schema` component,
 1126 setting the namespace token to `xsd`, setting the `elementFormDefault` to qualified
 1127 and setting the `attributeFormDefault` to unqualified.

1128 Example 7-2: Element and Attribute Form Default

```

1129 <xsd:schema targetNamespace=" ... see namespace ..."
1130 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1131 elementFormDefault="qualified" attributeFormDefault="unqualified">

```

1132 7.1.4 CCTS Artefact Metadata

1133 CCTS defines specific metadata associated with each CCTS artefact. The elements
 1134 that will be used to represent this metadata will be defined in a separate CCTS
 1135 Metadata XML Schema File.

1136 The CCTS XML Schema File will be named Core Components Technical
 1137 Specification Schema File.

1138 The CCTS XML Schema File will be assigned to its own namespace and use a prefix
 1139 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 <i>Core Components Technical Specification Schema File</i> 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

1140 7.1.5 Constraints on Schema Construction

1141 In addition to general XML Schema structure, constraints on certain XML Schema
 1142 rules are necessary to ensure maximum interoperability for business-to-business
 1143 and application-to-application interoperability.

[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

1144 7.2 Attribute and Element Declarations

1145 7.2.1 Attributes

1146 User defined attributes are only used in UN/CEFACT schema to convey BDT
 1147 Supplementary Components as part of a BDT `xsd:type` definition. For some BDTs,
 1148 the `xsd:attributes` of an XSD data type definition in XSD part two fully meet the
 1149 requirements of the BDT Supplementary Components as defined in the data model.
 1150 In those cases the BDT will use the XSD data type as its base type, and will use the
 1151 attributes to represent the necessary Supplementary Components. Where this is not
 1152 the case, user defined attributes will be declared to represent the necessary
 1153 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 <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]	An 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]	An xsd:attribute that represents a Supplementary Component which uses identifiers MUST be based on the xsd:simpleType of the appropriate identifier scheme.	1

1154 7.2.2 Elements

1155 Elements are declared for the document level business information payload, ABIEs,
1156 BBIEs, and ASBIEs whose **aggregationKind=shared**.

1157 7.2.2.1 Element Declaration

1158 Every **ccts:BBIE** artefact is declared as an **xsd:element** of the
1159 **xsd:simpleType** or **xsd:complexType** that instantiates its BDT.

1160 7.2.2.2 Empty Elements

1161 In general, the absence of an element in an XML document does not have any
1162 particular meaning - it may indicate that the information is unknown, or not
1163 applicable, or the element may be absent for some other reason. The XML Schema
1164 specification does provide a feature, the **xsd:nillable** attribute, whereby an
1165 element may be transferred with no content, with an **xsi:nil** attribute to indicate
1166 that it is intentionally empty.

1167 In order to respect the principles of the CCTS and to retain semantic clarity, empty
1168 elements and the nillability feature of XML Schema will not be used by UN/CEFACT
1169 XML Schemas.

[R B8B6]	Empty elements MUST NOT be used.	3
[R 8337]	The xsd:nillable attribute MUST NOT be used.	3

1170 [Note:]

1171 See rule [R B4D1] for the allowed restricted use of **xsi:nil** by organizations other
1172 than UN/CEFACT.

1173 7.3 Type Definitions

1174 XSD supports both named and anonymous types. Named types are reusable.
1175 Anonymous types are part of a single element declaration and are not reusable by
1176 other elements. In keeping with the guiding principles of CCTS, all types will be
1177 named.

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

1178 7.3.1 Simple Type Definitions

1179 **xsd:simpleTypes** must always be used where they satisfy the user's business
1180 requirements. This is typically for code lists, identifier schemes, and BDTs. Examples
1181 7-3 shows a simple type defined in the BDT XML Schema File.

1182 **Example 7-3: Simple Types in Business Data Type XML Schema File**

```
1183 <xsd:simpleType name="DateTimeType">
1184   <xsd:annotation>
1185     ... see annotation ...
1186   </xsd:annotation>
1187   <xsd:restriction base="xsd:dateTime"/>
1188 </xsd:simpleType>
```

1189 Example 7-4 shows a simple type defined in a Code List XML Schema File.

1190 **Example 7-4: Simple Types in a Code List XML Schema File**

```
1191 <xsd:simpleType name="CurrencyCodeContentType">
1192   <xsd:restriction base="xsd:token">
1193     <xsd:enumeration value="ADP">
1194       ...see enumeration of code lists ...
1195     </xsd:enumeration>
1196   </xsd:restriction>
1197 </xsd:simpleType>
```

1198 7.3.2 Complex Type Definitions

1199 An **xsd:complexType** will be defined to express the content model of each CCTS
1200 ABIE. An **xsd:complexType** will also be defined to express the value domain of a
1201 CCTS BDT when an XSD built-in data type does not meet the business
1202 requirements (e.g. Supplementary Components).

[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 whose value domain cannot be fully expressed using an xsd:simpleType .	1

1203 Example 7-5 shows a complex type defined for an **Account**. **Details** ABIE.

1204 **Example 7-5: Complex Type of Object Class "AccountType"**

```
1205 <xsd:complexType name="AccountType">
1206   <xsd:annotation>
1207     ... see annotation ...
1208   </xsd:annotation>
1209   <xsd:sequence>
1210     ... see element declaration ...
1211   </xsd:sequence>
1212 </xsd:complexType>
```

1213 In order to increase consistency in use and enable accurate and complete
1214 representation of what is allowed in the design of CCTS artefacts, the **xsd:sequence**
1215 and **xsd:choice** compositors will be used to express the content model for
1216 **xsd:complexType** definitions. The **xsd:a11** XML Schema compositor will not be
1217 used.

[R A010]	The <code>xsd:all</code> element MUST NOT be used.	1
----------	----------------------------------------------------	---

1218 7.4 Use of Extension and Restriction

1219 In keeping with CCTS, XML Schema Components are based on the concept that the
 1220 underlying semantic structures of the BIEs are normative forms of standards that
 1221 developers are not allowed to alter without coordination with the owner of the
 1222 component at the data model level. As business requirements dictate, new BIE
 1223 artefacts will be created in the data model and represented as XML Schema
 1224 Components by defining new types and declaring new elements. The concept of
 1225 derivation from existing types through the use of `xsd:extension` and
 1226 `xsd:restriction` will only be used in limited circumstances where their use does
 1227 not violate this principle.

1228 It is understood that other standards organizations using this specification may
 1229 choose to use `xsd:extension` and/or `xsd:restriction` to define new
 1230 constructs that are extended or restricted from existing constructs.

1231 7.4.1 Extension

1232 The `xsd:extension` component will only be used in an `xsd:complexType` within
 1233 the BDT XML Schema File, and only for declaring attributes to support
 1234 Supplementary Components.

[R AB3F]	<code>xsd:extension</code> MUST only be used in the BDT XML Schema File.	4, 6
[R 9D6E]	<code>xsd:extension</code> MUST only be used for declaring <code>xsd:attributes</code> to accommodate relevant Supplementary Components.	4, 6

1235 Example 7-6 shows an extension of a simple type using the `xsd:extension`
 1236 mechanism.

1237 Example 7-6: Extension of Simple Type

```

1238 <xsd:complexType name="AmountType">
1239   <xsd:annotation>
1240     ... see annotation ...
1241   </xsd:annotation>
1242   <xsd:simpleContent>
1243     <xsd:extension base="xsd:decimal">
1244       <xsd:attribute name="unitCode" type="xsd:token"/>
1245     </xsd:extension>
1246   </xsd:simpleContent>
1247 </xsd:complexType>

```

1248 7.4.2 Restriction

1249 The CCTS specification employs the concept of semantic restriction in creating
 1250 specific instantiations of core components. BIEs may be logical semantic restrictions
 1251 from a parent BIE. However the physical implementation in XML Schema is that
 1252 every BIE is directly restricted from its source ACC. Since ACCs are not instantiated

1253 as XML artefacts, and every BIE is directly restricted from its source ACC, the use of
1254 **xsd:restriction** is not supported for BIE type definitions.

1255 Unlike BIEs, qualified BDTs are a restriction of their direct parent BDT and represent
1256 a restricted value domain. Accordingly, **xsd:restriction** will be used as
1257 appropriate to define qualified BDT types that are derived from less qualified or
1258 unqualified BDT types. **xsd:restriction** can be used for facet restriction and/or
1259 attribute restriction. Both BDT **xsd:simpleType(s)** and **xsd:complexType(s)** can
1260 use **xsd:restriction**.

1261 BDT restriction may be accomplished through the restriction of CCLs by creating a
1262 new BCL, or the restriction of the allowed set of values by creating a new BIS. A BCL
1263 may restrict an existing code list to only the values allowed for a given business
1264 process. A BIS may restrict an existing information scheme to only the values
1265 allowed for a given business process.

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

1266 Where used, the derived types must always be named uniquely.

[R 8AF7]	When xsd:restriction is applied to a data type the resulting type MUST be uniquely named.	1
----------	--------------------------------------------------------------------------------------------------	---

1267 Example 7-7 shows a restriction of a simple type.

1268 **Example 7-7: Restriction of Simple Type**

```

1269 <xsd:simpleType name="TaxAmountType">
1270   <xsd:annotation>
1271     ... see annotation ...
1272   </xsd:annotation>
1273   <xsd:restriction base="AmountType">
1274     <xsd:totalDigits value="10"/>
1275     <xsd:fractionDigits value="3"/>
1276   </xsd:restriction>
1277 </xsd:simpleType>

```

1278 **7.5 Annotation**

1279 All UN/CEFACT XML Schema constructs will use the **xsd:documentation** and
1280 **xsd:appInfo** elements within an **xsd:annotation** element to provide CCTS
1281 artefact metadata, context values, and business rules.

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

1282 **7.5.1 Documentation**

1283 The annotation **xsd:documentation** will be used to convey the metadata specified
1284 by CCTS for CCTS artefacts. Conversely, all elements specified within an
1285 **xsd:documentation** element will be limited to expressions of CCTS artefact
1286 metadata.

- 1287 The following annotations are required as defined for each CCTS artefact in their
1288 sub-sections in [Section 8 XML Schema Files](#).
- 1289 • **UniqueID** – The unique identifier assigned to the artefact in the library.
1290 (UniqueID)
 - 1291 • **VersionID** – The unique identifier assigned to the version of the artefact in the
1292 library.
 - 1293 • **ObjectClassQualifierName** – A word or words which help define and
1294 differentiate an ABIE from its associated CC and other BIEs. It enhances the
1295 semantic meaning of the DEN to reflect a restriction of the concept,
1296 conceptual domain, content model or data value. The order in which the
1297 qualifiers are expressed indicate the order to be used, where the first one is to
1298 be the first order qualifier.
 - 1299 • **ObjectClassTermName** – A semantically meaningful name for the object
1300 class. It is the basis for the DEN.
 - 1301 • **Cardinality** – Indicates the cardinality of the associated artefact.
 - 1302 • **SequencingKey** – Indicates the sequence of the associated artefact within
1303 the larger BIE.
 - 1304 • **DictionaryEntryName** – The Dictionary Entry Name (DEN) of the artefact.
1305 (Name)
 - 1306 • **Definition** – The semantic meaning of the artefact. (Definition)
 - 1307 • **BusinessTermName** – A synonym term under which the artefact is
1308 commonly known and used in business. (BusinessTerm)
 - 1309 • **AssociationType** – Indicates if the UML Association Kind between the
1310 associating and associated ABIE =*shared* or =*composite*.
 - 1311 • **PropertyTermName** – Represents a distinguishing characteristic of the object
1312 class and shall occur naturally in the definition.
 - 1313 • **PropertyQualifierName** – Is a word or words which help define and
1314 differentiate a property. It further enhances the semantic meaning of the
1315 property. The order in which the qualifiers are expressed indicate the order to
1316 be used, where the first one is to be the first order qualifier.
 - 1317 • **RepresentationTermName** – An element of the component name which
1318 describes the form in which the component is represented.
 - 1319 • **AssociatedObjectClassTermName** – The Associated Object Class Term
1320 represented by the artefact.
 - 1321 • **AssociatedObjectClassQualifierTerm** – A term(s) that qualifies the
1322 Associated Object Class Term. The order in which the qualifiers are
1323 expressed indicate the order to be used, where the first one is to be the first
1324 order qualifier.
 - 1325 • **PrimitiveTypeName** – The name of the primitive type name from the Data
1326 Type Catalogue.
 - 1327 • **DataTypeName** – The name of the DataType. This DataType is defined in the
1328 Data Type Catalogue.

- 1329 • **DataTypeQualifierName** – Is a word or words which help define and
 1330 differentiate a Data Type. It further enhances the semantic meaning of the
 1331 DataType. The order in which the qualifiers are expressed indicate the order
 1332 to be used, where the first one is to be the first order qualifier.
- 1333 • **DefaultIndicator** – Indicates that the specific Value Domain is the default.
- 1334 • **DefaultValue** – Is the default value.
- 1335 • **SchemeOrListID** – The identifier assigned to the scheme or list that uniquely
 1336 identifies it.
- 1337 • **SchemeOrListAgencyID** – The unique identifier assigned to the Agency that
 1338 owns or is responsible for the Scheme or Code List being referenced.
- 1339 • **SchemeOrListModificationAllowed Indicator** – Indicates whether the
 1340 values being validated can be outside the enumerations specified by the
 1341 Scheme or Code List.
- 1342 • **Name** – The name of the Code List Value.
- 1343 • **Description** – The long description of the Code Value.

1344 Table 7-1 provides a summary view of the annotation data as defined in this section
 1345 and the CCTS artefacts in which each is expressed within the resulting XML
 1346 Schema.

1347 [Note:]

1348 It is important to realize that while this specification lists documentation, the
 1349 documentation varies for the different types of CCTS artefacts depending on their
 1350 status as a registry class. ABIEs, BBIEs, ASBIEs and BDTs are all Registry Classes
 1351 in that they are uniquely identifiable within the Core Component Library (CCL). An
 1352 RSM, is a high level ABIE aggregation, is also treated as a Registry Class.

1353 [Note:]

1354 BBIE, ASBIE, Code List, Code List Value and Supplementary Components are not
 1355 Registry Classes therefore they do not include the UniqueID or VersionID from the
 1356 Registry Class.

	rsm:RootSchema	ABIE xsd:complexType	BBIE xsd:element	ASBIE: xsd:element	bdt:BusinessDataType	bdt:ContentComponent ValueDomain	bdt:Supplementary Component	bdt:Supplementary ComponentValue Domain	Code List	Code List Value
Unique ID	M	M		M	M					
Version ID	M	M		M	M					
Object Class Qualifier	O	O	M	O						

	rsm:RootSchema	ABIE xsd:complexType	BBIE xsd:element	ASBIE: xsd:element	bdt:BusinessDataType	bdt:ContentComponent ValueDomain	bdt:Supplementary Component	bdt:Supplementary ComponentValue Domain	Code List	Code List Value
Name	R	R		R						
Object Class Term Name	M	M	M	M						
Cardinality			M	M			M			
Sequencing Key			M	M						
Dictionary Entry Name	M	M	M	M	M		M			
Definition	M	M	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			
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						O		O		
Data Type Term Name					M		M			
Data Type Qualifier Name					M		M			
Default Indicator						M		M		
Default Value						O		O		

	rsm:RootSchema	ABIE xsd:complexType	BBIE xsd:element	ASBIE: xsd:element	bdt:BusinessDataType	bdt:ContentComponent ValueDomain	bdt:Supplementary Component	bdt:Supplementary ComponentValue Domain	Code List	Code List Value
Scheme Or List ID						O		O	M	
Scheme Or List Version ID						O		O	M	
Scheme Or List Agency ID						O		O	M	
Scheme Or List Modification Allowed Indicator						O		O	M	
Name										M
Description										O
<p>Key: M – Mandatory O – Optional R – Repeating Yellow Shading – Not expressed in XML Schema</p>										

1357 **Table 7-1 Annotation Data Summary**

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

1358 [Section 8 XML Schemas](#) and [Appendix F](#) specify normative information for the
 1359 specific annotation required for each of the CCTS artefacts.

1360 This documentation is intended to be used to connect the XML Schema defined
 1361 artefact to the model artefact on which it is based. This is important for standard XML
 1362 Schemas and for fully expressed XML Schemas for a runtime implementation.
 1363 However, XML Schemas directly used in a runtime implementation may choose not
 1364 to include this documentation in order to reduce the size of the XML Schema. This is
 1365 often done in order to increase the throughput of XML Instances and to increase the
 1366 volume capacity for a particular system. If this approach is selected, the runtime XML
 1367 Schema may only be an exact copy of the fully documented XML Schema – with
 1368 only the annotation documentation (**xsd:documentation**) elements removed.

1369 As identified in [Section 7.1.4 CCTS artefact Metadata](#), the required
 1370 `xsd:documentation` elements are declared in the CCTS Metadata XML Schema
 1371 File. This file is imported in all Root, BIE, BDT and Code List XML Schema Files in
 1372 lieu of re-declaring them in each schema.

1373 Example 7-8 provides an example of annotation documentation for an ABIE that
 1374 conforms to the ccts structure.

1375 **Example 7-8: Example of Annotation Documentation of an ABIE**

```

1376 <xsd:annotation>
1377   <xsd:documentation xml:lang="en">
1378     <ccts:UniqueID>UNBE000000</ccts:UniqueID>
1379     <ccts:VersionID>1.0</ccts:VersionID>
1380     <ccts:DictionaryEntryName>Customer. Account</ccts:DictionaryEntryName>
1381     <ccts:Definition>The account for the customer</ccts:Definition>
1382     <ccts:ObjectClassQualifierName>Customer</ccts:ObjectClassQualifierName>
1383     <ccts:ObjectClassTermName>Account</ccts:ObjectClassTermName>
1384   </xsd:documentation>
1385 </xsd:annotation>
  
```

1386 Each UN/CEFACT construct containing a code must include documentation that will
 1387 identify the code list(s) that must be supported when the construct is used.

1388 [Appendix F Section F.1 Annotation Documentation](#) shows the XML Schema
 1389 definition of annotation documentation for each of the types of components from
 1390 CCTS.

1391 **7.5.2 Application Information (AppInfo)**

1392 The annotation `xsd:appInfo` will be used to convey the Usage Rules and the
 1393 business context that is applicable for each BIE and BDT artefact and the resulting
 1394 XML Schema artefacts used to express them.

1395 [Note:]

1396 The UN/CEFACT TMG UCM project is defining the context mechanism that will
 1397 support refining context categories in a given business circumstance. Once that
 1398 specification is finalized, this section may change.

1399 Example 7-9 shows the XML Schema definition of the annotation application
 1400 Information structure `ccts:UsageRule`.

1401

1402 **Example 7-9: XML Schema definition for annotation applInfo for ccts:UsageRule**

```

1403 <xsd:schema
1404   ...
1405 <xsd:element name="UsageRule" type="ccts:UsageRuleType" />
1406 <xsd:complexType name="UsageRuleType">
1407   <xsd:sequence>
1408     <xsd:element name="UniqueID" type="EntityUniqueIdentifierType" />
1409     <xsd:element name="Constraint" type="TextType" />
1410     <xsd:element name="ConstraintTypeCode" type="CodeType" />
1411     <xsd:element name="ConditionTypeCode"
1412 type="ConditionTypeCodeType" />
1413   </xsd:sequence>
1414   maxOccurs="unbounded" />
1415 </xsd:complexType>
1416   ...
1417 </xsd:schema>

```

1418 [Appendix F Section F.2 Annotation Application Information](#) shows the XML Schema
 1419 definition of the annotation application Information structure for
 1420 **ccts:BusinessContext**.

1421 Both **ccts:UsageRule** and **ccts:BusinessContext** are applied to each of the
 1422 XML Schema Components **xsd:element**, **xsd:complexType** and
 1423 **xsd:simpleType** in order to communicate the usage and context in which the
 1424 corresponding CCTS artefacts are applicable.

[R 9B07]	Each xsd:element declaration, and each xsd:complexType and xsd:simpleType definition MUST have an xsd:annotation xsd:appInfo declared that includes zero or more ccts:UsageRule elements and one or more ccts:BusinessContext elements.	1
----------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1425 **7.5.2.1 Usage Rules**

1426 CCTS defines the concept of usage rules to convey instructions on how to use a
 1427 CCTS artefact in a given context. Usage rules have a **ccts:ConstraintType**
 1428 which classifies the rules as being structured (expressed in a formal language such
 1429 as the Object Management Group's Object Constraint Language (OCL)) or
 1430 unstructured (free form text).

1431 Usage Rules are communicated through zero or more **ccts:UsageRule** XML
 1432 Schema Elements within an **xsd:appInfo** element. Unstructured usage rule
 1433 constraint values are expressed as free form text. Structured usage rule constraint
 1434 values are expressed in a formal constraint language such as the Object
 1435 Management Group (OMG) Object Constraint Language (OCL).and are suitable for
 1436 direct application processing.

[R 88DE]	Usage rules MUST be expressed within the appropriate BDT, Content Component or Supplementary Component xsd:annotation xsd:appInfo ccts:UsageRule element.	1
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

[R B851]	<p>The structure of the <code>ccts:UsageRule</code> element MUST be:</p> <ul style="list-style-type: none"> • <code>ccts:UniqueID</code> [1..1] – A unique identifier for the UsageRule. • <code>ccts:Constraint</code> [1..1] – The actual constraint expression. • <code>ccts:ConstraintTypeCode</code> [1..1] – The type of constraint E.g. unstructured, OCL. • <code>ccts:ConditionTypeCode</code> [1..1] – The type of condition. Allowed values are <code>pre-condition</code>, <code>post-condition</code>, and <code>invariant</code>. 	1
----------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1437 The `ccts:ConstraintTypeCode` will be taken from a code list schema.

[R A1CF]	A <code>ccts:ConstraintType</code> code list XML Schema File MUST be created.	1
----------	-------------------------------------------------------------------------------	---

1438 7.5.2.2 Business Context

1439 The BusinessContext defined within each `xsd:appInfo` contains one or more
 1440 `ccts:ContextUnit` elements which in turn contains one or more values for each
 1441 of the identified context categories recognized by CCTS. The following
 1442 `xsd:appInfo` structures are required as defined in each of the sub-sections in the
 1443 section [8 XML Schema Files](#) that correspond to the different CCTS artefacts.

- 1444 • Business Process Context Category
- 1445 • Business Process Role Context Category
- 1446 • Supporting Role Context Category
- 1447 • Industry Classification Context Category
- 1448 • Product Classification Context Category
- 1449 • Geopolitical Context Category
- 1450 • Official Constraints Context Category
- 1451 • System Capabilities Context Category

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

1452 Using this structure it is possible to indicate all of the context categories in which a
 1453 BIE is applicable, and all of the applicable context values within a context category
 1454 as shown in Example 7-10.

1455

1456 **Example 7-10: Use of the `xsd:appInfo` and `ccts:BusinessContext`**

```

1457 <xsd:element name="<name>" type="<type>">
1458 <xsd:annotation
1459 ... (documentation) ...
1460 <xsd:appinfo source="urn:un:unece:uncefact:...">
1461 <ccts:UsageRules>
1462 ...
1463 </ccts:UsageRules>
1464 <ccts:BusinessContext>
1465 <ccts:ContextUnit>
1466 <ccts:BusinessProcessContextCategory>
1467 <ccts:BusinessTransactionDocumentCode>0062
1468 </ccts:BusinessTransactionDocumentCode>
1469 <!-- PurchasingContractUseRequest -->
1470 <ccts:BusinessTransactionDocumentCode>0081
1471 </ccts:BusinessTransactionDocumentCode>
1472 <!-- CataloguePublicationRequest -->
1473 ... (further business transaction document codes) ...
1474 </ccts:BusinessProcessContextCategory>
1475 <ccts:IndustryClassificationContextCategory>
1476 <ccts:IndustryClassificationCode>0001
1477 </ccts:IndustryClassificationCode>
1478 <!-- Aerospace -->
1479 <ccts:IndustryClassificationCode>0002
1480 </ccts:IndustryClassificationCode>
1481 <!-- Defence -->
1482 <ccts:IndustryClassificationCode>0006
1483 </ccts:IndustryClassificationCode><!-- CP -->
1484 ... (further business transaction document codes) ...
1485 </ccts:IndustryClassificationContextCategory>
1486 <ccts:GeopoliticalContextCategory>
1487 <ccts:CountryCode>DE</ccts:CountryCode>
1488 <!-- Germany -->
1489 <ccts:CountryCode>FR</ccts:CountryCode>
1490 <!-- France -->
1491 <ccts:CountryCode>US</ccts:CountryCode>
1492 <!-- USA -->
1493 <ccts:CountryCode>AT</ccts:CountryCode>
1494 <!-- Austria -->
1495 ... (further business transaction document codes) ...
1496 </ccts:GeopoliticalContextCategory>
1497 ... (further business context categories) ...
1498 </ccts:ContextUnit>
1499 </ccts:BusinessContext>
1500 </xsd:appinfo>
1501 </xsd:annotation>
1502 </xsd:element>

```

1503 8 XML Schema Files

1504 This section describes how the requirements of the various XML Schema Files that
1505 are incorporated within the UN/CEFACT library are built through the application of
1506 context categories, unique namespaces and the rules of this specification.

- 1507 • XML Schema Files, Context and Namespaces
- 1508 • Root XML Schema Files
- 1509 • Business Information Entities XML Schema Files
- 1510 • Business Data Type XML Schema Files
- 1511 • Code List XML Schema Files
 - 1512 ○ General Code List XML Schema Components
 - 1513 ○ Common Code List XML Schema Components
 - 1514 ○ Business Code List XML Schema Components
- 1515 • Identifier Scheme XML Schema Files
 - 1516 ○ General Identifier Scheme XML Schema Components
 - 1517 ○ Common Identifier Scheme XML Schema Components
 - 1518 ○ Business Identifier Scheme XML Schema Components

1519 8.1 XML Schema Files, Context and Namespaces

1520 As indicated in section [5.7 XML Schema Files](#) the XML Schema Files have
1521 dependencies upon one another.

1522 Figure 8-1 shows these dependencies and how they are realized using the
1523 `xsd:include` and `xsd:import` XML Schema features. Since the CCTS
1524 conformant model is represented within a namespace scheme that mirrors the model
1525 packaging, all of the XML Schema Files for a given package are defined within a
1526 corresponding namespace. The XML Schema Files for other packages are likewise
1527 defined in namespaces that reflect the other package.

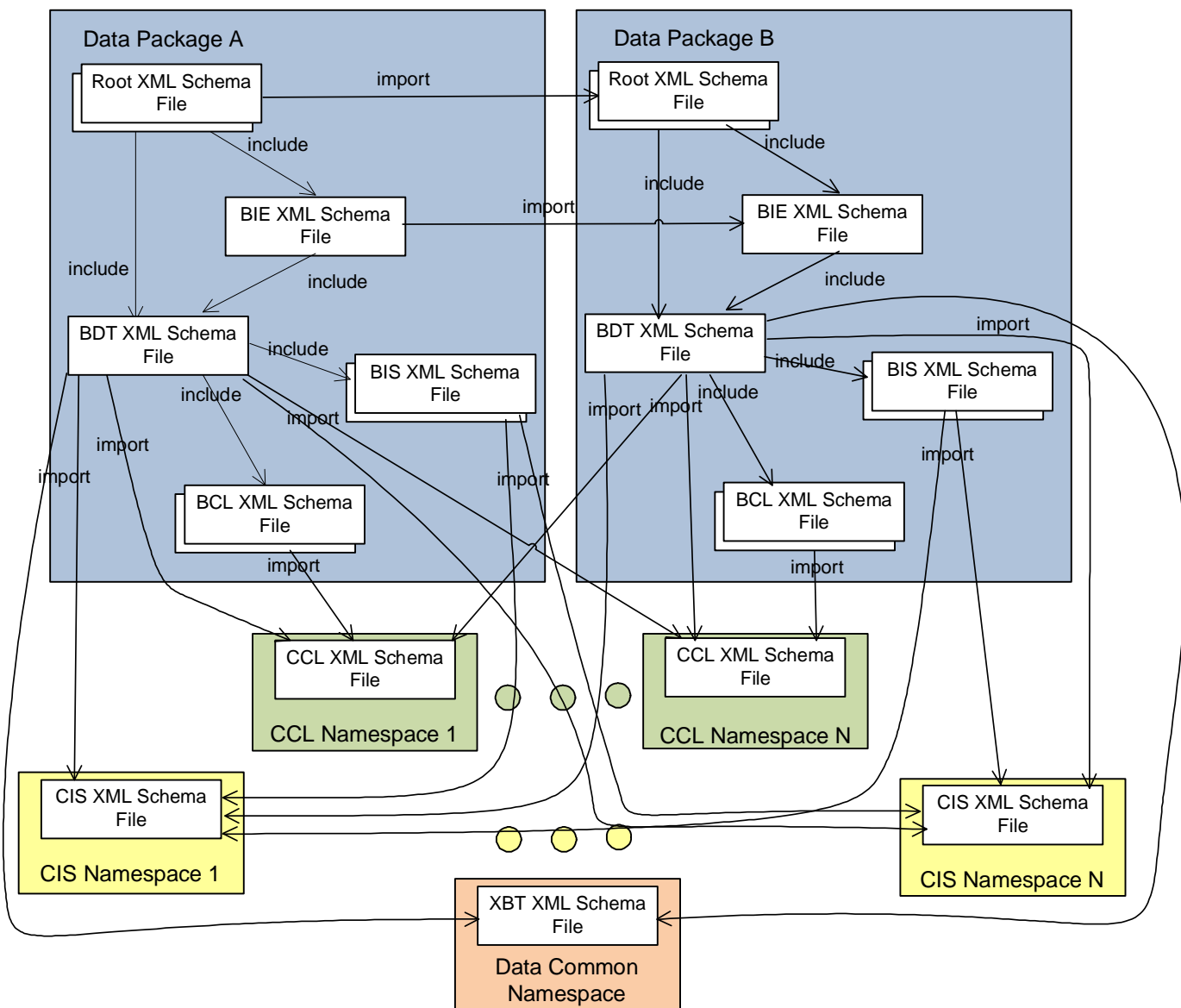
1528 Figure 8-1 shows two data packages – “Data Package A” and “Data Package B.”
1529 The namespaces used to express the two data packages are independent. However,
1530 in order to achieve reuse there may be shared dependencies in cases where BIEs
1531 are shared across data packages within the model.

1532 Additionally, Common Code Lists that are independent of all context may also be
1533 used by any number of XML Schema Files.

1534 All XML Schemas, other than CCL and CIS schema, are derived from a CCTS
1535 conformant model. The packaging expressed within these models are realized as
1536 namespaces. These packages reflect the context categories and the requirements of
1537 the given model and they are assigned to a unique namespace and given a unique
1538 token that represents the data package in which it is designed.

[R B96F]	Each Root, BIE, BDT and BCL XML Schema File MUST be defined in a unique namespace that is derived from the corresponding package within the CCTS conformant model.	1
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1539 [Note:]
 1540 It is important to note in Figure 8-1 the packaging of the CCTS model does not result
 1541 in a single XML Schema File per namespace. Instead, a set of XML Schema Files
 1542 are created per namespace as indicated in Section 5 of this document.
 1543 [Note:]
 1544 As indicated earlier in Section 5. By sharing common components between
 1545 packages the model developer must be aware that changes reflected in one
 1546 package or context are reflected in all that use the shared component, whether
 1547 intended or not. For this reason, developers are encouraged to restrict the reuse of
 1548 components across packages.



1549 **Figure 8-1: Imports and Includes of XML Schema Files for UN/CEFACT**
 1550 **Moularity Model**

1551 Example 8-1 shows a namespace declaration for data package where the value is
1552 **Order Management**.

1553 **Example 8-1: Namespace Declaration for Data Package – Order Management**

1554

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

1555 Example 8-2 shows a target namespace declaration for a data package where the
1556 value is **Order Management**.

1557 **Example 8-2: Schema-element target namespace declaration for Data Package – Order**
1558 **Management**

1559

```
<xsd:schema  
1560   targetNamespace="urn:un:unece:unefact:ordermangement:data:1:draft"  
1561   xmlns:ordman="urn:un:unece:unefact:ordermanagement:data:1:draft"
```

1562 [Note:]

1563 Implementations of this specification require the use of a semantically meaningful
1564 namespace prefix like **ordman** for the **Order Management** data package.

1565 **8.2 Root XML Schema Files**

1566 The Root XML Schema File serves as the container for all schema defined content
1567 required to fulfill a business information exchange for the given payload in a package
1568 namespace. All of the Root XML Schema Files that are necessary to fulfill the
1569 information for the package are defined within the package namespace.

1570 Figure 8-1 shows multiple Root XML Schema Files defined in two different packages
1571 which results in two different namespaces where namespace A uses content from
1572 namespace B. Each package based namespace will have one or more Root XML
1573 Schema Files.

1574 **8.2.1 XML Schema Structure**

1575 Each Root XML Schema File will be structured in a standardized format as specified
1576 in Appendix B in order to ensure consistency and ease of use. The specific format is
1577 shown in Example 8-3. The Root XML Schema File must adhere to the format of the
1578 relevant sections as detailed in Appendix B.

1579

1580 **Example 8-3: Root XML Schema File Structure**

```

1581 <?xml version="1.0" encoding="UTF-8"?>
1582 <!-- =====>
1583 <!-- ===== [MODULENAME] XML Schema File =====>
1584 <!-- =====>
1585 <!--
1586     Schema agency:      UN/CEFACT
1587     Schema version:    3.0
1588     Schema date:      14 July 2009
1589
1590     Copyright (C) UN/CEFACT (2009). All Rights Reserved.
1591
1592     ... see copyright information ...
1593 -->
1594 <xsd:schema
1595     xmlns:xsd=http://www.w3.org/2001/XMLSchema
1596     targetNamespace="urn:un:unece:unefact:data:ordermanagement:3:draft"
1597     ... see namespaces ...
1598
1599     elementFormDefault="qualified" attributeFormDefault="unqualified" version="3.0">
1600 <!-- =====>
1601 <!-- ===== Includes =====>
1602 <!-- =====>
1603 <!-- ===== Include of [MODULENAME] =====>
1604 <!-- =====>
1605     ... see includes ...
1606 <!-- =====>
1607 <!-- ===== Imports =====>
1608 <!-- =====>
1609 <!-- ===== Imports of [MODULENAME] =====>
1610 <!-- =====>
1611     ... see imports ...
1612 <!-- =====>
1613 <!-- ===== Element Declarations =====>
1614 <!-- =====>
1615 <!-- ===== Root Element Declarations =====>
1616 <!-- =====>
1617     See element declarations...
1618 <!-- =====>
1619 <!-- ===== Type Definitions =====>
1620 <!-- =====>
1621 <!-- ===== Type Definitions: [TYPE] =====>
1622 <!-- =====>
1623 <xsd:complexType name="[TYPENAME]">
1624     ... see type definition ....
1625 </xsd:complexType>
1626 </xsd:schema>
    
```

1627 **8.2.2 Imports and Includes**

1628 Every Root XML Schema File in a namespace will include the BIE XML Schema File,
 1629 and the BDT XML Schema File that reside in that namespace.

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

1630 Every Root XML Schema File in a namespace may import a Root XML Schema File
 1631 from another data package namespace in order to reuse artefacts defined in the
 1632 other namespace.

[R B71D]	If a Root XML Schema File in a namespace reuses artefacts defined in another namespace, it MUST import a Root Schema File that resides in the other namespace.	1
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1633 **8.2.3 Element Declarations**1634 **8.2.3.1 Root Element Declaration – MA Component**

1635 Each business information payload message has a single root element that is
 1636 globally declared in the Root XML Schema File representing it. The global element is
 1637 named according to the business information payload that it represents and is of the
 1638 **xsd:complexType** that represents the target information payload that contains the
 1639 actual business information.⁵

1640 Each root element and its corresponding **xsd:complexType** are realizations of the
 1641 Message Assembly Component as described in [Section 5.3](#).

[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

1642 Example 8-4 shows an example of Root Element declaration with in a Root XML
 1643 Schema File.

1644 **Example 8-4: Root Element declaration**

```

1645 <!-- =====>
1646 <!-- ===== Root Element =====>
1647 <!-- =====>
1648 <xsd:element name="Invoice" type="rsm:InvoiceType">
1649 <xsd:annotation>
1650 ... see annotation ...
1651 </xsd:annotation>
1652 </xsd:element>

```

1653 **8.2.3.2 ASMA Components**

1654 Each root element is defined to contain a SBDH component and at least one ASMA
 1655 component. Each ASMA component is a local element that is defined using the type
 1656 (**xsd:complexType**) definition of the top level ABIE.

1657 The ASMA serves as a proxy for the top level ABIE within the message structure.

[R A445]	Each ASMA component MUST be realized as a local element that is defined using the type (xsd:complexType) definition of the top level ABIE for that component.	3
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

⁵ 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.

[R 9CC0]	The name of the local element defined for the ASMA Component MUST consist of an optional property term followed by the name of the ABIE to which it is associated.	3
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1658 **8.2.3.3 SBDH Component**

1659 The SBDH element is declared in the SBDH XML Schema File and is referenced in
 1660 the root schema `xsd:complexType`. See [Section 8.2.4 Type Definitions](#).

1661 **8.2.4 Type Definitions**

1662 Root XML Schema Files are limited to defining a single MA `xsd:complexType`
 1663 whose content model contains one or more ASMAs and zero or one SBDH
 1664 components. Each ASMA is realized by local element declarations that represent
 1665 the first level BIEs for a business information payload. The SBDH component is
 1666 realized through an `xsd:element` reference to the root element declaration in the
 1667 appropriate SBDH XML Schema File.

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

1668 Example 8-5 shows the definition of a Root XML Schema Files complex type
 1669 definition.

1670 **Example 8-5: Root element complex type**

1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689

```

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

1690

1691 **8.2.5 Annotations**
 1692 **8.2.5.1 Root Element**
 1693 **8.2.5.1.1 Annotation Documentation**

1694 In the Root XML Schema File the root element declaration must contain a structured
 1695 set of annotation documentation.

[R 8010]	<p>The Root XML Schema File root element declaration MUST have a structured set of annotation documentation (xsd:annotation xsd:documentation) that contains:</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. • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the business information payload. • Definition (mandatory): The semantic meaning of 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. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the Object class. It is the basis for the DEN. • BusinessTermName (optional, repeating): A synonym term under which the payload object is known by in industry. 	1
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1696 Example 8-6 shows the definition of the annotation documentation for the Root
 1697 Element.

1698 **Example 8-6: Root element annotation documentation**

```

1699 <xsd:group name="RootSchemaDocumentation">
1700   <xsd:sequence>
1701     <xsd:element name="UniqueID" type="EntityUniqueIdentifierType"/>
1702     <xsd:element name="VersionID" type="VersionIdentifierType"/>
1703     <xsd:element name="DictionaryEntryName" type="NameType"/>
1704     <xsd:element name="Definition" type="TextType"/>
1705     <xsd:element name="ObjectClassQualifierName" type="NameType"
1706     minOccurs="0" maxOccurs="unbounded"/>
1707     <xsd:element name="ObjectClassTermName" type="NameType"/>
1708     <xsd:element name="BusinessTermName" type="NameType" minOccurs="0"
1709     maxOccurs="unbounded"/>
1710   </xsd:sequence>
1711 </xsd:group>
    
```

1712 [8.2.5.1.2 Annotation Application Information \(AppInfo\)](#)

1713 The annotation `xsd:appInfo` on the Root Element is used to convey the context
 1714 that is applicable for the Root Element. The structure of the context is provided in
 1715 section [7.5.2, Application Information \(AppInfo\)](#). The specific context values for the
 1716 Root Element represent the context values for the Root XML Schema File and the
 1717 overall message.

1718 **8.2.5.2 ASMA Component – Local Element**

1719 [8.2.5.2.1 Annotation Documentation](#)

1720 In the Root XML Schema File the local element declaration for the ASMA must have
 1721 a structured set of annotation documentation.

[R A86D]	<p>For every ASMA Copoment local <code>xsd:element</code> declaration definition, a structured set of annotations MUST contain:</p> <ul style="list-style-type: none"> • UniqueID (mandatory): The unique identifier that identifies an ASMA instance in a unique and unambiguous way. • VersionID (mandatory): An unique identifier that identifies the version of an ASMA. • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the ASMA. • Definition (mandatory): The semantic meaning of the ASMA or the underling ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differeniates the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE. • BusinessTermName (optional, repeating): A synonym term in which the ABIE is commonly known. 	1
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1722 [8.2.5.2.2 Annotation Application Information \(AppInfo\)](#)

1723 The annotation `xsd:appInfo` on the ASMA element is used to convey the context
 1724 that is applicable for the ASMA. The structure of the context is provided in section
 1725 [7.5.2, Application Information \(AppInfo\)](#). The specific context values for the ASMA
 1726 represent the context values for the Root XML Schema File.

1727 **8.3 Business Information Entity XML Schema Files**

1728 A UN/CEFACT BIE XML Schema File contains all of the ABIEs used for the data
 1729 package that is reflected in the namespace. Each data package namespace will
 1730 have one and only one BIE XML Schema File.

1731 Where BIEs make direct use of BIEs in another namespace as reflected in the CCTS
 1732 model the resulting BIE XML Schema File may import the corresponding BIE XML
 1733 Schema File.

1734 [Note:]

1735 As indicated earlier in this section. By sharing common components between
 1736 packages the model developer must be aware that changes reflected in one
 1737 package or context are reflected in all that use the shared component, whether
 1738 intended or not.

1739 8.3.1 Schema Structure

1740 Each BIE XML Schema File will be structured in the standardized format detailed in
 1741 [Appendix B](#). The specific format is shown in Example 8-7 and must adhere to the
 1742 format of the relevant sections in [Appendix B](#).

1743 Example 8-7: Structure of BIE XML Schema Files

```

1744 <?xml version="1.0" encoding="UTF-8"?>
1745 <!-- ===== -->
1746 <!-- ===== ABIE XML Schema File ===== -->
1747 <!-- ===== -->
1748 <!--
1749 Schema agency:      UN/CEFACT
1750 Schema version:    3.0
1751 Schema date:       18 November 2009
1752
1753 Copyright (C) UN/CEFACT (2009). All Rights Reserved.
1754
1755     ... see copyright information ...
1756 -->
1757 <xsd:schema
1758   targetNamespace=
1759     ... see namespace declaration ...
1760   xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
1761   attributeFormDefault="unqualified">
1762   <!-- ===== -->
1763   <!-- ===== Includes ===== -->
1764   <!-- ===== -->
1765   <!-- ===== Include of [MODULENAME] ===== -->
1766   <!-- ===== -->
1767   ... see includes ...
1768   <!-- ===== -->
1769   <!-- ===== Imports ===== -->
1770   <!-- ===== -->
1771   <!-- ===== Imports of [MODULENAME] ===== -->
1772   <!-- ===== -->
1773   ... see imports ...
1774   <!-- ===== -->
1775   <!-- ===== Type Definitions ===== -->
1776   <!-- ===== -->
1777   ... see type defintions ...
1778 </xsd:schema>

```

1779 8.3.2 Imports and Includes

1780 The BIE XML Schema File within a namespace will include the corresponding BDT
 1781 XML Schema File that resides in the same namespace.

[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
----------	--------------------------------------------------------------------------------------------------------------------------------------	---

1782 Example 8-8 shows the syntax for including the BDT XML Schema File.

1783 **Example 8-8: Include of BDT XML Schema File**

1784
1785
1786
1787
1788
1789

```

<!-- ===== -->
<!-- ===== Includes ===== -->
<!-- ===== -->
<!-- ===== Include of Business Data Type XML Schema File ===== -->
<!-- ===== -->
<xsd:include schemaLocation="BusinessDataType_lp0.xsd"/>
    
```

1790 Every BIE XML Schema File in a namespace may import a BIE XML Schema File
1791 from another package namespace in order to reuse artefacts defined in the other
1792 namespace.

1793 **8.3.3 Type Definitions**
1794 **8.3.3.1 ABIE Type Definitions**

1795 Every ABIE in a data package is defined as an **xsd:complexType** in the single BIE
1796 XML Schema File for that data package namespace.

[R AF95]	For every object class (ABIE) identified in a data package, a named xsd:complexType MUST be defined in its corresponding BIE XML Schema File.	1
----------	------------------------------------------------------------------------------------------------------------------------------------------------------	---

1797 The name of the **xsd:complexType** will represent the DEN of the BIE.

[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
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1798 The content model of the **xsd:complexType** will be defined such that it reflects
1799 each property of the object class. The content model of the ABIE complex type
1800 definitions will include element declarations for BBIEs, element declarations for
1801 ASBIEs whose **associationKind=composite**, or element references for ASBIEs
1802 whose **associationKind=shared**.

1803 The cardinality and sequencing of each ABIE Property will be determined by the
1804 **Cardinality** and **Sequencing Key** values of the source ABIE.

[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
----------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1805 In defining the content model, both **xsd:sequence** and **xsd:choice** compositors
1806 are allowed.

[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
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1807 When using the **xsd:sequence** and **xsd:choice** content models in a type
 1808 definition their order must be carefully managed. An **xsd:sequence** should not
 1809 contain another **xsd:sequence** directly as there is no additional value. An
 1810 **xsd:choice** should not contain another **xsd:choice** directly as there is no
 1811 additional value. However, it is permissible to interweave **xsd:sequence** and
 1812 **xsd:choice** within a single **xsd:complexType** definition to whatever level of
 1813 nesting is desired.

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

1814 Example 8-9 show an example of using **xsd:sequence** compositor.

1815 **Example 8-9: Sequence compositor within an ABIE type definition**

```

1816 <xsd:complexType name="AccountType" >
1817   <xsd:annotation>
1818     ...see annotation...
1819   </xsd:annotation>
1820   <xsd:sequence>
1821     <xsd:element name="ID" type="IDType"
1822       minOccurs="0" maxOccurs="unbounded">
1823       <xsd:annotation>
1824         ...see annotation...
1825       </xsd:annotation>
1826     </xsd:element>
1827     <xsd:element name="Status" type="bie:StatusType"
1828       minOccurs="0" maxOccurs="unbounded">
1829       <xsd:annotation>
1830         ...see annotation...
1831       </xsd:annotation>
1832     </xsd:element>
1833     <xsd:element name="Name" type="NameType"
1834       minOccurs="0" maxOccurs="unbounded">
1835       <xsd:annotation>
1836         ...see annotation...
1837       </xsd:annotation>
1838     </xsd:element>
1839     ...
1840   </xsd:sequence>
1841 </xsd:complexType>
  
```

1842 Example 8-10 show an example of using **xsd:choice** compositor.

1843 **Example 8-10: Choice compositor within an ABIE type definition**

```

1844 <xsd:complexType name="LocationType">
1845   <xsd:annotation>
1846     ... see annotation ...
1847   </xsd:annotation>
1848   <xsd:choice>
1849     <xsd:element name="GeoCoordinate" type="bie:GeoCoordinateType"
1850       minOccurs="0">
1851       <xsd:annotation>
1852         ... see annotation ...
  
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38

```

        </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>

```

1869 Example 8-11 shows an example of interweaving `xsd:sequence` and `xsd:choice`
1870 compositors.

1871 **Example 8-11: Sequence + Choice compositors within an ABIE type definition**

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55

```

<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="TimeType"
        minOccurs="0">
        ...
      </xsd:element>
      <xsd:element name="EndTime" type="TimeType"
        minOccurs="0">
        ...
      </xsd:element>
    </xsd:sequence>
    <xsd:sequence>
      <xsd:element name="StartDate" type="DateType"
        minOccurs="0">
        ...
      </xsd:element>
      <xsd:element name="EndDate" type="DateType"
        minOccurs="0">
        ...
      </xsd:element>
    </xsd:sequence>
    <xsd:sequence>
      <xsd:element name="StartDateTime"
type="DateTimeType"
        minOccurs="0">
        ...
      </xsd:element>
      <xsd:element name="EndDateTime" type="DateTimeType"
        minOccurs="0">
        ...
      </xsd:element>
    </xsd:sequence>
  </xsd:choice>
</xsd:sequence>
</xsd:complexType>

```

1916 **8.3.3.2 BBIE Type Definitions**

1917 BBIEs are instantiated as local `xsd:element` declarations. The BBIE element is of
 1918 a an `xsd:simpleType` definition or an `xsd:complexType` definition that
 1919 represents its BDT.

[R A21A]	Every BBIE within its containing ABIE MUST be of an <code>xsd:simpleType</code> or <code>xsd:complexType</code> that represents the BDT that defines it.	1
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------	---

1920 **8.3.3.3 ASBIE Type Definitions**

1921 ASBIEs are declared as either a local or global `xsd:element` whose
 1922 `xsd:complexType` is that of the `xsd:complexType` of the associated ABIE it
 1923 represents. No additional type definition is required.

1924 **8.3.4 Element Declarations and References**1925 **8.3.4.1 ABIE Element Declarations**

1926 Every ABIE will have a globally declared element. This global element reflects the
 1927 unique DEN of the ABIE within the namespace to which it is assigned and will be of
 1928 the `xsd:complexType` that represents it.

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

1929 **8.3.4.2 BBIE Element Declarations**

1930 Every BBIE will have a locally declared element that is part of the content model of
 1931 the ABIE to which it belongs.

[R 89A6]	For each BBIE identified in an ABIE, a named <code>xsd:element</code> MUST be locally declared within the <code>xsd:complexType</code> that represents the ABIE.	1
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

1932 The name of the BBIE element will reflect the name of the BBIE devoid of the object
 1933 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
----------	----------------------------------------------------------------------------------------------------------------------	---

- 1934 The BBIE Property name for the representation terms of **Identification**,
1935 **Indicator**, and **Text** are simplified to improve semantic 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 from the property term.	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]	For each BBIE element name declaration where the word Text is the representation term, the word Text MUST be removed from the name of the element or type definition.	1

- 1936 The BBIE element will be of the `xsd:simpleType` or `xsd:complexType` as
1937 defined in [Section 8.3.3.2](#).

[R BCD6]	Every BBIE element declaration MUST be of the BDT <code>xsd:simpleType</code> or <code>xsd:complexType</code> that represents the source BBIE business data type.	1
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

- 1938 Example 8-12 shows an Account. Details ABIE `xsd:complexType` declaration that
1939 contains BBIE element declarations that make use of the appropriate BDT
1940 `xsd:simpleType` or `xsd:complexType`.

1941 Example 8-12: BBIE Element Declaration

1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967

```

<xsd:complexType name="AccountType">
  <xsd:annotation>
    ...see annotation...
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="ID" type="IDType_234DS7"
      minOccurs="0" maxOccurs="unbounded">
      <xsd:annotation>
        ...see annotation...
      </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="NameType_9438SD"
      minOccurs="0" maxOccurs="unbounded">
      <xsd:annotation>
        ...see annotation...
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="BuyerParty" type="bie:BuyerPartyType"/>
  </xsd:sequence>
</xsd:complexType>

```

1968 **8.3.4.3 ASBIE Element Declarations**

1969 For ASBIEs whose `ccts:AggregationKind` value is `composite`, a local element
 1970 for the associated ABIE will be declared in the content model of the associating ABIE
 1971 `xsd:complexType`.

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

1972 For each ASBIE whose `ccts:AggregationKind` value is `shared`, a global
 1973 element is declared. See section [5.5 Reusability Schema](#).

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

1974 The name of the ASBIE local or global element will reflect the name of the ASBIE,
 1975 devoid of the associating ABIE object class term and object class qualifier term(s).

[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
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------	---

1976 The ASBIE local or global element will be of the `xsd:complexType` of the
 1977 associated ABIE.

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

1978 Example 8-13 shows an ABIE type definition with a local element declaration for a
 1979 BBIE `Invoice. Identification. Identifier`, a local element declaration for
 1980 two `AggregationKind` value = `composite` ASBIEs `Invoice. Seller. Party`
 1981 and `Invoice. Buyer. Party`, and a global element reference for the
 1982 `AggregationKind` value = `shared` ASBIE of `Invoice. Trade. LineItem`.
 1983 **Example 8-13: ASBIE element declaration and reference within an ABIE type definition**

```

1984 <xsd:element name="InvoiceTradeLineItem" type="InvoiceTradeLineItemType"/>
1985 <xsd:complexType name="InvoiceType">
1986   <xsd:sequence>
1987     <xsd:element name="ID" type="IDType"/>
1988     <xsd:element name="SellerParty" type="ordman:SellerPartyType"/>
1989     <xsd:element name="BuyerParty" type="ordman:BuyerPartyType"/>
1990     <xsd:element ref="ordman:InvoiceTradeLineItem"
1991     maxOccurs="unbounded" />
1992   </xsd:sequence>

```

1993

1994 **8.3.5 Annotation**1995 **8.3.5.1 ABIE**1996 **8.3.5.1.1 ABIE Complex Type**1997 **8.3.5.1.1.1 Annotation Documentation**

1998 Every ABIE `xsd:complexType` definition must include structured annotation
 1999 documentation.

[R ACB9]	<p>For every ABIE <code>xsd:complexType</code> definition a structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements MUST contain:</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. • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the ABIE. • Definition (mandatory): The semantic meaning of the ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differeniates the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE. • BusinessTermName (optional, repeating): A synonym term in which the ABIE is commonly known. 	1
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2000 Example 8-14 shows the annotation documentation of an ABIE complexType
 2001 definition.

2002 **Example 8-14: ABIE complex type definition annotation**

```

2003 <xsd:complexType name="AccountType" >
2004   <xsd:annotation>
2005     <xsd:documentation xml:lang="en-US">
2006       <ccts:UniqueID>UNBE000000</ccts:UniqueID>
2007       <ccts:VersionID>0.00</ccts:VersionID>
2008       <ccts:DictionaryEntryName>Account</ccts:DictionaryEntryName>
2009       <ccts:Definition>Communicates the Account information.</ccts:Definition>
2010       <ccts:ObjectClassTermName>Account</ccts:ObjectClassTermName>
2011     </xsd:documentation>
2012     <xsd:appInfo>
2013       As shown in Appendix F
2014     </xsd:appInfo>
2015   </xsd:annotation>
2016 </xsd:complexType>

```

2017

2018 **8.3.5.1.1.2 Annotation Application Information**

2019 Every ABIE `xsd:complexType` definition will have a structured set of
 2020 `xsd:annotation xsd:appInfo` information that reflects its context and any
 2021 defined usage rules.

[R B0BA]	For every ABIE <code>xsd:complexType</code> definition a structured set of <code>xsd:annotation xsd:appInfo</code> elements MUST be present that fully declare its context.	1
[R BCE9]	<p>For every ABIE usage rule, the ABIE <code>xsd:complexType</code> definition MUST contain a structured set of <code>xsd:annotation 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

2022 **8.3.5.1.2 ABIE Element**2023 **8.3.5.1.2.1 Annotation Documentation**

2024 Every ABIE element declaration must include structured annotation documentation.

[R 88B6]	<p>For every ABIE <code>xsd:element</code> declaration definition, a structured set of <code>xsd:annotation xsd:documentation</code> elements MUST contain:</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. • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the ABIE. • Definition (mandatory): The semantic meaning of the ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE. • BusinessTermName (optional, repeating): A synonym term in which the ABIE is commonly known. 	1
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2025

2026 **8.3.5.1.2.2 Annotation Application Information**

2027 The global element declaration for ABIEs is used exclusively for referencing by
 2028 ASMAs. Since multiple ASMAs can reference a single global ABIE element
 2029 declaration in different contexts with different usage rules, the context and usage
 2030 rules for global ABIE element declarations can not be explicitly stated in the BIE XML
 2031 Schema File. However, the context and usage rules are stated when the global ABIE
 2032 element is referenced using xsd:ref as part of the content model of the MA.

2033 **8.3.5.2 BBIE Element**

2034 **8.3.5.2.1 Annotation Documentation**

2035 Every BBIE element declaration will include structured annotation documentation.

<p>[R B8BE]</p>	<p>For every BBIE <code>xsd:element</code> declaration a structured set of <code>xsd:annotation xsd:documentation</code> elements MUST contain:</p> <ul style="list-style-type: none"> • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the BBIE. • Definition (mandatory): The semantic meaning of the associated BBIE. • Cardinality (mandatory): Indicates the cardinality of the BBIE within the containing ABIE. • SequencingKey (mandatory): Indicates the sequence of the BBIE within the containing ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE • PropertyTermName (mandatory): Represents a distinguishing characteristic of the BBIE. • PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the BBIE. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • RepresentationTermName (mandatory): An element of the component name that describes the form in which the BBIE is represented. • BusinessTermName (optional, repeating): A synonym term in which the BBIE is commonly known. 	<p>1</p>
-----------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------

2036 Example 8-15 shows the annotation documentation of a BBIE Element.

2037 **Example 8-15: BBIE element annotation**

```

2038 <xsd:element name="ID" type="IDType" minOccurs="0" maxOccurs="unbounded">
2039 <xsd:annotation>
2040 <xsd:documentation xml:lang="en-US">
2041 <cts:DictionaryEntryName>Account. Identificaton.
2042 Identifier</cts:DictionaryEntryName>
2043 <cts:Definition>The Account Identification Identifier.</cts:Definition>
2044 <cts:Cardinality>1</cts:Cardinality>
2045 <cts:SequencingKey>1</cts:SequencingKey>
2046 <cts:ObjectClassQualifierName></cts:ObjectClassQualifierName>
2047 <cts:ObjectClassTermName>Account</cts:ObjectClassTermName>
2048 <cts:PropertyTermName></cts:PropertyTermName>
2049 <cts:PropertyQualifierName></cts:PropertyQualifierName>
2050 <cts:RepresentationTermName></cts:RepresentationTermName>
2051 <cts:BusinessTermName></cts:BusinessTermName>
2052 </xsd:documentation>
2053 <xsd:appInfo>
2054 As shown in Appendix F for context and usage rules
2055 </xsd:appInfo>
2056 </xsd:annotation>
2057 </xsd:element>
    
```

2058 **8.3.5.2.2 Annotation Application Information**

2059 Every BBIE will have structured annotation application information that reflects its
 2060 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 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>cts:UniqueID</code> • <code>cts:Constraint</code> • <code>cts:ConstraintType</code> • <code>cts:ConditionType</code> 	1

2061 **8.3.5.3 ASBIE Element**

2062 **8.3.5.3.1 Global Element Declaration**

2063 **8.3.5.3.1.1 Annotation Documentation**

2064 The global element declaration for `AggregationKind` value = `shared` ASBIEs is
 2065 used exclusively for referencing by associating ABIEs. Since multiple ABIEs can
 2066 reference a single global ASBIE element declaration in different contexts with
 2067 different usage rules, much of the metadata for global ASBIE element declarations
 2068 can not be explicitly stated in the global element declaration and the
 2069 `xsd:annotation` `xsd:documentation` elements will be limited to only that
 2070 metadata that is universally applicable.

<p>[R 8D3E]</p>	<p>Every ASBIE global element declaration MUST have a structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements that contain::</p> <ul style="list-style-type: none"> • UniqueID (mandatory): The unique identifier that identifies an ASBIE instance in a unique and unambiguous way. • VersionID (mandatory): An unique identifier that identifies the version of an ASBIE. • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the ASBIE. • Definition (mandatory): The semantic meaning of the associated ASBIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ASBIE • PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the ASBIE. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • PropertyTermName (mandatory): Represents a distinguishing characteristic of the ASBIE. • AssociationType (mandatory): Indicates the UML AssociationKind value of shared or composite of the associated ABIE. • AssociatedObjectClassQualifierName (optional, repeating): a name or names that qualify the associated object class. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • AssociatedObjectClassName (Mandatory): The name of the associated object class. • BusinessTermName (optional, repeating): A synonym term in which the ASBIE is commonly known. 	<p>1</p>
-----------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------

2071 Example 8-16 shows the annotation documentation of an ASBIE Element. In this
 2072 case the ASBIE is declared as a shared AggregationKind which results in a global
 2073 element.

2074 **Example 8-16: ASBIE global element declaration annotation**

```

2075 <xsd:element name="Country" type="bie:CountryType" minOccurs="0"
2076 maxOccurs="unbounded">
2077   <xsd:annotation>
2078     <xsd:documentation xml:lang="en-US">
2079       <cts:UniqueID>UN00000007</cts:UniqueID>
2080       <cts:Version>3.0</cts:Version>
2081       <cts:DictionaryEntryName>Account. Country</cts:DictionaryEntryName>
2082       <cts:Definition>Country information related to account
2083 details.</cts:Definition>
2084       <cts:Cardinality>0..n</cts:Cardinality>
2085       <cts:SequencingKey>6</cts:SequencingKey>
2086       <cts:ObjectClassTermName>Account</cts:ObjectClassTermName>
2087       <cts:PropertyTermName>Country</cts:PropertyTermName>
2088       <cts:AssociationType>Shared</cts:AssociationType>
2089       <cts:AssociatedObjectClassTermName>Country
2090     </cts:AssociatedObjectClassTermName>
2091     </xsd:documentation>
2092   </xsd:annotation>
2093 </xsd:element>

```

2094 **8.3.5.3.1.2 Annotation ApplInfo**

2095 The global element declaration for **AggregationKind** value = **shared** ASBIEs is
 2096 used exclusively for referencing by associating ABIEs. Since multiple ABIEs can
 2097 reference a single global ASBIE element declaration in different contexts with
 2098 different usage rules, no context values or usage rules will be defined. Context and
 2099 usage rules can be stated when the global ASBIE element is referenced using
 2100 **xsd:ref** as part of the content model of the ABIE. See section [8.3.5.3.2.2](#)
 2101 [Annotation Application Information](#).

2102 **8.3.5.3.2 Local Element Declaration and Global Element References**2103 **8.3.5.3.2.1 Annotation Documentation**

2104 ASBIEs declared locally, and every **xsd:ref** occurrence of an ASBIE declared
 2105 globally, will include structured annotation documentation. Every ASBIE local
 2106 element declaration or **xsd:ref** occurrence in the content model of an ABIE will
 2107 include structured annotation documentation.

<p>[R 926A]</p>	<p>Every ASBIE <code>xsd:element</code> declaration or <code>xsd:ref</code> occurrence within the containing ABIE MUST have a structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements that contain:</p> <ul style="list-style-type: none"> • UniqueID (mandatory): The unique identifier that identifies an ASBIE instance in a unique and unambiguous way. • VersionID (mandatory): An unique identifier that identifies the version of an ASBIE. • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the ASBIE. • Definition (mandatory): The semantic meaning of the associated ASBIE. • Cardinality (mandatory): Indicates the cardinality of the ASBIE within the containing ABIE. • SequencingKey (mandatory): Indicates the sequence of the ASBIE within the containing ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ASBIE • PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the ASBIE. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • PropertyTermName (mandatory): Represents a distinguishing characteristic of the ASBIE. • AssociationType (mandatory): Indicates the UML AssociationKind value of shared or composite of the associated ABIE. • AssociatedObjectClassQualifierName (optional, repeating): a name or names that qualify the associated object class. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • AssociatedObjectClassName (Mandatory): The name of the associated object class. • BusinessTermName (optional, repeating): A synonym term in which the ASBIE is commonly known. 	<p>1</p>
-----------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------

2108 Example 8-17 shows the annotation documentation of an ASBIE whose
2109 **aggregationKind=Composite** and is locally declared.

2110 **Example 8-17: ASBIE local element declaration annotation**

```

2111 <xsd:element name="Country" type="bie:CountryType" minOccurs="0"
2112 maxOccurs="unbounded">
2113   <xsd:annotation>
2114     <xsd:documentation xml:lang="en-US">
2115       <ccts:UniqueID>UN00000007</ccts:UniqueID>
2116       <ccts:Version>3.0</ccts:Version>
2117       <ccts:DictionaryEntryName>Account. Country</ccts:DictionaryEntryName>
2118       <ccts:Definition>Country information related to account
2119 details.</ccts:Definition>
2120       <ccts:Cardinality>0..n</ccts:Cardinality>
2121       <ccts:SequencingKey>6</ccts:SequencingKey>
2122       <ccts:ObjectClassTermName>Account</ccts:ObjectClassTermName>
2123       <ccts:PropertyTermName>Country</ccts:PropertyTermName>
2124       <ccts:AssociationType>Composite</ccts:AssociationType>
2125       <ccts:AssociatedObjectClassTermName>Country
2126 </ccts:AssociatedObjectClassTermName>
2127     </xsd:documentation>
2128   </xsd:annotation>
2129 </xsd:element>

```

2130 Example 8-18 shows the annotation documentation of a reference to an ASBIE
2131 Element.

2132 **Example 8-18. ASBIE element REF annotation**

```

2133 <xsd:element ref="Country" type="bie:CountryType" minOccurs="0"
2134 maxOccurs="unbounded">
2135   <xsd:annotation>
2136     <xsd:documentation xml:lang="en-US">
2137       <ccts:UniqueID>UN00000007</ccts:UniqueID>
2138       <ccts:Version>3.0</ccts:Version>
2139       <ccts:DictionaryEntryName>Account. Country</ccts:DictionaryEntryName>
2140       <ccts:Definition>Country information related to account
2141 details.</ccts:Definition>
2142       <ccts:Cardinality>0..n</ccts:Cardinality>
2143       <ccts:SequencingKey>6</ccts:SequencingKey>
2144       <ccts:ObjectClassTermName>Account</ccts:ObjectClassTermName>
2145       <ccts:PropertyTermName>Country</ccts:PropertyTermName>
2146       <ccts:AssociationType>Composite</ccts:AssociationType>
2147       <ccts:AssociatedObjectClassTermName>Country
2148 </ccts:AssociatedObjectClassTermName>
2149     </xsd:documentation>
2150     <xsd:appInfo>
2151       As shown in Appendix F for context and usage rules
2152     </xsd:appInfo>
2153   </xsd:annotation>
2154 </xsd:element>

```

2155 **8.3.5.3.2 Annotation Application Information**

2156 Every ASBIE **xsd:element** local declaration or **xsd:ref** occurrence in the content
2157 model of an ABIE will have structured annotation application information that reflects
2158 its context and any defined usage rules.

[R 9D87]	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.	1
----------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

[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
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2159 8.4 Business Data Type XML Schema Files

2160 Multiple BDT XML Schema Files are created in the UN/CEFACT modularity model.
 2161 One Reference BDT XML Schema File will be created that contains default BDTs for
 2162 all approved CDTs published in the UN/CEFACT DT catalogue. An additional BDT
 2163 XML Schema File will be created for each data package namespace that defines all
 2164 BDTs used in that namespace. The BDT XML Schema File names must follow the
 2165 UN/CEFACT XML Schema File naming approach.

2166 8.4.1 Use of Business Data Type XML Schema Files

2167 The Reference BDT XML Schema File is not included as part of the modularity
 2168 model as it is intended to be used as a reference template for schema developers.
 2169 The data package BDT XML Schema File will be used by the BIE XML Schema File
 2170 and all Root Element XML Schema Files defined in the same data package
 2171 namespace.

2172 8.4.2 XML Schema Structure

2173 Each BDT XML Schema File will be structured in a standard format to ensure
 2174 consistency and ease of use.

2175 The format is shown in Example 8-19. Each BDT XML Schema File must adhere to
 2176 the format of the relevant sections as detailed in [Appendix B](#).

2177 Example 8-19: BDT XML Schema file structure

2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200

```

<?xml version="1.0" encoding="utf-8"?>
<!-- ===== -->
<!-- ===== Business Data Type XML Schema File ===== -->
<!-- ===== -->
<!--
Schema agency:      UN/CEFACT
Schema version:    3.0
Schema date:       14 July 2009

Copyright (C) UN/CEFACT (2009). All Rights Reserved.

... see copyright information ...

-->
<xsd:schema targetNamespace=
... see namespace ...
xmlns:xsd="http://www.w3.org/2001/XMLSchema"

```

2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210

```

elementFormDefault="qualified" attributeFormDefault="unqualified">
<!-- ===== Includes ===== -->
<!-- ===== Imports ===== -->
... see imports ...
<!-- ===== Type Definitions ===== -->
... see type definitions ...
</xsd:schema>
    
```

2211 **8.4.3 Imports and Includes**

2212 Each BDT XML Schema File will use **xsd:include** to make use of any BCL XML
 2213 Schema Files and BIS XML Schema Files being used by the BDT XML Schema
 2214 Components. Each BDT XML Schema File will use **xsd:import** to make use of the
 2215 XBT XML Schema File, any CCL XML Schema Files and any CIS XML Schema
 2216 Files being used by a BDT within the BDT XML Schema File.

[R 8E0D]	Each BDT XML Schema File MUST include (xsd:include) all BCL XML Schema Files and BIS XML Schema Files that are defined in the same namespace.	1
[R B4C0]	Each BDT XML Schema File MUST import (xsd:import) the XBT XML Schema File, and each CCL XML Schema File and CIS XML Schema File that is used by BDTs contained within the BDT XML Schema File.	1

2217 **8.4.4 Type Definitions**

2218 BDT XML Schema Components are defined as either an **xsd:complexType** or
 2219 **xsd:simpleType**.

[R AE00]	Each BDT 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 for that namespace.	1
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2220 A BDT type name reflects the data type qualifiers and data type term and a six
 2221 character unique identifier. The six character identifier is unique within the
 2222 namespace to which it occurs.

[R A7B8]	<p>The name of a BDT MUST be the:</p> <ul style="list-style-type: none"> • BDT <code>ccts:DataTypeQualifierTerm(s)</code> if any, plus. • The <code>ccts:DataTypeTerm</code>, plus. • The word <code>Type</code>, plus. • The underscore character [<code>_</code>], plus. • A six character unique identifier, unique within the given namespace, consisting of lowercase alphabetic characters [a-z], uppercase alphabetic characters [A-Z], and digit characters [0-9]. <p>All separators are removed and approved abbreviations and acronyms are applied.</p>	1
[R 8437]	The six character unique identifier used for the BDT Type name MUST be unique within the namespace in which it is defined.	1

2223 Example 8-20 provides examples of BDT names.

2224 Example 8-20 BDT Type Definition Names

```

2225 CodeType_000001
2226 Where Code is the Data Type Term and 000001 is the six character unique identifier
2227
2228 PercentType_000005
2229 Where Percent is the Data Type Term and 000005 is the six character unique
2230 identifier.
2231
2232 AstronomicalUnitValueType_ABDEC1
2233 Where Astronomical Unit is the Data Type Qualifier, Value is the Data Type Term,
2234 and ABDEC1 sthe six character unique identifier.

```

2235 [Note:]

2236 The six character unique identifier does not have to be sequential.

2237 [Note:]

2238 This naming convention is the same regardless if the BVD is a primitive, a code list,
2239 multiple code lists, or an identifier scheme.

2240 As defined in the Data Type Catalogue a BDT content component BVD can contain
2241 either a set of primitives or a code list or point to an identifier scheme. This means
2242 that a data type can be defined to have one of several possible primitives or one or
2243 more possible code lists or one or more possible identifier schemes. When the BDT
2244 `xsd:simpleType` or `xsd:complexType` is defined in the BDT XML Schema File,
2245 it will be defined to reflect a single primitive, single code list, the list of code list
2246 combinations, or a single identifier scheme. The modeller chooses which of these
2247 combinations is used when they identify the specific BDT, primitive, code list or
2248 identifier scheme to use for a BBIE.

2249 For the Date, Time and DateTime BDTs the content component BVD may need to
2250 support variable precision beyond what is possible the corresponding XSD built-in
2251 data types. In cases where the model must support multiple formats for these BDTs,

2252 a **formatCode** attribute maybe used to indicate the format of the content, if and only
 2253 if the format is not the default. However, it is recommended that where possible a
 2254 specific format be specified by the BDT.

[R B43E]	When a BDT for Date, Time, and DateTime needs to support variable precision beyond what is possible within the XML Schema types, the BDT MUST use a formatCode attribute to indicate the format of the content, if and only if the format is not the default.	1
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2255 The **formatCode** attribute must be defined by a code list that defines the formats
 2256 allowed.

[R 9B37]	All formatCode attributes for the BDTs Date, Time or DateTime MUST define the formats allowed for the BDT.	1
----------	-------------------------------------------------------------------------------------------------------------------	---

2257 8.4.4.1 BDT Simple Type Definitions

2258 If a BDT has no Supplementary Components it is defined as an **xsd:simpleType**.
 2259 If a BDT has Supplementary Components that map directly to the facets of an XML
 2260 Schema built-in datatype, it is defined as an **xsd:simpleType**. If a BDT has
 2261 Supplementary Components whose BVD does not map directly to the facets of an
 2262 XML Schema built-in datatype, it is defined as an **xsd:complexType** (See Section
 2263 8.4.4.2 BDT Complex Type Definitions).

[R 9908]	Every BDT devoid of ccts:supplementaryComponents , or whose ccts:supplementaryComponents BVD facets map directly to the facets of an XML Schema built-in data type, MUST be defined as a named xsd:simpleType .	1
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2264 8.4.4.1.1 Content Component Business Value Domain Expressed By Primitives

2265 When a BDT Content Component BVD is defined by a primitive, and the primitive
 2266 facets are supported by the facets of an XSD built-in data type, the BDT
 2267 **xsd:simpleType** will have an **xsd:restriction** element whose **base** attribute
 2268 is set to the XSD built-in **xsd:simpleType** that represents the primitive.

[R B91F]	The xsd:simpleType definition of a BDT whose content component BVD is defined by a primitive whose facets map directly to the facets of an XML Schema built-in datatype MUST contain an xsd:restriction element with the base attribute set to the XML Schema built-in data type that represents the primitive.	1
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2269 Example 8-21 shows a BDT **xsd:simpleType** that uses an **xsd:integer** built-in
 2270 type to define the Content Component BVD with no Supplementary Components.

2271 **Example 8-21: BDT Simple Type Definition where Content Component BVD is expressed by a**
 2272 **primitive and no Supplementary Component attributes**

```

2273 <xsd:simpleType name="OrdinalType_56473">
2274 <xsd:annotation>
2275 ... see annotation ...
2276 </xsd:annotation>
2277 <xsd:restriction base="xsd:integer"/>
2278 </xsd:simpleType>

```

2279 When a BDT Content Component BVD is defined by a primitive, and the primitive
 2280 facets are not supported by the facets of an XML Schema built-in data type, the BDT
 2281 will be defined as an **xsd:complexType** (See Section 8.4.4.2 BDT Complex Type
 2282 Definitions).

2283 8.4.4.1.2 Content Component Business Value Domain Expressed By A Single Code 2284 List

2285 When a BDT content component BVD is defined by a single code list (BCL or CCL),
 2286 the BDT is defined as an **xsd:simpleType** that contains an **xsd:restriction**
 2287 element whose **base** attribute is set to the defined **xsd:simpleType** for the code
 2288 list (See section [8.6.1.4 Type Definitions](#)).

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

2289 Example 8-22 shows a BDT **xsd:simpleType** declaration using a code list to
 2290 define the Content Component BVD.

2291 **Example 8-22: BDT type definition using one code list to define the BVD**

```

2292 <xsd:simpleType name="TemperatureMeasureTypeCodeType_1AZS2B">
2293 <xsd:annotation>
2294 ... see annotation ...
2295 </xsd:annotation>
2296 <xsd:restriction
2297 base="clm6Recommendation20:MeasurementUnitCommonCodeContentType">
2298 <xsd:length value="3"/>
2299 </xsd:restriction>
2300 </xsd:simpleType>

```

2301 8.4.4.1.3 Content Component Business Value Domain Expressed By Multiple Code 2302 Lists

2303 When a BDT Content Component BVD is defined by two or more code lists (BCL or
 2304 CCL), the BDT is defined as an **xsd:simpleType** that contains an
 2305 **xsd:restriction** element whose **base** attribute is set to the defined
 2306 **xsd:simpleType** of a BCL that unions all of the possible code lists together (See
 2307 [Section 8.6.3.4.3 Combining Multiple Code Lists](#)).

2308

2309 8.4.4.1.4 Content Component Business Value Domain Expressed By An Identifier 2310 Scheme

2311 When a BDT Content Component BVD is defined by an identifier scheme (BIS or
2312 CIS), the BDT is defined as an **xsd:simpleType** that contains an
2313 **xsd:restriction** element whose **base** attribute is set to the identifier scheme
2314 defined **xsd:simpleType** (See Section [7.3.1 Simple Type Definitions](#)).

[R A861]	The xsd:simpleType definition of a BDT whose content component BVD is defined by an identifier scheme MUST contain an xsd:restriction element with the base attribute set to the identifier scheme's defined xsd:simpleType .	1
----------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2315 Example 8-23 shows an BDT **xsd:simpleType** definition using an identifier
2316 scheme to define the Content Component BVD.

2317 Example 8-23: BDT type definition using an identifier scheme to define the BVD

```

2318 <xsd:simpleType name="SocialSecurityIdentifierType_5647X3">
2319   <xsd:annotation>
2320     ... see annotation ...
2321   </xsd:annotation>
2322   <xsd:restriction base="ism244SSN:SocialSecurityNumberContentType">
2323     <xsd:length value="9" />
2324   </xsd:restriction>
2325 </xsd:simpleType>

```

2326 8.4.4.2 BDT Complex Type Definitions

2327 Supplementary Components refine the BDT Content Component by providing
2328 additional information. Every BDT has zero or more Supplementary Components. If
2329 a BDT has Supplementary Components, and those Supplementary Components
2330 map directly to the facets of an XML Schema built-in datatype, the BDT is defined as
2331 an **xsd:simpleType** (See Section 8.4.4.1 BDT SimpleType Definitions). If a BDT
2332 has Supplementary Components, and those Supplementary Components do not
2333 map directly to the facets of an XML Schema built-in datatype, the BDT will be
2334 defined as an **xsd:complexType** with **xsd:simpleContent** and an
2335 **xsd:extension** element whose **base** attribute is set to either a primitive type or an
2336 identifier scheme or a code list or union of code lists. Each Supplementary
2337 Component is expressed as an **xsd:attribute** declaration whose **name** is set to
2338 the DEN of the given Supplementary Component.

[R AB05]	Every BDT that includes one or more Supplementary Components that do not map directly to the facets of an XSD built-in datatype MUST be defined as an xsd:complexType .	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 the Supplementary Component Property Term Name and Representation Term Name with periods, spaces, and other separators removed.	1
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2339 Example 8-24 shows an example of a BDT with a Supplementary Component whose
 2340 BVD is defined by a code list.

2341 **Example 8-24: Business Data type with a Supplementary Component BVD defined by a code**
 2342 **list**

2343
 2344
 2345
 2346
 2347
 2348
 2349
 2350
 2351
 2352
 2353
 2354
 2355
 2356
 2357

```

<xsd:complexType name="AmountType_SDC90X">
  <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>
    
```

2358 [8.4.4.2.1 Content Component Business Value Domain Expressed By Primitives](#)

2359 When a BDT Content Component BVD is defined by a primitive, and the primitive
 2360 facets are not directly supported by the facets of an XSD built-in data type, the BDT
 2361 **xsd:complexType** will contain an **xsd:simpleContent** element that will contain
 2362 an **xsd:extension** element whose **base** attribute is set to the XSD built-in
 2363 **xsd:simpleType** that represents the primitive.

[R BBCB]	The xsd:complexType definition of a BDT whose Content Component BVD is defined by a primitive whose facets do not map directly to the facets of an XML Schema built-in datatype MUST contain an xsd:simpleContent element that contains an xsd:extension whose base attribute is set to the XML Schema built-in data type that represents the primitive.	1
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2364 Example 8-25 shows an example of a complex BDT with a Content Component
 2365 whose BVD is defined by a primitive.
 2366

2367 **Example 8-25: BDT Complex Type Definition where the Content Component BVD is expressed**
 2368 **by a primitive with Supplementary Component attributes**

```

2369 <xsd:complexType name="AmountType_SDC90X">
2370   <xsd:annotation>
2371     ... see annotation ...
2372   </xsd:annotation>
2373   <xsd:simpleContent>
2374     <xsd:extension base="xsd:decimal">
2375       <xsd:attribute name="currencyCode"
2376         type="clm54217:CurrencyCodeContentType" use="optional">
2377         <xsd:annotation>
2378           ... see annotation ...
2379         </xsd:annotation>
2380       </xsd:attribute>
2381     </xsd:extension>
2382   </xsd:simpleContent>
2383 </xsd:complexType>
  
```

2384 8.4.4.2.2 Content Component Business Value Domain Expressed By A Single Code 2385 List

2386 When a BDT Content Component BVD is defined by a single code list (BCL or CCL),
 2387 the BDT is defined as an **xsd:complexType** that will contain an
 2388 **xsd:simpleContent** element that will contain an **xsd:extension** element
 2389 whose **base** attribute is set to the defined **xsd:simpleType** for the code list ([See](#)
 2390 [Section 8.6.1.4 Type Definitions](#)).

[R BD8E]	The xsd:complexType definition of a BDT whose Content Component BVD is defined as a single code list MUST contain an xsd:simpleContent element that contains an xsd:extension element whose base attribute is set to the defined xsd:simpleType for the code list.	1
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2391 8.4.4.2.3 Content Component Business Value Domain Expressed By Multiple Code 2392 Lists

2393 When a BDT Content Component BVD is defined by two or more code lists (BCL or
 2394 CCL), the BDT is defined as an **xsd:complexType** that will contain an
 2395 **xsd:simpleContent** element that will contain an **xsd:extension** element
 2396 whose **base** attribute is set to the defined **xsd:simpleType** of a BCL that unions
 2397 all of the possible code lists together ([See Section 8.6.3.4.3 Combining Multiple](#)
 2398 [Code Lists](#)).

2399 8.4.4.2.4 Content Component Business Value Domain Expressed By An Identifier 2400 Scheme

2401 When a BDT Content Component BVD is defined by an identifier scheme (BIS or
 2402 CIS), the BDT is defined as an **xsd:complexType** that will contain an
 2403 **xsd:simpleContent** element that will contain an **xsd:extension** element
 2404 whose **base** attribute is set to the identifier scheme defined **xsd:simpleType** (See
 2405 [Section 7.3.1 Simple Type Definitions](#)).

[R 91E8]	The xsd:complexType definition of a BDT whose Content Component BVD is defined by an identifier scheme MUST contain an xsd:simpleContent element that contains an xsd:extension whose base attribute is set to the identifier scheme's defined xsd:simpleType .	1
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2406 **8.4.4.3 BDT Restrictions**

2407 BDTs may have either their content component, and/or supplementary component
 2408 restricted. At the data model level, restrictions can take the form of restrictions to the
 2409 Business Value Domain (BVD) of the BDT content component or supplementary
 2410 component. Restrictions can also take the form of restrictions to the cardinality of the
 2411 BDT supplementary component – to include the presence or absence of the
 2412 supplementary component. Restrictions to the BVD can be in the form of restrictions
 2413 to the primitive facets or to the scheme or list used to define the content component
 2414 or supplementary component BVD.

2415 At the XML level, restrictions can take the form of restrictions to the BDT content
 2416 component BVD. This is accomplished by creating a new restricted BDT
 2417 **xsd:simpleType** or **xsd:complexType** that is derived from the less restricted or
 2418 unrestricted BDT **xsd:simpleType** or **xsd:complexType**. Restrictions can also take the
 2419 form of restrictions to the occurrence of a supplementary component attribute.

[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
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2420 Non-standard variations are defined as those that are outside the bounds of the
 2421 normally defined BVD for the underlying BDT. If non-standard variations from the
 2422 base type are required, these will be defined as an **xsd:restriction** derivation
 2423 from a custom type.

[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
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2424 [Note:]
 2425 If a non-standard variation of the standard date time built-in data types is required,
 2426 for example year month, then a BDT of the Core Data Type **TextType** needs to be
 2427 defined, with the appropriate restrictions specified, e.g. a pattern, to specify the
 2428 required format.

2429 Example 8-26 shows a restricted BDT definition.

2430

2431 **Example 8-26: Restricted BDT Type Definitions**

```

2432 <!-- ===== Type Definitions ===== -->
2433 <!-- ===== Type Definitions ===== -->
2434 <!-- ===== Business Data Type based on DateTime Type ===== -->
2435 <!-- ===== Business Data Type based on DateTime Type ===== -->
2436 <!-- ===== Day_Date. Type ===== -->
2437 <!-- ===== Day_Date. Type ===== -->
2438 <!-- ===== Day_Date. Type ===== -->
2439 <xsd:simpleType name="DayDateType_SADF54">
2440 <xsd:annotation>
2441 <!-- ... see annotation ... -->
2442 </xsd:annotation>
2443 <xsd:restriction base="xsd:gDay"/>
2444 </xsd:simpleType>
2445 ...
2446 <!-- ===== Description_Text. Type ===== -->
2447 <!-- ===== Description_Text. Type ===== -->
2448 <!-- ===== Description_Text. Type ===== -->
2449 <xsd:complexType name="DescriptionTextType_X4B81X">
2450 <xsd:annotation>
2451 <!-- ... see annotation ... -->
2452 </xsd:annotation>
2453 <xsd:simpleContent>
2454 <xsd:restriction base="TextType_VCX675"/>
2455 </xsd:simpleContent>
2456 </xsd:complexType>
2457 ...
2458 <!-- ===== Country_Identifier. Type ===== -->
2459 <!-- ===== Country_Identifier. Type ===== -->
2460 <!-- ===== Country_Identifier. Type ===== -->
2461 <xsd:simpleType name="CountryIDType_09456">
2462 <xsd:annotation>
2463 <!-- ... see annotation ... -->
2464 </xsd:annotation>
2465 <xsd:restriction base="ids53166:CountryCodeContentType"/>
2466 </xsd:simpleType>
2467 ...

```

2468 **8.4.4.3.1 Restrictions to Content Component**

2469 Restrictions to the content component result in the creation of a new qualified BDT
 2470 through restriction to the allowed `ccts:ContentComponent` and/or
 2471 `ccts:SupplementaryComponent` primitive facets of the unrestricted BDT type
 2472 definition, or through restrictions to the common code list, business code list,
 2473 common identifier scheme or business identifier scheme used to define the BVD
 2474 when those are used in lieu of a primitive.

2475 **8.4.4.3.2 Restrictions to Supplementary Component**

2476 Restrictions to the supplementary component result in the creation of a new qualified
 2477 BDT through restriction to the allowed `ccts:ContentComponent` and/or
 2478 `ccts:SupplementaryComponent` primitive facets of the unrestricted BDT type
 2479 definition, or through restrictions to the common code list, business code list,
 2480 common identifier scheme or business identifier scheme used to define the BVD
 2481 when those are used in lieu of a primitive.

2482 **8.4.5 BDT Attribute and Element Declarations**

2483 There are no element declarations in the BDT XML Schema Files. The only allowed
 2484 attributes are the defined Supplementary Components or the `formatCode` attribute

2485 for the **Date. Type**, **Date Time. Type**, and **Time. Type**. All attributes are
 2486 defined locally.

[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 or the formatCode attribute for Date. Type , Date Time Type , and Time. Type .	1

2487 **8.4.6 BDT Annotations**
 2488 **8.4.6.1 Annotation Documentation**
 2489 **8.4.6.1.1 BDT Types**

2490 Every BDT element declaration and type definition must include structured
 2491 annotation documentation.

[R BFE5]	<p>Every BDT XML Schema type definition MUST contain a structured set of xsd:annotation xsd:documentation elements that contain:</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. • DataTypeTermName (mandatory): The name of the DataType. The possible values for the DataType are defined in the Data Type Catalogue. • DataTypeQualifierTerm Name (optional, repeating): Is a word or words which help define and differentiate a Data Type. It further enhances the semantic meaning of the DataType. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. 	1
----------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2492 Example 8-27 shows the annotation documentation structure declaration for BDT.
 2493

2494 **Example 8-27: BDT annotation documentation definition**

```

2495 <xsd:group name="BDTDocumentation">
2496   <xsd:sequence>
2497     <xsd:element name="UniqueID" type="EntityUniqueIDType_76U810"/>
2498     <xsd:element name="VersionID" type="VersionIDType_0192SK"/>
2499     <xsd:element name="DictionaryEntryName" type="NameType_4392S1"/>
2500     <xsd:element name="Definition" type="TextType_SDF657"/>
2501     <xsd:element name="BusinessTermName" type="NameType_43921S"
2502     minOccurs="0" maxOccurs="unbounded"/>
2503     <xsd:element name="DataTypeTermName" type="NameType_43921S"/>
2504     <xsd:element name="DataTypeQualifierTermName"
2505     type="NameType_43921S" minOccurs="0"/>
2506   </xsd:sequence>
2507 </xsd:group>
    
```

2508 Example 8-28 shows an example annotation documentation of a BDT.

2509 **Example 8-28: BDT type definition annotation element**

```

2510 ... see type definition ...
2511 <xsd:annotation>
2512   <xsd:documentation xml:lang="en">
2513     <ccts:UniqueID>BDT000027</ccts:UniqueID>
2514     <ccts:VersionID>1.0</ccts:VersionID>
2515     <ccts:DictionaryEntryName>Loss_ Quantity. Type</ccts:DictionaryEntryName>
2516     <ccts:Definition>A loss quantity is a counted number of non-monetary
2517     units, possibly including fractions that represents the difference between te book
2518     quantity and the actual quantity</ccts:Definition>
2519     <ccts:DataTypeTermName>Quantity</ccts:DataTypeTermName>
2520     <ccts:DataTypeQualifierTermName>Loss</ccts:DataTypeQualifierTermName>
2521   </xsd:documentation>
2522 </xsd:annotation>
2523 ... see type definition ...
    
```

2524 **8.4.6.1.1 BDT Type Content Component Business Value Domain**

2525 Every BDT type declaration must include structured annotation documentation within
 2526 the Content Component `xsd:simpleContent` element.

<p>[R 8095]</p>	<p>Every BDT <code>xsd:simpleContent</code> element MUST contain a structured set of ContentComponentValueDomain <code>xsd:annotation</code> <code>xsd:documentation</code> elements that contain:</p> <ul style="list-style-type: none"> • Definition (mandatory): The semantic meaning of the BDT. • DefaultIndicator (mandatory): Indicates if the primitive, scheme or list is the default BVD for the data type. • PrimitiveTypeName (optional): The primitive type of the BDT Content Component. One of PrimitiveTypeName, or SchemeOrListID must be present. • SchemeOrListID (optional): The unique identifier assigned to the scheme or list that uniquely identifies it. One of PrimitiveTypeName or SchemeOrListID must be present. • SchemeOrListVersionID (optional): The version of the scheme or list. Must be present if SchemeOrListID is present. • SchemeOrListAgencyID (optional): The unique identifier assigned to the Agency that owns or is responsible for the 	<p>1</p>
-----------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------

	<p>Scheme or Code List being referenced. Must be present if SchemeOrListID is present.</p> <ul style="list-style-type: none"> • SchemeOrListModificationAllowedIndicator (optional): Indicates whether the Identifier Scheme or Code List can be modified. • DefaultValue (optional): The default value for the BDT Content Component. 	
--	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--

2527 Example 8-29 shows the annotation documentation structure declaration for each
2528 BDT Content Component.

2529 **Example 8-29: BDT Content Component BVD annotation documentation definition**

```

2530 <xsd:group name="ContentComponentValueDomain">
2531   <xsd:sequence>
2532     <xsd:element name="Definition" type="TextType_SDF657"/>
2533     <xsd:element name="DefaultIndicator" type="IndicatorType_V5C6X7"/>
2534
2535     <xsd:element name="PrimitiveTypeName" type=" NameType_43921S"
2536     minOccurs="0"
2537
2538     <xsd:element name="SchemeOrListID" type=" IDType_LKI4DX"
2539     minOccurs="0" />
2540
2541     <xsd:element name="SchemeOrListVersionID" type=" IDType_LKI4DX"
2542     minOccurs="0" />
2543
2544     <xsd:element name="SchemeOrListAgencyID" type=" IDType_LKI4DX"
2545     type="IndicatorType_V5C6X7" minOccurs="0" />
2546     <xsd:element name="DefaultValue" type="TextType_6589AZ"
2547     minOccurs="0" />
2548   </xsd:sequence>
</xsd:group>

```

2549 Example 8-30 shows an example annotation documentation of a BDT Content
2550 Component.

2551 **Example 8-30: BDT Content Component annotation element**

```

2552 ... see type definition ...
2553 <xsd:annotation>
2554   <xsd:documentation>
2555     <ccts:ContentComponentValueDomain>
2556       <ccts:Definition>A number of monetary units</ccts:Definition>
2557       <ccts:DefaultIndicator>True</ccts:DefaultIndicator>
2558       <ccts:PrimitiveTypeName>Decimal</ccts:PrimitiveTypeName>
2559     </ccts:ContentComponentValueDomain>
2560   </xsd:documentation>
2561 </xsd:annotation>
2562 ... see type definition ...

```

2563 8.4.6.1.2 BDT Type Supplementary Components

2564 Every BDT Supplementary Component attribute declaration must include structured
2565 annotation documentation.

[R 9C95]	<p>Every BDT Supplementary Component xsd:attribute declaration MUST contain a structured set of xsd:annotation xsd:documentation elements that contain:</p> <ul style="list-style-type: none"> • Cardinality (mandatory): Indicates the cardinality of the SC within the containing BDT. • DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the BDT SC. • Definition (mandatory): The semantic meaning of the BDT SC. • 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. • DataTypeTermName (mandatory): The name of the DataType Term. The possible values for the DataType Term are defined in the Data Type Catalogue. • DataTypeQualifierTermName (mandatory): A word or words which help define and differentiate a Data Type. It further enhances the semantic meaning of the DataType. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. 	1
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2566 Example 8-31 shows the annotation documentation definition for each BDT SC.

2567 **Example 8-31: BDT SC annotation documentation definition**

2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580

```

<xsd:group name="BDTSCDocumentation">
  <xsd:sequence>
    <xsd:element name="Cardinality" type="OrdinalType_1241SS"/>
    <xsd:element name="DictionaryEntryName" type="NameType_43921S"/>
    <xsd:element name="Definition" type="TextType_SDF657"/>
    <xsd:element name="PropertyTermName" type="NameType_43921S"/>
    <xsd:element name="RepresentationTermName"
type="NameType_43921S"/>
    <xsd:element name="DataTypeName" type="NameType_43921S"/>
    <xsd:element name="DataTypeQualifierTermName"
type="NameType_43921S"/>
  </xsd:sequence>
</xsd:group>

```

2581 **8.4.6.1.2.1 BDT Type Supplementary Component Business Value Domain**

2582 Every BDT Supplementary Component attribute declaration must also include within
2583 the structured annotation documentation a structure for the Supplementary
2584 Component BVD.

[R 91C3]	<p>Every Supplementary Component <code>xsd:attribute</code> declaration MUST contain within the structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements a containing <code>SupplementaryComponentValueDomain</code> element that contains:</p> <ul style="list-style-type: none"> • <code>DefaultIndicator</code> (mandatory): Indicates if the primitive, scheme or list is the default BVD for the data type. • <code>PrimitiveTypeName</code> (optional): The primitive type of the BDT Supplementary Component. One of <code>PrimitiveTypeName</code> or <code>SchemeOrListID</code> must be present. • <code>SchemeOrListID</code> (optional): The unique identifier assigned to the scheme or list that uniquely identifies it. One of <code>PrimitiveTypeName</code> or <code>SchemeOrListID</code> must be present. • <code>SchemeOrListVersionID</code> (optional): The version of the scheme or list. Must be present if <code>SchemeOrListID</code> is present. • <code>SchemeOrListAgencyID</code> (optional): The unique identifier assigned to the Agency that owns or is responsible for the Scheme or Code List being referenced. Must be present if <code>SchemeOrListID</code> is present. • <code>SchemeOrListModificationAllowedIndicator</code> (optional): Indicates whether the Identifier Scheme or Code List can be modified. • <code>DefaultValue</code> (optional): Is the default value. 	1
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2585 Example 8-32 shows the annotation documentation definition for each BDT SC BVD
2586 and an example BDT SC annotation documentation.

2587 **Example 8-32: BDT SC annotation documentation definition**

```

2588 <xsd:complexType name="SupplementaryComponentValueDomainType">
2589   <xsd:sequence>
2590     <xsd:element name="DefaultIndicator" type="IndicatorType_V5C6X7"/>
2591     <xsd:element name="PrimitiveTypeName" type="NameType_43921S"
2592 minOccurs="0"/>
2593     <xsd:element name="SchemeOrListID" type="IDType_LKI4DX" minOccurs="0"/>
2594     <xsd:element name="SchemeOrListVersionID" type="IDType_LKI4DZ"
2595 minOccurs="0"/>
2596     <xsd:element name="SchemeOrListAgencyID" type="IDType_LKI4DX"
2597 minOccurs="0"/>
2598     <xsd:element name="SchemeOrListModificationAllowedIndicator"
2599 type="IndicatorType_V5C6X7" minOccurs="0"/>
2600     <xsd:element name="DefaultValue" type="TextType_6589AZ" minOccurs="0"/>
2601   </xsd:sequence>
2602 </xsd:complexType>

```

2603 Example 8-33 shows an example BDT SC annotation documentation.

2604

2605 **Example 8-33: BDT SC annotation documentation**

```

2606 <xsd:attribute name="currencyCode"
2607 type="clm542173A20090305:ISO3AlphaCurrencyCodeContentType" use="optional">
2608 <xsd:annotation>
2609 <xsd:documentation xml:lang="en">
2610 <ccts:Cardinality>0..1</ccts:Cardinality>
2611 <ccts:DictionaryEntryName>Amount. Currency.
2612 Code</ccts:DictionaryEntryName>
2613 <ccts:Definition>The currency of the amount</ccts:Definition>
2614 <ccts:PropertyTermName>Currency</ccts:PropertyTermName>
2615 <ccts:RepresentationTermName>Code</ccts:RepresentationTermName>
2616 <ccts:DataTypeTermName>Amount</ccts:DataTypeTermName>
2617 <ccts:SupplementaryComponentValueDomain>
2618 <ccts:DefaultIndicator>True</ccts:DefaultIndicator>
2619 <ccts:SchemeOrListID>42173A</ccts:SchemeOrListID>
2620 <ccts:SchemeOrListVersionID>2009-03-05
2621 </ccts:SchemeOrListVersionID>
2622 <ccts:SchemeOrListAgencyID>5</ccts:SchemeOrListAgencyID>
2623 <ccts:SchemeOrListModificationAllowedIndicator>True
2624 </ccts:SchemeOrListModificationAllowedIndicator>
2625 </ccts:SupplementaryComponentValueDomain>
2626 </xsd:documentation>
2627 </xsd:annotation>
2628 </xsd:attribute>

```

2629 **8.4.6.2 Annotation Application Information (AppInfo)**

2630 The annotation `xsd:appInfo` is expressed for all BDT artefacts defined in BDT
 2631 XML Schema Files. The UsageRules and the context is communicated as defined in
 2632 section [7.5.2, Application Information \(AppInfo\)](#). All UsageRules and contexts in
 2633 which the BDT is applicable is expressed in the `xsd:appInfo`.

2634 **8.5 XML Schema Built-in Type Extension XML Schema File**

2635 In order to support the UN/CEFACT Core Components DT Catalogue Version 3.0,
 2636 additional custom types must be defined to support the ISO 8601 datetime formats
 2637 that are not supported by W3C XML Schema. These custom types are defined in the
 2638 XBT XML Schema File. The XBT XML Schema File is in the data common
 2639 namespace.

[R 8866]	The XML Schema Built-in Type Extension XML Schema File (XBT) MUST be defined in the data common namespace.	1
----------	------------------------------------------------------------------------------------------------------------	---

2640 **8.5.1 XML Schema Structure**

2641 The format is shown in Example 8-34. Each BDT XML Schema File must adhere to
 2642 the format of the relevant sections as detailed in [Appendix B](#).

2643

2644 **Example 8-34: XBT XML Schema file structure**

```

2645 <?xml version="1.0" encoding="utf-8"?>
2646 <!-- ===== -->
2647 <!-- ===== XML Schema Built-in Type Extension XML Schema File ===== -->
2648 <!-- ===== -->
2649 <!--
2650     Schema agency:      UN/CEFACT
2651     Schema version:    3.0
2652     Schema date:       27 January 2009
2653
2654
2655
2656     Copyright (C) UN/CEFACT (2009). All Rights Reserved.
2657
2658     ... see copyright information ...
2659
2660 -->
2661 <xsd:schema targetNamespace=
2662     ... see namespace ...
2663     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
2664     elementFormDefault="qualified" attributeFormDefault="unqualified">
2665 <!-- ===== -->
2666 <!-- ===== Type Definitions ===== -->
2667 <!-- ===== -->
2668     ... see type definitions ...
2669
2670 </xsd:schema>

```

2670 **8.5.2 Type Definitions**

2671 The XBT contains types that are defined using `xsd:simpleType` with regular
 2672 expressions to define the formats for each of the types.

2673 **8.6 Code List XML Schema Files**

2674 Codes are an integral component of any information flow. Codes have been
 2675 developed over time to facilitate the flow of compressed, standardized values that
 2676 can be easily validated for correctness to ensure consistent data. In order for XML
 2677 instance documents to be fully validated by parsers, any codes used within the XML
 2678 document need to be available as part of the schema validation process. Many
 2679 international, national and sectorial agencies create and maintain code lists relevant
 2680 to their area. If required to be used within an information flow, these code lists are
 2681 stored in their own XML Schema File, and are referred to as Common Code Lists.
 2682 For example, many of the code lists that exist in the United Nations Code List
 2683 (UNCL) are stored as Common Code List XML Schema Files for use within other
 2684 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
----------	----------------------------------------------------------------------------------	---

2685 UN/CEFACT recognizes two basic types of code lists:

- 2686 • Common Code List (CCL) – Universally defined for use in all contexts.
 2687 Generally maintained by UN/CEFACT and other standards bodies.
- 2688 • Business Code List (BCL) which are defined within a given context of their
 2689 use. They may be defined as:
 - 2690 ○ a new code list, or
 - 2691 ○ a restriction to an existing CCL, or

- 2692 ○ a combination of existing Code List that is needed within the context of
- 2693 use for a given data package namespace.

2694 Additionally, code lists may exist only within an implementation. When this occurs
 2695 the agency and the code list itself potentially may not have identifiers registered with
 2696 UN/CEFACT or another ID registration organization. In these cases it is
 2697 recommended for organizations to register the agency itself and any code list with
 2698 UN/CEFACT. However, this may not be possible or may not be practical. In these
 2699 cases the agency name in CamelCase format may be used as the Agency Identifier.
 2700 In cases where a Scheme or List Identifier has not been assigned, the Scheme or
 2701 List Name in CamelCase format may be used as the Scheme or List Identifier.

[R 89D1]	Agencies that do not have an Agency Identifier assigned by UN/CEFACT MUST use the Agency Name in CamelCase as the Agency Identifier.	1
[R AD5F]	Agencies that do not have a Scheme or List Identifier assigned MUST use the Scheme or List Name in CamelCase as the SchemeOrList Identifier.	1

2702 **8.6.1 General Code List XML Schema Components**

2703 Both Common Code List XML Schema Files and Business Code List XML Schema
 2704 Files define codes using a consistent approach.

2705 **8.6.1.1 Code List XML Schema File Structure**

2706 Each Code List XML Schema File will be structured in a standard format in order to
 2707 ensure consistency and ease of use. This structure is show in Example 8-35.

2708 **Example 8-35: Code List XML Schema File structure**

```

2709 <?xml version="1.0" encoding="UTF-8"?>
2710 <!-- ===== -->
2711 <!-- ===== 6Recommendation20 - Code List XML Schema File ===== -->
2712 <!-- ===== -->
2713 <!--
2714 Schema agency:      UN/CEFACT
2715 Schema version:    2.0
2716 Schema date:       16 January 2006
2717
2718 Code list name:     Measurement Unit Common Code
2719 Code list agency:   UNECE
2720 Code list version:  3
2721
2722 Copyright (C) UN/CEFACT (2006). All Rights Reserved.
2723
2724 ... see copyright information ...
2725
2726 -->
2727 <xsd:schema targetNamespace=" ... see namespace ...
2728             xmlns:xsd="http://www.w3.org/2001/XMLSchema"
2729             elementFormDefault="qualified" attributeFormDefault="unqualified">
2730 <!-- ===== -->
2731 <!-- ===== Root Element ===== -->
2732 <!-- ===== -->
2733 ... see root element declaration ...
2734 <!-- ===== -->
2735 <!-- ===== Type Definitions ===== -->
2736 <!-- ===== -->
2737 <!-- ===== Type Definition: Measurement Unit Common Code Content Type == -->
    
```

2738
2739
2740

```
<!-- ----- -->
... see type definition ...
</xsd:schema>
```

2741 8.6.1.2 Code List XML Schema Name

2742 The name of Code List XML Schema Files are dependent upon the agency that
2743 defines them and the name of the code list itself.

[R 849E]	<p>Code List XML Schema File names MUST be of the form: <List Agency Identifier>_<List Identifier>_<List 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"> • List 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. • List Identifier – Identifies a list of the respective corresponding ids. • List Version Identifier – Identifies the version. 	2
----------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2744 8.6.1.3 Element Declarations

2745 A Code List XML Schema File contains one global element declaration. This global
2746 element is a unique identifier for the code list and is mandatory for UN/CEFACT
2747 Code List XML Schema Files. Other organizations using this specification may
2748 choose to not provide the Code List Root Element and still be in compliance with this
2749 specification.

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

2750 The global element serves as the root element and is of the one **xsd:simpleType**
2751 that is defined in the Code List XML Schema File.

[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.	1
----------	-------------------------------------------------------------------------------------------------------------------------------------	---

2752 The global element is named using the formal code list name.

[R B5EC]	The Code List XML Schema File global element name MUST be the formal name of the code list with the word Code appended if not present in the code list name.	1
----------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2753 Example 8-36 shows a root element declaration for a code list.

2754

2755 **Example 8-36: Code list global root element declaration**

```

2756 <!-- ===== -->
2757 <!-- ===== Root Element ===== -->
2758 <!-- ===== -->
2759 <xsd:element name="AccountTypeCode" type="clm64437:AccountTypeCodeContentType"/>

```

2760 The actual implementation of the code list is through the use of its
 2761 **xsd:simpleType** by a BDT BVD or BBIE.

2762 **8.6.1.4 Type Definitions**

2763 Each Code List XML Schema File will have one named **xsd:simpleType** defined.

2764 The name of this type will correspond to the code list name with the word
 2765 **ContentType** appended.

[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 with the word code appended if it is not part of the code list name, and with the word ContentType appended.	1

2766 Code List contents are expressed using **xsd:enumeration**, where each value of
 2767 the code list is defined using **xsd:value**.

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

2768 Example 8-37 shows a simple type definition used in a code list.

2769 **Example 8-37: Code list xsd:simpleType definition**

```

2770 <!-- ===== -->
2771 <!-- ===== Type Definitions ===== -->
2772 <!-- ===== -->
2773 <!-- ===== Type Definition: Account Type Code ===== -->
2774 <!-- ===== -->
2775 <xsd:simpleType name="AccountTypeCodeContentType">
2776   <xsd:restriction base="xsd:token">
2777     <xsd:enumeration value="2">
2778       ... see enumeration ...
2779     </xsd:enumeration>
2780   </xsd:restriction>
2781 </xsd:simpleType>

```

2782 **8.6.1.5 Annotation**2783 **8.6.1.5.1 Annotation Documentation**2784 **8.6.1.5.1.1 Code List Documentation**

2785 Every Code List XML Schema file must include structured annotation documentation.

[R A142]	<p>Every Code List MUST contain a structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements that contain:</p> <ul style="list-style-type: none"> • SchemeOrListID (mandatory): The unique identifier assigned to the code list. • SchemeOrListAgencyID (mandatory): The unique identifier assigned to the Agency that owns or is responsible for the code list being referenced. • SchemeOrListVersionID (mandatory): The version of the scheme or list. • SchemeOrListModificationAllowedIndicator (mandatory): Indicates whether the values being validated can be outside the enumerations specified by the code list. 	1
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2786 Example 8-38 shows the declaration of the code list documentation structure.

2787 **Example 8-38: Code list documentation structure**

```

2788 <xsd:group name="SchemeOrListDocumentation">
2789   <xsd:sequence>
2790     <xsd:element name="SchemeOrListID" type="IDType"/>
2791     <xsd:element name="SchemeOrListVersionID" type="IDType" />
2792     <xsd:element name="SchemeOrListAgencyID" type="IDType" />
2793     <xsd:element name="SchemeOrListModificationAllowedIndicator"
2794 type="IndicatorType" />
2795   </xsd:sequence>
2796 </xsd:group>
2797
    
```

2798 **8.6.1.5.1.2 Code List Value Documentation**

2799 In order to facilitate a clear and unambiguous understanding of the list of allowable
 2800 codes within an element, annotation documentation will be provided for each
 2801 enumeration. This documentation will be the name of the value and a description of
 2802 the code.

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

2803 Example 8-39 shows the annotation documentation definition for the enumerations
 2804 values of a code list.

2805

2806 **Example 8-39: Code list enumeration annotation documentation**

```

2807 <xsd:simpleType name="PaymentMethodCodeContentType">
2808   <xsd:restriction base="xsd:token">
2809     <xsd:enumeration value="1"> Name (mandatory): The name of the
2810     code.
2811     Description (optional): Descriptive information concerning the code.
2812
2813     <xsd:annotation>
2814       <xsd:documentation xml:lang="en">
2815         <ccts:Name>Direct payment</ccts:Name>
2816         <ccts:Description>An assigned invoice has
2817         been paid by the buyer to the factor.</ccts:Description>
2818       </xsd:documentation>
2819     </xsd:annotation>
2820   </xsd:enumeration>
2821 </xsd:restriction>
2822 </xsd:simpleType>

```

2823 **8.6.2 Common Code List XML Schema Components**

2824 CCL's are universally defined for all contexts and maintained by standards bodies.
 2825 CCL XML Schema Files will be imported into the context specific namespaces that
 2826 use them.

2827 **8.6.2.1 Namespace Name for Common Code Lists**

2828 The namespace name for a CCL is somewhat unique in order to convey some of the
 2829 Supplementary Components rather than including them as attributes. Specifically,
 2830 the namespace structure for a code list extends the earlier rules for namespace
 2831 names to include the code list name in the namespace.

[R 992A]	Code list XML Schema File namespaces MUST use the following pattern:	3		
	<table border="1"> <tr> <td data-bbox="440 1192 548 1354">URN:</td> <td data-bbox="548 1192 1300 1354">urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:codelist:common:<major>:<status>:<name></td> </tr> <tr> <td data-bbox="440 1354 548 1516">URL:</td> <td data-bbox="548 1354 1300 1516">http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/codelist/common/<major>/<status>/<name></td> </tr> </table>		URN:	urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:codelist:common:<major>:<status>:<name>
URN:	urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:codelist:common:<major>:<status>:<name>			
URL:	http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/codelist/common/<major>/<status>/<name>			
	Where: <ul style="list-style-type: none"> • organization – Identifier of the organization providing the standard. • organization hierarchy – The first level of the hierarchy within the organization providing the standard. • organization hierarchy level – Zero to n level hierarchy of the organization providing the standard. • codelist – A fixed value token for common codelists. 			

	<ul style="list-style-type: none"> • 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. <p style="text-align: center;">Code list names are further defined as:</p> <p style="text-align: center;"><Code List Agency Identifier><divider><Code List Identifier></p> <p style="text-align: center;">Where:</p> <ul style="list-style-type: none"> ▪ Code List Agency Identifier – The identifier for the agency that the code list is from. ▪ Divider – The divider character for URN is ‘:’ the divider character for URL is ‘/’. ▪ Code List Identifier – The identifier for the given code list. 	
--	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--

2832 Example 8-40 shows a namespace name of a code list using an agency and a code
2833 list identifier at draft status.

2834 **Example 8-40: Code list namespace name with an agency and a code list**
2835 **identifier at draft status**

```
2836 "urn:un:unece:unefact:codelist:common:D.04A:draft:6:3403: "
2837 where
2838 D.04A = the version of the UN/CEFACT directory
2839 6 = the value for UN/ECE in UN/CEFACT data element 3055 representing
2840 the Code List. Agency. Identifier
2841 3403 = UN/CEFACT data element tag for Name type code representing
2842 the Code List. Identification. Identifier
```

2843 Example 8-41 shows a namespace name of a code list with and agency and code
2844 list identifier at standard status.

2845 **Example 8-41: Code list namespace name with an agency and a code list**
2846 **identifier at standard status**

```
2847 "urn:un:unece:unefact:codelist:common:D.04A:standard:6:3403"
2848 where
2849 6 = the value for UN/ECE in UN/CEFACT data element 3055 representing
2850 the Code List. Agency. Identifier
2851 3403 = UN/CEFACT data element tag for Name status code representing
2852 the Code List. Identification. Identifier
2853 D.04A = the version of the UN/CEFACT directory
```

2854 While the versioning of code lists published by external organisations is outside of
2855 the control of UN/CEFACT, UN/CEFACT published code lists expressed in XML
2856 Schema Files will follow the rules expressed in this specification.

2857 **8.6.2.2 XML Schema Namespace Token for Common Code Lists**

2858 A unique token will be defined for each namespace for common code lists. The
 2859 token is constructed based on the identifier of the agency maintaining the code list
 2860 and the identifier of the specific code list as issued by the maintenance agency.

2861 The agency maintaining the code list will be identified either by the agency code as
 2862 specified in data element 3055 in the UN/CEFACT Code List directory, or some
 2863 other unique agency identifier if the agency does not have a code value in 3055. The
 2864 identifier of the specific code list will be the data element tag of the corresponding list
 2865 in the UN/CEFACT directory. If there is no corresponding data element, then some
 2866 other unique code list identifier 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 Identifier><Code List Version Identifier></p> <p>Such that any repeated words are eliminated.</p> <p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Identifier – The identifier for the agency that the code list is from. • Code List Identifier – The identifier for the given code list. • Code List Version Identifier – The identifier for the version of the given code list. 	2
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2867 Example 8-42 shows a code list token with an agency and code list identifier.

2868 **Example 8-42: Code list token with an agency and a code list identifier**

```

2869 The code list token for Name Type. Code is clm63403D07B
2870 where
2871 6 = the value for UN/ECE in UN/CEFACT data element 3055 representing
2872 the Code. List Agency. Identifier
2873 3403 = UN/CEFACT data element tag for Name status code representing
2874 the Code. List. Identifier
2875 D07B = UN/CEFACT Code. List Version. Identifier
  
```

2876 Example 8-43 shows a code list token for a business data type with an agency and
 2877 code list identifiers.

2878 **Example 8-43: Code list token for a qualified BDT with an agency and code list
 2879 identifiers**

```

2880 Code list token for Person_Name Type. Code is clmPersonNameType63403D07B
2881 where
2882 PersonNameType_01987A = name of the qualified data type
2883 6 = the value for UN/ECE in UN/CEFACT data element 3055 representing
2884 the Code. List Agency. Identifier
2885 3403 = UN/CEFACT data element tag for Name status code representing
2886 the Code. List. Identifier
2887 D07B = UN/CEFACT Code. List Version. Identifier
  
```

2888 Based on the constructs identified in the above examples, a namespace declaration
 2889 for a code list would appear as shown in Example 8-44.

2890 **Example 8-44: Target namespace declaration for a code list**

```

2891 <xsd:schema
2892   targetNamespace="urn:un:unece:uncefact:codelist:common:D.04A:draft:6:4437"
2893   xmlns:clm64437="urn:un:unece:uncefact:codelist:common:D.04A:draft:6:4437"
2894   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
2895   elementFormDefault="qualified" attributeFormDefault="unqualified">

```

2896 [Note:]

2897 Developers are encouraged to follow the above rules when customizing XML
 2898 Schema for code lists to ensure that there are no namespace conflicts.

2899 **8.6.2.3 Imports and Includes**

2900 UN/CEFACT CCL XML Schema Files are standalone XML Schema Files and will not
 2901 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
----------	-----------------------------------------------------------------------------	---

2902 **8.6.2.4 Type Definitions**

2903 Each CCL XML Schema file will have a single `xsd:simpleType` defined. This type
 2904 definition will have an `xsd:restriction` expression whose base type is the XML
 2905 Schema `xsd:token` built-in data type. The `xsd:restriction` element will be
 2906 used to convey the 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 <code>base</code> attribute is <code>xsd:token</code> and contains one <code>xsd:enumeration</code> element for each value expressed in the code list.	1
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2907 Example 8-45 shows the simple type definition for a code list.

2908 **Example 8-45: CCL `xsd:simpleType` definition**

```

2909 <xsd:simpleType name="PaymentMethodCodeContentType">
2910   <xsd:restriction base="xsd:token">
2911     <xsd:enumeration value="1">
2912       <xsd:annotation>
2913         See annotation
2914       </xsd:annotation>
2915     </xsd:enumeration>
2916   </xsd:restriction>
2917 </xsd:simpleType>...

```

2918 **8.6.2.5 Annotation**2919 **8.6.2.5.1 Annotation Documentation**

2920 CCL XML Schema documentation follows the same structure as defined in section
 2921 [8.5.1.4.1 Annotation Documentation](#) of this specification.

2922

2923 **8.6.2.5.2 Annotation Application Information (AppInfo)**

2924 Common code lists are applicable to all contexts and therefore do not have context
 2925 specified within an `xsd:appInfo` element. Common code lists do not have usage
 2926 rules and therefore do not have usage rules specified within an `xsd:appInfo`
 2927 element.

2928 **8.6.3 Business Code List XML Schema Components**

2929 Business code lists are expressed as Code List XML Schema Files that contain
 2930 codes that are applicable within the data package namespace where it is defined. A
 2931 BCL XML Schema file maybe used where an existing CCL XML Schema File needs
 2932 to be extended, where no suitable CCL XML Schema exists, or where the context in
 2933 which the code list is to be used only needs to make use of a subset of a CCL. This
 2934 is accomplished by:

- 2935 • A combination of several individual code lists using `xsd:union`,
- 2936 • A new code list that is applicable for the context, or
- 2937 • 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"> • Define a codelist where one does not exist or • Restrict the value of an existing code list or • Combine several individual code lists using <code>xsd:union</code>. 	1
----------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2938 **8.6.3.1 Namespace Name for Business Code Lists**

2939 BCLs use the namespace name for the data package in which it is defined. [See](#)
 2940 [Section 5.6 Namespace Scheme](#).

2941 **8.6.3.2 UN/CEFACT XML Schema Namespace Token for Business Code Lists**

2942 BCLs use the namespace token for the data package in which it is defined. [See](#)
 2943 [Section 5.6.2 Namespace Tokens](#). In cases where the BCL is a restricted set of
 2944 values of a published CCL, the BCL will be associated with a BDT, and the name of
 2945 the BDT will be included as part of the namespace token to ensure uniqueness from
 2946 the CCL XML Schema File.

2947 **8.6.3.3 Imports and Includes**

2948 BCL Schema Files may import CCL XML Schema File(s) if the BCL restricts the CCL
 2949 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
----------	------------------------------------------------------------------------------	---

2950 **8.6.3.4 Type Definitions**

2951 Each of the three types of BCLs have different requirements for the XSD types that
 2952 define them.

2953 **8.6.3.4.1 Creating A New BCL Code List**

2954 Each BCL XML Schema File that defines a new code list will have a single
 2955 `xsd:simpleType` defined with an `xsd:restriction` element whose `base`
 2956 attribute is `xsd:token`. The `xsd:restriction` will be used to convey the content
 2957 component enumeration value(s) by using one `xsd:enumeration` element for each
 2958 value expressed in the code list.

[R 8104]	The <code>xsd:simpleType</code> definition for each BCL XML Schema File that defines a new code list MUST use an <code>xsd:restriction</code> element whose <code>base</code> attribute is <code>xsd:token</code> which contains one <code>xsd:enumeration</code> element for each value expressed for the code list.	1
----------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2959 **8.6.3.4.2 Restricting The Value Of An Existing Code List**

2960 Each BCL XML Schema File that restricts the values of an existing code list will have
 2961 a single `xsd:simpleType` defined with an `xsd:restriction` element whose
 2962 `base` attribute is the `xsd:simpleType` of the code list being restricted. The
 2963 `xsd:restriction` will be used to convey the content component enumeration
 2964 value(s) by using an `xsd:enumeration` elements one for each value expressed for
 2965 the restricted code list.

[R 882D]	The <code>xsd:simpleType</code> definition for each BCL XML Schema File that restricts an existing code list MUST use an <code>xsd:restriction</code> element whose <code>base</code> attribute is the <code>xsd:simpleType</code> of the code list being restricted which contains one <code>xsd:enumeration</code> element for each value expressed in the restricted code list.	1
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2966 **8.6.3.4.3 Combining Multiple Code Lists**

2967 Each BCL XML Schema File that combines the values of multiple Code Lists will
 2968 have a single `xsd:simpleType` defined with an `xsd:union` element whose
 2969 `memberTypes` attribute contain the `xsd:simpleTypes` of the code lists being
 2970 unioned together.

[R 9A22]	The <code>xsd:simpleType</code> definition for each BCL XML Schema File that combines the values of multiple code lists MUST use an <code>xsd:union</code> element whose <code>memberTypes</code> attribute contains the <code>xsd:simpleTypes</code> of the code lists being unioned together.	1
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

2971 [Note:] Sequence of Code Lists

2972 As defined in XML Schema, the sequence of code lists in an `xsd:memberType`
 2973 attribute is significant. Schema authors should take this into consideration in defining
 2974 the type.

2975 Example 8-46 shows an example of using two code lists in a BDT.

2976 **Example 8-46: Combination of Two Code Lists**

```

2977 <xsd:simpleType name="AccountDutyCodeContentType">
2978   <xsd:annotation>
2979     ... see annotation ...
2980   </xsd:annotation>
2981   <xsd:union memberType="clm64437:AccountTypeCodeContentType
2982     clm65153:DutyTaxFeeTypeCodeContentType" />
2983 </xsd:simpleType>
  
```

2984 **8.6.3.5 Annotation**

2985 **8.6.3.5.1 Annotation Documentation**

2986 BCL XML Schema documentation is the same as CCL XML Schema documentation
 2987 described in Section [8.6.1.5.1 Annotation Documentation](#).

2988 **8.6.3.5.2 Annotation Application Information (AppInfo)**

2989 BCL usage rules and context information is as defined in section [7.5.2, Application](#)
 2990 [Information \(AppInfo\)](#).

2991 **8.7 Identifier Scheme XML Schema Files**

2992 Identifiers are an integral component of managing business objects. Identifiers have
 2993 been developed over time to provide for uniquely identifying one object from another.
 2994 When identifiers are part of an XML based business information exchange, any
 2995 identifiers used within the XML document need to be able to be validated by the XML
 2996 parser as to the identifiers adherence to the scheme that defines it.

2997 Many international, national and sectorial agencies create and maintain identifier
 2998 schemes. If required to be used within an information flow, these schemes will be
 2999 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
----------	------------------------------------------------------------------------------------------	---

3000 UN/CEFACT recognizes two basic types of identifier schemes:

- 3001 • Common Identifier Scheme (CIS) – Universally defined for use in all contexts.
 3002 Generally maintained by UN/CEFACT or other standards bodies.
- 3003 • Business Identifier Scheme (BIS) – These are identifiers that are defined
 3004 within a given context of their use. The may be defined as:
 - 3005 ○ A restriction on the pattern or allowed values of an existing CIS, or
 - 3006 ○ An extension on the pattern or allowed values of an existing CIS, or
 - 3007 ○ A new CIS that is needed within the context of use for a given context
 3008 category namespace.

3009 **8.7.1 General Identifier Scheme XML Schema Components**

3010 Both Common Identifier Scheme XML Schema Files and Business Identifier Scheme
3011 XML Schema Files define the schemes using a consistent approach.

3012 **8.7.1.1 Identifier Scheme XML Schema File Structure**

3013 Each Identifier Scheme XML Schema File will be structured in a standard format in
3014 order to ensure consistency and ease of use. This structure is show in Example 8-
3015 47.

3016 **Example 8-47: Identifier scheme XML Schema File structure**

```

3017 <?xml version="1.0" encoding="UTF-8"?>
3018 <!-- ===== -->
3019 <!-- ===== Global Trade Identification Number - Identifier Scheme XML Schema
3020 File===== -->
3021 <!-- ===== -->
3022 <!--
3023 Schema agency:          GS1
3024 Schema version:        1.0
3025 Schema date:           21 December 2008
3026
3027 Identifier Scheme name: Global Trade Identification Number
3028 Identification Scheme agency:      GS1
3029 Identification Scheme version:     1
3030
3031 Copyright (C) UN/CEFACT (2009). All Rights Reserved.
3032
3033 ... see copyright information ...
3034
3035 -->
3036 <xsd:schema targetNamespace=" ... see namespace ...
3037           xmlns:xsd="http://www.w3.org/2001/XMLSchema"
3038           elementFormDefault="qualified" attributeFormDefault="unqualified">
3039 <!-- ===== -->
3040 <!-- ===== Root Element ===== -->
3041 <!-- ===== -->
3042           ... see root element declaration ...
3043 <!-- ===== -->
3044 <!-- ===== Type Definitions ===== -->
3045 <!-- ===== -->
3046 <!--= Type Definition: Global Trade Identification Number Content Type =-->
3047 <!-- ===== -->
3048           ... see type definition ...
3049 </xsd:schema>

```

3050 **8.7.1.2 Identifier Scheme XML Schema Name**

3051 The name of Identifier Scheme XML Schema Files are dependent upon the agency
3052 that defines them and the identifier scheme itself.

[R A50B]	<p>Identifier Scheme XML Schema File names MUST be of the form: <Scheme Agency Identifier>_<Scheme Identifier>_<Scheme 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"> • Scheme Agency Identifier – Identifies the agency that manages the identifier scheme. The default agency IDs used are those from UN/EDIFACT DE 3055, however, roles defined in DE 3055 cannot be used. • Scheme Identifier – Identifies the identifier scheme. • Scheme Version Identifier – Identifies the version of the scheme. 	2
----------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

3053 **8.7.1.3 Element Declarations**

3054 An Identifier Scheme XML Schema File contains one global element declaration.
 3055 This global element is a unique identifier for the identifier scheme and is mandatory
 3056 for UN/CEFACT Identifier Scheme XML Schema Files. Other organizations using
 3057 this specification may choose to not provide the Identifier Scheme Root Element and
 3058 still be in compliance with this specification.

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

3059 The global element serves as the root element and is of the one xsd:simpleType
 3060 that is defined in the Identifier Scheme XML Schema File.

[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.	1
----------	---------------------------------------------------------------------------------------------------------------------------------------------------	---

3061 The global element is named using the formal identifier scheme name.

[R 9B48]	The Identifier Scheme XML Schema File global element name MUST be the formal name of the Identifier Scheme with the word identifier appended if not present in the Identifier Scheme name	1
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

3062 Example 8-48 shows a root element declaration for an identifier scheme.

3063 **Example 8-48: Identifier scheme root element declaration**

```

3064 <!-- ===== -->
3065 <!-- ===== Root Element ===== -->
3066 <!-- ===== -->
3067 <xsd:element name="GlobalTradeIdentificationNumber"
3068 type="ism8GTIN:GlobalTradeIdentificationNumberType"/>
    
```


3069 The actual implementation of the identifier scheme is through the use of its
3070 **xsd:simpleType** by a BDT BVD or BBIE.

3071 8.7.1.4 Type Definitions

3072 Each Identifier XML Schema File will have one named **xsd:simpleType** defined.
3073 The name of this type will correspond to the identifier scheme name with the word
3074 'Content**Type**' appended.

[R 9451]	Each Identifier Scheme XML Schema File MUST define one, and only one, named xsd:simpleType for the content component.	1
[R B79A]	The Identifier Scheme XML Schema File xsd:simpleType name MUST be the name of the identifier scheme with the word Identifier appended if not part of the identifier scheme name and the word ContentType appended.	1

3075 The identifiers created by an identifier scheme are never enumerated.

3076 Example 8-49 shows the definition of a Global Trade Identification Number
3077 **xsd:simpleType**.

3078 **Example 8-49: Identifier scheme **xsd:simpleType** name**

```

3079 <!-- ===== -->
3080 <!-- ===== Root Element ===== -->
3081 <!-- ===== -->
3082 <xsd:element name="GlobalTradeIdentificationNumber"
3083 type="ism8GTIN:GlobalTradeIdentificationNumberType"/>
3084 <!-- ===== -->
3085 <!-- ===== Type Definitions ===== -->
3086 <!-- ===== -->
3087 <!-- == Type Definition: Global Trade Identification Number Identifier= -->
3088 <!-- ===== -->
3089 <xsd:simpleType name="GlobalTradeIdentificationNumberContentType">
3090 See type definition
3091 </xsd:simpleType>

```

3092 8.7.1.5 Annotation

3093 8.7.1.5.1 Annotation Documentation

3094 8.7.1.5.1.1 Identifier Scheme Documentation

3095 Every Identifier Scheme XML Schema file must include structured annotation
3096 documentation.

<p>[R B30A]</p>	<p>Every Identifier Scheme MUST contain a structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements in the following sequence and pattern:</p> <ul style="list-style-type: none"> • SchemeOrListID (mandatory): The unique identifier assigned to the Identifier Scheme. • SchemeOrListVersionID (mandatory): Identifies the version of the scheme. • SchemeOrListAgencyID (mandatory): The unique identifier assigned to the Agency that owns or is responsible for the identifier scheme being referenced. <p>SchemeOrListModificationAllowedIndicator (mandatory): Indicates whether the values being validated can be outside the pattern specified by the scheme.</p>	<p>1</p>
-----------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------

3097 Example 8-50 shows the declaration of the annotation documentation for each
 3098 Identifier Scheme.

3099 **Example 8-50: Identifier scheme documentation structure**

3100
 3101
 3102
 3103
 3104
 3105
 3106
 3107
 3108

```

<xsd:group name="SchemeOrListDocumentation">
  <xsd:sequence>
    <xsd:element name="SchemeOrListID" type="IDType"/>
    <xsd:element name="SchemeOrListVersionID" type="IDType"/>
    <xsd:element name="SchemeOrListAgencyID" type="IDType" />
    <xsd:element name="SchemeOrListModificationAllowedIndicator"
type="IndicatorType"/>
  </xsd:sequence>
</xsd:group>
    
```

3109 **8.7.2 Common Identifier Scheme XML Schema Components**

3110 CIS are universally defined for all contexts and maintained by standards bodies. CIS
 3111 XML Schema Files will be imported into the context specific namespaces that use
 3112 them.

3113 **8.7.2.1 Namespace Name for Common Identifier Scheme**

3114 The namespace name for a CIS is somewhat unique in order to convey some of the
 3115 Supplementary Components rather than including them as attributes. Specifically,
 3116 the namespace structure for an identifier scheme extends the earlier rules for
 3117 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="440 279 1300 611"> <tr> <td data-bbox="440 279 553 447">URN:</td> <td data-bbox="553 279 1300 447">urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:identifierscheme:common:<major>:<status>:<name></td> </tr> <tr> <td data-bbox="440 447 553 611">URL:</td> <td data-bbox="553 447 1300 611">http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/identifierscheme/common/<major>/<status>/<name></td> </tr> </table>	URN:	urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:identifierscheme:common:<major>:<status>:<name>	URL:	http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/identifierscheme/common/<major>/<status>/<name>	3
	URN:	urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:identifierscheme:common:<major>:<status>:<name>				
URL:	http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/identifierscheme/common/<major>/<status>/<name>					
<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. • 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 <i>XML Schema File</i> removed. <ul style="list-style-type: none"> ○ Identifier scheme names are further defined as: <Identifier Scheme Agency Identifier> <divider><Identifier Scheme Identifier> <p>Where:</p> <ul style="list-style-type: none"> ▪ Identifier Scheme Agency Identifier – The identifier for the agency that identifier scheme is from. ▪ Divider – The divider character for URN is : the divider character for URL is /. ▪ Identifier Scheme Identifier – The identifier for the given identifier scheme. 						

3118 Example 8-51 shows an identifier scheme namespace where the status of the
 3119 identifier scheme is in draft status.

3120 **Example 8-51: Identifier scheme namespace name with an agency and a**
 3121 **identifier scheme identifier at draft status**

```

3122 "urn:un:unece:unefact:identifierscheme:common:D.04A:draft:8:GTIN: "
3123 where
3124 D.04A = the version of the UN/CEFACT directory
3125 8 = the value for GS1 in UN/CEFACT data element 3055 representing
3126 the Identifier. Scheme Agency. Identifier
3127 GTIN = GS1 data element tag for Global Trade Identification Number representing
3128 the Identifier. Scheme. Identifier
    
```

3129 While the versioning of identifier schemes published by external organisations is
 3130 outside of the control of UN/CEFACT, UN/CEFACT published code lists expressed
 3131 in XML Schema Files will follow the rules expressed in this specification.

3132 **8.7.2.2 XML Schema Namespace Token for Common Identifier Schemes**

3133 A unique token will be defined for each namespace for common identifier schemes.
 3134 The token is constructed based on the identifier of the agency maintaining the
 3135 identifier scheme and the identifier of the specific identifier scheme as issued by the
 3136 maintenance agency.

3137 The agency maintaining the identifier scheme will be identified by the agency code
 3138 as specified in data element 3055 in the UN/CEFACT Code List directory or some
 3139 other unique identifier for the agency. The identifier of the specific identifier scheme
 3140 will be the data element tag of the corresponding list in the UN/CEFACT directory, or
 3141 some other unique identifier for the scheme.

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

3142 [Note:]
 3143 Developers are encouraged to follow the above rules when customizing XML
 3144 Schema for Identifier Schemes to ensure that there are no namespace conflicts.

3145 Example 8-52 shows an identifier scheme token.

3146

3147 **Example 8-52: Identifier scheme token with an agency and an identifier**
 3148 **scheme identifier**

3149
3150
3151
3152
3153
3154
3155
3156

```
The identifier scheme token for Global Trade Identification Number Identifier is
ism8gtin200912
where
8 = the value for GS1 in UN/CEFACT data element 3055 representing
the Identifier Scheme. Agency. Identifier
gtin = GS1 data element tag for Global Trade Identification Number representing
the Identifier Scheme. Identification. Identifier
=200912 = the version
```

3157 **8.7.2.3 Imports and Includes**

3158 UN/CEFACT CIS XML Schema Files are standalone XML Schema Files and will not
 3159 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
----------	-----------------------------------------------------------------------------	---

3160 **8.7.2.4 Type Definitions**

3161 Each CIS XML Schema file will have a single `xsd:simpleType` defined. This type
 3162 definition will have an `xsd:restriction` element whose base type is the
 3163 `xsd:token` XML Schema built-in data type.

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

3164 Example 8-53 shows an CIS simpleType definition.

3165 **Example 8-53: CIS `xsd:simpleType` definition**

3166
3167
3168

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

3169 A CIS XML Schema File is only identifying the metadata about the identifier scheme,
 3170 it is not defining the actual scheme itself since that information is publicly available.

3171 **8.7.2.5 Annotation**

3172 **8.7.2.5.1 Annotation Documentation**

3173 CIS XML Schema documentation follows the same structure as defined in section
 3174 [8.6.1.5.1 Annotation Documentation](#) of this specification.

3175 **8.7.2.5.2 Annotation Application Information (AppInfo)**

3176 Common identifier schemes are applicable to all context and therefore do not have
 3177 context specified within `xsd:appInfo`. Common identifier schemes are devoid of
 3178 business rules and therefor do not have business rules specified within
 3179 `xsd:appInfo`.

3180 **8.7.3 Business Identifier Scheme XML Schema Components**

3181 Business identifier schemes are Identifier Scheme XML Schema Files that define a
 3182 scheme that is applicable within a context category namespace. A BIS XML Schema
 3183 file may be used where an existing CIS XML Schema identifier scheme needs to be
 3184 modified, or where no suitable CIS XML Schema exists. In all cases this is
 3185 accomplished by creating a new identifier scheme. The BIS will:

- 3186 ○ Define a new CIS that is needed within the context of use for a given
 3187 context category namespace.
- 3188 ○ Redefine an existing CIS by defining:
 - 3189 ▪ A restriction on the pattern or allowed values of an existing CIS.
 - 3190 ▪ An extension on the pattern or allowed values of an existing
 3191 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
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

3192 **8.7.3.1 Namespace Name for Business Information Scheme**

3193 A BIS uses the namespace name for the data package in which it is defined. This is
 3194 described earlier in this specification in section [5.6 Namespace Scheme](#).

3195 **8.7.3.2 UN/CEFACT XML Schema Namespace Token for Business Information 3196 Scheme**

3197 A BIS uses the namespace token for the data package in which it is defined. This is
 3198 described earlier in this specification in section [5.6.2 Namespace Tokens](#).

3199 **8.7.3.3 Imports and Includes**

3200 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
----------	--------------------------------------------------------------------------------------------	---

3201 **8.7.3.4 Type Definitions**

3202 Each BIS XML Schema file will have a single `xsd:simpleType` defined. This type
 3203 definition will have a `xsd:restriction` expression whose base type is an XML
 3204 Schema built-in data type of `xsd:token`. The `xsd:restriction xsd:token`
 3205 facets may be used to define the actual identifier scheme as part of the type
 3206 definition.

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

3207 Example 8-54 shows a BIS `simpleType` definition.

3208 **Example 8-54: BIS `xsd:simpleType` definition**

```

3209 <xsd:simpleType name="SupplyWarehouseIdentificationNumberContentType">
3210 <xsd:restriction base="xsd:token">
3211 </xsd:simpleType>

```

3212 **8.7.3.5 Annotation**

3213 [8.7.3.5.1 Annotation Documentation](#)

3214 BIS XML Schema documentation is the same as CIS XML Schema documentation
 3215 described in section [8.6.1.5.1.1 Annotation Documentation](#).

3216 [8.7.3.5.2 Annotation Application Information \(AppInfo\)](#)

3217 BIS usage rules and context information is as defined in section [7.5.2, Application
 3218 \[Information \\(AppInfo\\)\]\(#\)](#).

3219 9 XML Instance Documents

3220 In order to be UN/CEFACT conformant, an instance document must be valid against
 3221 the relevant UN/CEFACT compliant XML Schema file(s). XML instance documents
 3222 should be readable and understandable by both humans and applications, and
 3223 should enable reasonably intuitive interactions. An XPath navigation path should
 3224 describe the complete semantic understanding by concatenating the nested
 3225 elements. This navigation path should also reflect the meaning of each dictionary
 3226 entry name of a ABIE, BBIE or ASBIE.

3227 This section further describes the requirements XML Instance documents:

- 3228 • Character Encoding
- 3229 • xsi:schemaLocation
- 3230 • Empty Content
- 3231 • xsi:type

3232 9.1 Character Encoding

3233 In conformance with ISO/IETF/ITU/UNCEFACT Memorandum of Understanding
 3234 Management Group (MOUMG) Resolution 01/08 (MOU/MG01n83) as agreed to by
 3235 UN/CEFACT, all UN/CEFACT XML will be instantiated using UTF. UTF-8 is the
 3236 preferred encoding, but UTF-16 may be used where necessary to support other
 3237 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
----------	---------------------------------------------------------------------------------------------------------	---

3238 9.2 xsi:schemaLocation

3239 The `xsi:schemaLocation` and `xsi:noNamespaceLocation` attributes are part
 3240 of the XML schema instance namespace (<http://www.w3.org/2001/XMLSchema-instance>). To ensure consistency, the token `xsi` will be used to represent the XML
 3241 schema instance namespace.
 3242

[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
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

3243 9.3 Empty Content

3244 Empty elements do not provide the level of assurance necessary for business
 3245 information exchanges and as such, will not be used.

3246 The only case in which elements maybe empty are in cases of where the key and
 3247 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.	3
----------	--------------------------------------------------------------------------------	---

3248

3249

[R B4D1]	If used by other than UN/CEFACT organizations, the <code>xsi:nil</code> attribute MUST only be used to signal the intentional removal of a previously communicated value.	1
----------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

3250

9.4 `xsi:type`

3251 The `xsi:type` attribute allows for substitution during an instantiation of a xml
 3252 document. In the same way that substitution groups are not allowed, the `xsi:type`
 3253 attribute is not allowed.

[R 8250]	The <code>xsi:type</code> attribute MUST NOT be used within an XML Instance.	1
----------	------------------------------------------------------------------------------	---

3254

9.5 Supplementary Components

3255 Code lists and identifier schemes can be defined for a business value domain either
 3256 at model design time or at instance run time. When the code list or identifier scheme
 3257 is defined at model design time, it is included as part of the BDT definition in the BDT
 3258 XML Schema File. If a code list or identifier scheme is defined at instance run time,
 3259 the supplementary component attributes are used to identify the list or scheme. To
 3260 maximize interoperability and minimize human intervention required at runtime, the
 3261 preferred approach is to define the scheme or list at model design time. Only in very
 3262 rare circumstances should the supplementary component attributes for identifying a
 3263 scheme or list be used.

[R A884]	The attributes for scheme or list supplementary components SHOULD NOT be used within an XML Instance.	1
----------	-------------------------------------------------------------------------------------------------------	---

3264 **Appendix A. Related Documents**

3265 The following documents provided significant levels of influence in the development
3266 of this document:

- 3267 • UN/CEFACT Core Components Technical Specification Version 3.0 of 29
3268 September 2009
- 3269 • UN/CEFACT Core Components Data Type Catalogue Version 3.0 of 29
3270 September 2009

3271 **Appendix B. Overall Structure**

3272 The structure of each UN/CEFACT NDR 3.0 compliant XML Schema File must
3273 contain one or more of the following sections as relevant. Relevant sections must
3274 appear in the order given:

- 3275 • XML Declaration
- 3276 • Schema Module Identification and Copyright Information
- 3277 • Schema Start-Tag
- 3278 • Includes
- 3279 • Imports
- 3280 • Element Declarations
 - 3281 ○ Root Element Declarations
 - 3282 ○ Global Element Declarations
- 3283 • Type Definitions

3284 **B.1 XML Declaration**

3285 A UTF-8 encoding is adopted throughout all UN/CEFACT XML Schema Files.

3286 **Example B-1: XML Declaration**

3287

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

3288 See [Section 9.1](#) for exceptions.

3289 **B.2 Schema Module Identification and Copyright Information**

3290 Each UN/CEFACT NDR 3.0 compliant XML Schema File must contain a header
3291 structured as shown in Example B-2, which contains XML Schema File Identification
3292 and Copyright Information.

3293

3294 **Example B-2: XML Schema File Identification and Copyright Information**

```

3295 <!-- ===== -->
3296 <!-- ===== Example - Schema Module Name ===== -->
3297 <!-- ===== -->
3298 <!--
3299 Schema agency: UN/CEFACT
3300 Schema version: 3.0
3301 Schema date: 16 November 2009
3302
3303
3304 Copyright (C) UN/CEFACT (2009). All Rights Reserved.
3305
3306 This document and translations of it may be copied and furnished to others, and
3307 derivative works that comment on or otherwise explain it or assist in its
3308 implementation may be prepared, copied, published and distributed, in whole or in
3309 part, without restriction of any kind, provided that the above copyright notice and
3310 this paragraph are included on all such copies and derivative works. However, this
3311 document itself may not be modified in any way, such as by removing the copyright
3312 notice or references to UN/CEFACT, except as needed for the purpose of developing
3313 UN/CEFACT specifications, in which case the procedures for copyrights defined in
3314 the UN/CEFACT Intellectual Property Rights document must be followed, or as
3315 required to translate it into languages other than English.
3316
3317 The limited permissions granted above are perpetual and will not be revoked by
3318 UN/CEFACT or its successors or assigns.
3319
3320 This document and the information contained herein is provided on an "AS IS" basis
3321 and UN/CEFACT DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT
3322 LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE
3323 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR
3324 PURPOSE.
3325 -->

```

3326 **B.3 Schema Start-Tag**

3327 The Schema Start-Tag section of UN/CEFACT compliant XML Schema must contain
 3328 one or more of the below declarations as relevant. Relevant declarations must
 3329 appear in the order given:

- 3330 • Namespaces
 - 3331 ○ targetNamespace attribute
 - 3332 ○ **xmlns:xsd** attribute
 - 3333 ○ namespace declaration for current schema
 - 3334 ○ namespace declaration for common CCTS XML Builtin Types used
3335 in the schema
 - 3336 ○ namespace declaration for common code lists actually used in the
3337 schema
 - 3338 ○ namespace declaration for common identifier schemes actually
3339 used in the schema
 - 3340 ○ namespace declaration for CCTS documentation
- 3341 • Form Defaults
 - 3342 ○ **elementFormDefault**
 - 3343 ○ **attributeFormDefault**
- 3344 • Version

- 3345 • Others
- 3346 ○ other schema attributes with schema namespace
- 3347 ○ other schema attributes with non-schema namespace

3348 **Example B-3: XML Schema Start Tag**

```

3349 <xsd:schema
3350 targetNamespace="urn:un:unece:uncefact:data:ordermanagement:1:draft"
3351 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
3352 xmlns="urn:un:unece:uncefact:data:ordermanagement:1:draft"
3353 xmlns:xbt="urn:un:unece:uncefact:data:common:1:draft"
3354 xmlns:clm6Recommendation20="urn:un:unece:uncefact:codelist:common:6:standard:6:Reco
3355 mmentation20:6"
3356 xmlns:clm60133="urn:un:unece:uncefact:codelist:common:1:standard:6:0133:40106"
3357 xmlns:clm5ISO6392A="urn:un:unece:uncefact:codelist:common:2009-06-
3358 02:standard:5:iso6392A:2009-06-02"
3359 xmlns:clm5ISO42173A="urn:un:unece:uncefact:codelist:common:2009-03-
3360 05:standard:5:ISO42173A:2009-03-05"
3361 xmlns:ids5ISO316612A="urn:un:unece:uncefact:identifierlist:common:
3362 SecondEdition2006VI-4:standard:5:ISO316612A:SecondEdition2006VI-4"
3363 xmlns:clmIANAMIMEMediaType="urn:un:unece:uncefact:codelist:common: 2009-03-
3364 04:standard:IANA:MIMEMediaType:2009-03-04"
3365 xmlns:clmIANANCharacterSetCode="urn:un:unece:uncefact:codelist:common: 2007-05-
3366 14:standard:IANA:CharacterSetCode:2007-05-14"
3367 xmlns:clm63055="urn:un:unece:uncefact:codelist:common:D08B:standard:6:3055:D08B"
3368 xmlns:ccts="urn:un:unece:uncefact:documentation:common:3:standard:CoreComponentsTec
3369 hnicalSpecification:3"
3370 elementFormDefault="qualified"
3371 attributeFormDefault="unqualified"
3372 version="1.0">

```

3373 **B.4 Includes**

3374 The Include section of an UN/CEFACT compliant XML schema must contain one or
3375 more of the below declarations as relevant. Relevant declarations must appear in the
3376 order given:

- 3377 • Inclusion of the package specific BIE XML Schema File.
- 3378 • Inclusion of the package specific BDT XML Schema File.
- 3379 • Inclusion of the package specific Business Code List XML Schema Files, if
3380 used.
- 3381 • Inclusion of the package specific Business Identifier Scheme XML Schema
3382 Files, if used.

3383 All schemaLocations are relative from the XML Schema File that is making the
3384 reference. For the purposes of this appendix we are assuming the references are
3385 from a Root Schema File within the same namespace as the includes.

3386

3387 **Example B-4: Includes**

```

3388 <!-- ===== Includes ===== -->
3389 <!-- ===== Inclusion of context category BIE XML Schema File ===== -->
3390 <!-- ===== Inclusion of context category BDT XML Schema File ===== -->
3391 <xsd:include schemaLocation="BusinessInformationEntity_3p0.xsd"/>
3392 <!-- ===== Inclusion of context category BCL XML Schema File ===== -->
3393 <xsd:include schemaLocation="BusinessDataType_3p0.xsd"/>
3394 <!-- Inclusion of context specific BCL XML Schema File = -->
3395 <!-- ===== Inclusion of context category BIS XML Schema File ===== -->
3396 <xsd:include schemaLocation="BusinessCodeList_1p0.xsd"/>
3397 <!-- Inclusion of context specific BIS XML Schema File = -->
3398 <!-- ===== Inclusion of context category BIE XML Schema File ===== -->
3400 <xsd:include schemaLocation="BusinessIdentifierScheme_1p0.xsd"/>
3401
3402
3403
3404
3405

```

3406 **B.5 Imports**

3407 The Import section of an UN/CEFACT compliant XML Schema File must contain one
3408 or more of the below declarations as relevant. Relevant declarations must appear in
3409 the order given:

- 3410 • Import of Data Common XML Built-in Types XML Schema File
- 3411 • Import of all Common Code List XML Schema Files actually used
- 3412 • Import of all Common Identifier Scheme XML Schema Files actually used
- 3413 • Import of all other Root XML Schema Files that the importing Root XML
3414 Schema File uses BIEs
- 3415 • Import of all other BIE XML Schema Files that the importing BIE XML Schema
3416 File uses BIEs

3417 **Example B-5: Imports**

```

3418 <!-- ===== Imports ===== -->
3419 <!-- ===== Import of Code lists ===== -->
3420 <xsd:import namespace="urn:un:unece:unefact:data:common:1:draft"
3421 schemaLocation="../../../data/common/1/draft/XMLBuilt-InType.xsd"/>
3422 <!-- ===== Import of Identifier Schemes ===== -->
3423 <xsd:import namespace="urn:un:unece:unefact:codelist:common:2001:standard:5:4217"
3424 schemaLocation="../../../codelist/common/2001/standard/ISO_CurrencyCode_2001.xsd"
3425 />
3426 ...
3427 <!-- ===== Import of Identifier Schemes ===== -->
3428 <xsd:import
3429 namespace="urn:un:unece:unefact:identifierlist:standard:5:ISO6393A:2008-11-07"
3430 schemaLocation="../../../identifierlist/standard/ISO_ISOCodesForTheRepresentatio
3431 nOfNamesOfLanguages_2008-11-07.xsd"/>
3432 ...
3433 <!-- ===== Import of Other Root Schemas ===== -->
3434 <xsd:import namespace="urn:un:unece:unefact:data:common:1:standard"
3435 schemaLocation="../../../common/1/standard/CommonRoot.xsd"/>
3436 ...
3437 <!-- ===== Import of Other Root Schemas ===== -->
3438
3439
3440
3441
3442
3443
3444

```

```

3445 <!-- ===== Import of Other BIE Schema ===== -->
3446 <!-- ===== -->
3447 <xsd:import namespace="urn:un:unece:uncefact:data:common:l:standard"
3448 schemaLocation="../../../common/l/standard/BIE.xsd"/>
3449 ...

```

3450 B.6 Elements

3451 The root element is declared first when needed in an XML Schema File that are used
 3452 to support XML instance documents. Global elements are then declared following
 3453 the root element as required.

3454 Example B-6:

```

3455 <!-- ===== -->
3456 <!-- ===== Element Declarations ===== -->
3457 <!-- ===== -->
3458 <!-- ===== Root element ===== -->
3459 <!-- ===== -->
3460 <xsd:element name="[ELEMENTNAME]" type="[TOKEN]:[TYPENAME]">
3461 <!-- ===== -->
3462 <!-- ===== Global Element Declarations ===== -->
3463 <!-- ===== -->
3464 <xsd:element name="[ELEMENTNAME]" type="[TOKEN]:[TYPENAME]">
3465 <!-- ===== -->

```

3466 B.7 Root element

3467 The root element's type definition is defined immediately following the definition of
 3468 the global root element to provide clear visibility of the root element's type, of which
 3469 this particular schema is defined.

3470 Example B-7:

```

3471 <!-- ===== -->
3472 <!-- ===== Root element ===== -->
3473 <!-- ===== -->
3474 <xsd:element name="Invoice" type="rsm:InvoiceType">
3475 <xsd:annotation>
3476 <xsd:documentation>
3477 <ccts:UniqueID>UNM0000001</ccts:UniqueID>
3478 <ccts:VersionID>3.0</ccts:VersionID>
3479 <ccts:DictionaryEntryName>Invoice</ccts:DictionaryEntryName>
3480 <ccts:Definition>Document used to communicate the Invoice for a
3481 Purchase.</ccts:Definition>
3482 <ccts:ObjectClassTermName>Invoice</ccts:ObjectClassTermName>
3483 </xsd:documentation>
3484 </xsd:annotation>
3485 </xsd:element>

```

3486 Example B-8: Global elements

```

3487 <!-- ===== -->
3488 <!-- ===== Global element ===== -->
3489 <!-- ===== -->
3490 <xsd:element name="BuyerParty" type="bie:BuyerPartyType"/>
3491 <xsd:annotation>
3492 <xsd:documentation>
3493 <ccts:UniqueID>UNM0000002</ccts:UniqueID>
3494 <ccts:VersionID>3.0</ccts:VersionID>
3495 <ccts:DictionaryEntryName>Buyer. Party</ccts:DictionaryEntryName>
3496 <ccts:Definition>The Party that initiated the a
3497 Purchase.</ccts:Definition>
3498 <ccts:ObjectClassQualifierName>Buyer
3499 </ccts:ObjectClassQualifierName>

```

3500
3501
3502
3503

```

    <ccts:ObjectClassTermName>Party</ccts:ObjectClassTermName>
  </xsd:documentation>
</xsd:annotation>
</xsd:element>

```

3504 **B.8 Type Definitions**

3505 The definition of the BIEs.

- 3506
- 3507 • Definition of types for Basic Business Information Entities in alphabetical order, if applicable.
 - 3508 • Definition of types for Aggregate Business Information Entities in alphabetical order, if applicable.
- 3509

3510 **Example B-9: Type Definitions**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
3547
3548
3549
3550
3551
3552
3553
3554
3555
3556
3557
3558
3559
3560
3561
3562
3563
3564

```

<!-- =====>
<!-- ===== Type Definitions =====>
<!-- =====>
<!-- ===== Type Definition: Account type =====>
<!-- =====>
  <xsd:complexType name="AccountType">
    <xsd:annotation>
      <xsd:documentation xml:lang="en">
        <ccts:UniqueID>UN00000001</ccts:UniqueID>
        <ccts:VersionID>3.0</ccts:VersionID>
        <ccts:DictionaryEntryName>Account.
Details</ccts:DictionaryEntryName>
        <ccts:Definition>A business arrangement whereby debits
and/or credits arising from transactions are recorded. This could be with a bank,
i.e. a financial account, or a trading partner offering supplies or services 'on
account', i.e. a commercial account</ccts:Definition>
        <ccts: ObjectClassTermName >Account</ccts:
ObjectClassTermName >
        </xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
        <xsd:element name="ID" type="IDType" minOccurs="0"
maxOccurs="unbounded">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">
              <ccts:DictionaryEntryName>Account.
Identifier</ccts:DictionaryEntryName>
              <ccts:Definition>The identification of a
specific account.</ccts:Definition>
              <ccts:Cardinality>0..n</ccts:Cardinality>
              <ccts:SequencingKey>1</ccts:SequencingKey>
              <ccts:ObjectClassTermName>Account
</ccts:ObjectClassTermName>
              <ccts:PropertyTermName>Identifier
</ccts:PropertyTermName>
              <ccts:RepresentationTermName>Identifier
</ccts:RepresentationTermName>
              <ccts:BusinessTermName>Account Number
</ccts:BusinessTermName>
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="Status" type="StatusType" minOccurs="0"
maxOccurs="unbounded">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">
              <ccts:UniqueID>UN00000003</ccts:UniqueID>
              <ccts:Version>3.0</ccts:Version>
              <ccts:DictionaryEntryName>Account. Status
to account details.</ccts:Definition>
              <ccts:Cardinality>0..n</ccts:Cardinality>
              <ccts:SequencingKey>2</ccts:SequencingKey>
            </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>

```


33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

</ccts:ObjectClassTermName> <ccts:ObjectClassTermName>Account
</ccts:PropertyTermName> <ccts:PropertyTermName>Status
</ccts:AssociationType> <ccts:AssociationType>Composite
</ccts:AssociatedObjectClassTermName> <ccts:AssociatedObjectClassTermName>Status
</xsd:documentation>
</xsd:annotation>
</xsd:element>
<xsd:element name="Name" type="NameType" minOccurs="0"
maxOccurs="unbounded">
<xsd:annotation>
<xsd:documentation xml:lang="en">
<ccts:DictionaryEntryName> <ccts:DictionaryEntryName>Account. Name
specific account</ccts:Definition> <ccts:Definition>The text name for a
<ccts:Cardinality>0..n</ccts:Cardinality>
<ccts:SequencingKey>3</ccts:SequencingKey>
<ccts:ObjectClassTermName>Account
</ccts:ObjectClassTermName> <ccts:PropertyTermName>Name
</ccts:PropertyTermName> <ccts:RepresentationTermName>Name
</ccts:RepresentationTermName> <ccts:RepresentationTermName>Name
</xsd:documentation>
</xsd:annotation>
</xsd:element>
<xsd:element name="CurrencyCode" type="CurrencyCodeType"
minOccurs="0" maxOccurs="unbounded">
<xsd:annotation>
<xsd:documentation xml:lang="en">
<ccts:DictionaryEntryName>Account.
Currency. Code</ccts:DictionaryEntryName>
<ccts:Definition>A code specifying the
currency in which monies are held within the account.</ccts:Definition>
<ccts:Cardinality>0..n</ccts:Cardinality>
<ccts:SequencingKey>4</ccts:SequencingKey>
<ccts:ObjectClassTermName>Account
</ccts:ObjectClassTermName> <ccts:PropertyTermName>Currency
</ccts:PropertyTermName> <ccts:RepresentationTermName>Code
</ccts:RepresentationTermName> <ccts:RepresentationTermName>Code
</xsd:documentation>
</xsd:annotation>
</xsd:element>
<xsd:element name="TypeCode" type="AccountTypeCodeType"
minOccurs="0" maxOccurs="unbounded">
<xsd:annotation>
<xsd:documentation xml:lang="en">
<ccts:DictionaryEntryName>Account. Type.
Code</ccts:DictionaryEntryName>
<ccts:Definition>This provides the ability
to indicate what type of account this is (checking, savings,
etc).</ccts:Definition>
<ccts:Cardinality>0..1</ccts:Cardinality>
<ccts:SequencingKey>5</ccts:SequencingKey>
<ccts:ObjectClassTermName>Account
</ccts:ObjectClassTermName> <ccts:PropertyTermName>Type
</ccts:PropertyTermName> <ccts:RepresentationTermName>Code
</ccts:RepresentationTermName> <ccts:RepresentationTermName>Code
</xsd:documentation>
</xsd:annotation>
</xsd:element>
<xsd:element name="Country" type="CountryType" minOccurs="0"
maxOccurs="unbounded">
<xsd:annotation>
<xsd:documentation xml:lang="en">
<ccts:UniqueID>UN00000007</ccts:UniqueID>
<ccts:Version>3.0</ccts:Version>

```

3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690
3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707

```

Country</ccts:DictionaryEntryName>      <ccts:DictionaryEntryName>Account.
                                         <ccts:Definition>Country information
related to account details.</ccts:Definition>
                                         <ccts:Cardinality>0..n<ccts:Cardinality>
                                         <ccts:SequencingKey>6</ccts:SequencingKey>
                                         <ccts:ObjectClassTermName>Account
</ccts:ObjectClassTermName>
                                         <ccts:PropertyTermName>Country
</ccts:PropertyTermName>
                                         <ccts:AssociationType>Composite
</ccts:AssociationType>
                                         <ccts:AssociatedObjectClassTermName>Country
</ccts:AssociatedObjectClassTermName>
                                         </xsd:documentation>
                                         </xsd:annotation>
                                         </xsd:element>
                                         <xsd:element name="Person" type="PersonType" minOccurs="0"
maxOccurs="unbounded">
                                         <xsd:annotation>
                                         <xsd:documentation xml:lang="en">
                                         <ccts:UniqueID>UN00000008</ccts:UniqueID>
                                         <ccts:Version>3.0</ccts:Version>
                                         <ccts:DictionaryEntryName>Account.
Person</ccts:DictionaryEntryName>
                                         <ccts: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.</ccts:Definition>
                                         <ccts:Cardinality>0..n<ccts:Cardinality>
                                         <ccts:SequencingKey>7</ccts:SequencingKey>
                                         <ccts:ObjectClassTermName>Account
</ccts:ObjectClassTermName>
                                         <ccts:PropertyTermName>Person
</ccts:PropertyTermName>
                                         <ccts:AssociationType>Composite
</ccts:AssociationType>
                                         <ccts:AssociatedObjectClassTermName>Person
</ccts:AssociatedObjectClassTermName>
                                         </xsd:documentation>
                                         </xsd:annotation>
                                         </xsd:element>
                                         <xsd:element name="Organisation" type="OrganisationType"
minOccurs="0" maxOccurs="unbounded">
                                         <xsd:annotation>
                                         <xsd:documentation xml:lang="en">
                                         <ccts:UniqueID>UN00000009</ccts:UniqueID>
                                         <ccts:Version>3.0</ccts:Version>
                                         <ccts:DictionaryEntryName>Account.
Organisation</ccts:DictionaryEntryName>
                                         <ccts: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.</ccts:Definition>
                                         <ccts:Cardinality>0..n<ccts:Cardinality>
                                         <ccts:SequencingKey>8</ccts:SequencingKey>
                                         <ccts:ObjectClassTermName>Account
</ccts:ObjectClassTermName>
                                         <ccts:PropertyTermName>Organisation
</ccts:PropertyTermName>
                                         <ccts:AssociationType>Composite
</ccts:AssociationType>
                                         <ccts:AssociatedObjectClassTermName>
Organisation</ccts:AssociatedObjectClassTermName>
                                         </xsd:documentation>
                                         </xsd:annotation>
                                         </xsd:element>
                                         </xsd:sequence>
</xsd:complexType>

```

3708

3709 **Example B-10: Complete Structure**

```

3710 <?xml version="1.0" encoding="UTF-8"?>
3711 <!-- ===== -->
3712 <!-- ===== [SCHEMA MODULE TYPE] Schema Module ===== -->
3713 <!-- ===== -->
3714 <!-- -->
3715 Schema agency: [SCHEMA AGENCY NAME]
3716 Schema version: [SCHEMA VERSION]
3717 Schema date: [DATE OF SCHEMA]
3718
3719
3720 [Code list name:] [NAME OF CODE LIST]
3721 [Code list agency:] [CODE LIST AGENCY]
3722 [Code list version:] [VERSION OF CODE LIST]
3723 [Identifier list name:] [NAME OF IDENTIFIER LIST]
3724 [Identifier list agency:] [IDENTIFIER LIST AGENCY]
3725 [Identifier list version:] [VERSION OF IDENTIFIER LIST]
3726
3727
3728 Copyright (C) UN/CEFACT (2006). All Rights Reserved.
3729
3730 This document and translations of it may be copied and furnished to others, and
3731 derivative works that comment on or otherwise explain it or assist in its
3732 implementation may be prepared, copied, published and distributed, in whole or in
3733 part, without restriction of any kind, provided that the above copyright notice and
3734 this paragraph are included on all such copies and derivative works. However, this
3735 document itself may not be modified in any way, such as by removing the copyright
3736 notice or references to UN/CEFACT, except as needed for the purpose of developing
3737 UN/CEFACT specifications, in which case the procedures for copyrights defined in
3738 the UN/CEFACT Intellectual Property Rights document must be followed, or as
3739 required to translate it into languages other than English.
3740
3741 The limited permissions granted above are perpetual and will not be revoked by
3742 UN/CEFACT or its successors or assigns.
3743
3744 This document and the information contained herein is provided on an "AS IS" basis
3745 and UN/CEFACT DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT
3746 LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE
3747 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR
3748 PURPOSE.
3749 -->
3750 <xsd:schema
3751 targetNamespace="urn:un:unece:unefact:data:draft:[MODULENAME]:[VERSION]"
3752 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
3753 ... FURTHER NAMESPACE ...
3754 elementFormDefault="qualified" attributeFormDefault="unqualified">
3755 <!-- ===== -->
3756 <!-- ===== Include ===== -->
3757 <!-- ===== -->
3758 <!-- ===== Inclusion of [TYPE OF MODULE] ===== -->
3759 <!-- ===== -->
3760 <xsd:include schemaLocation="..." />
3761 <!-- ===== -->
3762 <!-- ===== Imports ===== -->
3763 <!-- ===== -->
3764 <!-- ===== Import of [TYPE OF MODULE] ===== -->
3765 <!-- ===== -->
3766 <xsd:import namespace="..." schemaLocation="..." />
3767 <!-- ===== -->
3768 <!-- ===== Element Declarations ===== -->
3769 <!-- ===== -->
3770 <!-- ===== Root element ===== -->
3771 <!-- ===== -->
3772 <xsd:element name="[ELEMENTNAME]" type="[TOKEN]:[TYPENAME]" />
3773 <!-- ===== -->
3774 <!-- ===== Global Element Declarations ===== -->
3775 <!-- ===== -->
3776 <xsd:element name="[ELEMENTNAME]" type="[TOKEN]:[TYPENAME]" />
3777 <!-- ===== -->
3778 <!-- ===== Type Definitions ===== -->
3779 <!-- ===== -->
3780 <!-- ===== Type Definition: [TYPE] ===== -->
3781 <!-- ===== -->
3782 <xsd:complexType name="[TYPENAME]" />

```

3783
3784
3785

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

3786 **Appendix C. ATG Approved Acronyms and Abbreviations**

3787 The following constitutes a list of ATG approved acronyms and abbreviations which
3788 must be used within tag names when these words are part of the dictionary entry
3789 name:

3790 ABIE – Aggregate Business Information Entity

3791 ACC – Aggregate Core Component

3792 BBIE – Basic Business Information Entity

3793 BCC – Basic Core Component

3794 BDT – Business Data Type

3795 BIE – Business Information Entity

3796 CC – Core Component

3797 ID – Identifier

3798 URI – Uniform Resource Identifier

3799 URL – Uniform Resource Locator

3800 URN – Uniform Resource Name

3801 UUID – Universally Unique Identifier

3802 **Appendix D. Core Component XML Schema File**

3803 The Core Component XML Schema File is published as a separate file in the
3804 UN/CEFACT Data Common file structure as CoreComponentType_3p0.xsd.

3805 **Appendix E. Business Data Type XML Schema File**

3806 The reference Business Data Type XML Schema File is published in the
3807 UN/CEFACT Data Common file structure as BusinessDataType_3p0.xsd.

3808

Appendix F. Annotation Templates

00
01
02
03
04
05
06
07
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- ===== -->
<!-- ===== XMLNDR Documentation Schema File ===== -->
<!-- ===== -->
<!--
Schema agency:      UN/CEFACT
Schema version:     3.0
Schema date:        16 November 2009

Copyright (C) UN/CEFACT (2009). 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
targetNamespace="urn:un:unece:unefact:data:ordermanagement:1:draft"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="urn:un:unece:unefact:data:ordermanagement:1:draft"
xmlns:xbt="urn:un:unece:unefact:data:common:1:draft"
xmlns:clm6Recommendation20="urn:un:unece:unefact:codelist:common:6:standard:6:Reco
mmendation20:6"
xmlns:clm60133="urn:un:unece:unefact:codelist:common:1:standard:6:0133:40106"
xmlns:clm5ISO6392A="urn:un:unece:unefact:codelist:common:2009-06-
02:standard:5:iso6392A:2009-06-02"
xmlns:clm5ISO42173A="urn:un:unece:unefact:codelist:common:2009-03-
05:standard:5:ISO42173A:2009-03-05"
xmlns:ids5ISO316612A="urn:un:unece:unefact:identifierlist:common:
SecondEdition2006VI-4:standard:5:ISO316612A:SecondEdition2006VI-4"
xmlns:clmIANAMIMEMediaType="urn:un:unece:unefact:codelist:common:2009-03-
04:standard:IANA:MIMEMediaType:2009-03-04"
xmlns:clmIANACHaracterSetCode="urn:un:unece:unefact:codelist:common:2007-05-
14:standard:IANA:CharacterSetCode:2007-05-14"
xmlns:clm63055="urn:un:unece:unefact:codelist:common:D08B:standard:6:3055:D08B"
xmlns:ccts="urn:un:unece:unefact:documentation:common:3:standard:CoreComponentsTec
hnicalSpecification:3"
elementFormDefault="qualified"
attributeFormDefault="unqualified"
version="1.0">
<!-- ===== -->
<!-- ===== Include ===== -->
<!-- ===== -->
<!-- ===== Inclusion of same package BIE XML Schema File ===== -->
<!-- ===== -->
<xsd:include schemaLocation="BusinessInformationEntity_3p0.xsd"/>
<!-- ===== -->
<!-- ===== Inclusion of same package BDT XML Schema File ===== -->
<!-- ===== -->
<xsd:include schemaLocation="BusinessDataType_3p0.xsd"/>
<!-- ===== -->
<!-- ===== Inclusion of same package Business Code List XML Schema File ===== -->
<!-- ===== -->
<xsd:include schemaLocation="BusinessCodeList_1p0.xsd"/>

```



```

<!-- ===== -->
<!-- ===== -->
<!-- ===== Imports ===== -->
<!-- ===== -->
<xsd:import namespace="urn:un:unece:unefact:data:common:1:draft"
schemaLocation="../../../data/common/1/draft/XMLBuilt-InType.xsd"/>
<!-- ===== -->
<!-- ===== Import of Code lists ===== -->
<!-- ===== -->
<xsd:import namespace="urn:un:unece:unefact:codelist:common:2001:standard:5:4217"
schemaLocation="../../../codelist/common/2001/standard/ISO_CurrencyCode_2001.xsd"
"/>
...
<!-- ===== -->
<!-- ===== Import of Identifier Schemes ===== -->
<!-- ===== -->
<xsd:import
namespace="urn:un:unece:unefact:identifierlist:standard:5:ISO6393A:2008-11-07"
schemaLocation="../../../identifierlist/standard/ISO_ISOCodesForTheRepresentatio
nOfNamesOfLanguages_2008-11-07.xsd"/>
...
<!-- ===== -->
<!-- ===== Import of Other Root Schema ===== -->
<!-- ===== -->
<xsd:import namespace="urn:un:unece:unefact:data:common:1:standard"
schemaLocation="../../../common/1/standard/CommonRoot.xsd"/>
...
<!-- ===== -->
<!-- ===== Import of Other BIE ===== -->
<!-- ===== -->
<xsd:import namespace="urn:un:unece:unefact:data:common:1:standard"
schemaLocation="../../../common/1/standard/BIE.xsd"/>
...

```

3913

F.1 Annotation Documentation

```

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

```

33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
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

```
<xsd:group name="BBIEDocumentation">  
  <xsd:sequence>  
    <xsd:element name="DictionaryEntryName" type="NameType"/>  
    <xsd:element name="Definition" type="TextType"/>  
    <xsd:element name="Cardinality" type="NumericType"/>  
    <xsd:element name="SequencingKey" type="NumericType"/>  
    <xsd:element name="ObjectClassQualifierName" type="NameType"  
minOccurs="0" maxOccurs="unbounded"/>  
    <xsd:element name="ObjectClassTermName" type="NameType"/>  
    <xsd:element name="PropertyTermName" type="NameType"/>  
    <xsd:element name="PropertyQualifierName" type="NameType"  
minOccurs="0" maxOccurs="unbounded"/>  
    <xsd:element name="RepresentationTermName" type="NameType"/>  
    <xsd:element name="BusinessTermName" type="NameType" minOccurs="0"  
maxOccurs="unbounded"/>  
  </xsd:sequence>  
</xsd:group>  
<xsd:group name="ASBIEDocumentation">  
  <xsd:sequence>  
    <xsd:element name="UniqueID" type="EntityUniqueIdentifierType"/>  
    <xsd:element name="VersionID" type="VersionIdentifierType"/>  
    <xsd:element name="DictionaryEntryName" type="NameType"/>  
    <xsd:element name="Definition" type="TextType"/>  
    <xsd:element name="Cardinality" type="NumericType"/>  
    <xsd:element name="SequencingKey" type="TextType"/>  
    <xsd:element name="ObjectClassQualifierName" type="NameType"  
minOccurs="0" maxOccurs="unbounded"/>  
    <xsd:element name="ObjectClassTermName" type="NameType"/>  
    <xsd:element name="PropertyQualifierName" type="NameType"  
minOccurs="0" maxOccurs="unbounded"/>  
    <xsd:element name="PropertyTermName" type="NameType"/>  
    <xsd:element name="AssociationType" type="AssociationTypeCodeType"/>  
    <xsd:element name="AssociatedObjectClassQualifierName" type="NameType" minOccurs="0" maxOccurs="unbounded"/>  
    <xsd:element name="AssociatedObjectClassName" type="NameType"/>  
    <xsd:element name="BusinessTermName" type="NameType" minOccurs="0" maxOccurs="unbounded"/>  
  </xsd:sequence>  
</xsd:group>  
<xsd:group name="BDTDocumentation">  
  <xsd:sequence>  
    <xsd:element name="UniqueID" type="EntityUniqueIdentifierType"/>  
    <xsd:element name="VersionID" type="VersionIdentifierType"/>  
    <xsd:element name="DictionaryEntryName" type="NameType"/>  
    <xsd:element name="Definition" type="TextType"/>  
    <xsd:element name="BusinessTermName" type="NameType" minOccurs="0" maxOccurs="unbounded"/>  
    <xsd:element name="DataTypeTermName" type="NameType"/>  
    <xsd:element name="DataQualifierTermName" type="NameType" minOccurs="0"/>  
  </xsd:sequence>  
</xsd:group>  
<!-->  
<xsd:group name="ContentComponentValueDomain">  
  <xsd:sequence>  
    <xsd:element name="Definition" type="TextType"/>  
    <xsd:element name="DefaultIndicator" type="IndicatorType"/>  
    <xsd:element name="PrimitiveTypeName" type="NameType"/>  
    <xsd:element name="SchemeOrListID" type="IDType" minOccurs="0"/>  
    <xsd:element name="SchemeOrListVersionID" type="IDType"/>  
    <xsd:element name="SchemeOrListAgencyID" type="IDType"/>  
minOccurs="0"/>  
    <xsd:element name="SchemeOrListModificationAllowedIndicator" type="IndicatorType" minOccurs="0"/>  
    <xsd:element name="DefaultValue" type="ValueType" minOccurs="0"/>  
  </xsd:sequence>  
</xsd:group>  
<!-->
```

```

4011 <xsd:group name="BDTSCDocumentation">
4012 <xsd:sequence>
4013 <xsd:element name="Cardinality" type="NumericType"/>
4014 <xsd:element name="DictionaryEntryName" type="NameType"/>
4015 <xsd:element name="Definition" type="TextType"/>
4016 <xsd:element name="PropertyTermName" type="NameType"/>
4017 <xsd:element name="RepresentationTermName" type="NameType"/>
4018 <xsd:element name="DataTypeTermName" type="NameType"/>
4019 <xsd:element name="DataTypeQualifierTermName" type="NameType"/>
4020 </xsd:sequence>
4021 </xsd:group>
4022 <!-->
4023 <xsd:group name="SupplementaryComponentValueDomainType">
4024 <xsd:sequence>
4025 <xsd:element name="DefaultIndicator" type="IndicatorType"/>
4026 <xsd:element name="PrimitiveTypeName" type="NameType"/>
4027 <xsd:element name="SchemeOrListID" type="IDType" minOccurs="0"/>
4028 <xsd:element name="SchemeOrListVersionID" type="IDType"
4029 minOccurs="0"/>
4030 <xsd:element name="SchemeOrListAgencyID" type="IDType"
4031 minOccurs="0"/>
4032 <xsd:element name="SchemeOrListModificationAllowedIndicator"
4033 type="IndicatorType" minOccurs="0"/>
4034 <xsd:element name="DefaultValue" type="ValueType" minOccurs="0"/>
4035 </xsd:sequence>
4036 </xsd:group>
4037 <!-->
4038 <xsd:group name="SchemeOrListDocumentation">
4039 <xsd:sequence>
4040 <xsd:element name="SchemeOrListID" type="IDType"/>
4041 <xsd:element name="SchemeOrListVersionID" type="IDType"
4042 minOccurs="0"/>
4043 <xsd:element name="SchemeOrListAgencyID" type="IDType"
4044 minOccurs="0"/>
4045 <xsd:element name="SchemeOrListModificationAllowedIndicator"
4046 type="IndicatorType"/>
4047 </xsd:sequence>
4048 </xsd:group>
4049 <xsd:group name="SchemeOrListValueDocumentation">
4050 <xsd:sequence>
4051 <xsd:element name="Name" type="NameType"/>
4052 <xsd:element name="Description" type="TextType" minOccurs="0"
4053 maxOccurs="unbounded"/>
4054 </xsd:sequence>
4055 </xsd:group>

```

4056 F.2 Annotation Application Information

```

4057 <xsd:element name="BusinessContext">
4058 <xsd:complexType>
4059 <xsd:sequence>
4060 <xsd:element name="ContextUnit" maxOccurs="unbounded">
4061 <xsd:complexType>
4062 <xsd:sequence>
4063 <xsd:element
4064 name="BusinessProcessContextCategory"
4065 type="ccts:BusinessProcessContextCategoryType" minOccurs="0"
4066 maxOccurs="unbounded"/>
4067 <xsd:element
4068 name="BusinessProcessRoleContextCategory"
4069 type="ccts:BusinessProcessRoleContextCategoryType" minOccurs="0"
4070 maxOccurs="unbounded"/>
4071 <xsd:element
4072 name="SupportingRoleContextCategory" type="ccts:SupportingRoleContextCategoryType"
4073 minOccurs="0" maxOccurs="unbounded"/>
4074 <xsd:element
4075 name="IndustryClassificationContextCategory"
4076 type="ccts:IndustryClassificationContextCategoryType" minOccurs="0"
4077 maxOccurs="unbounded"/>
4078 <xsd:element
4079 name="ProductClassificationContextCategory"
4080 type="ccts:ProductClassificationContextCategoryType" minOccurs="0"
4081 maxOccurs="unbounded"/>

```

```

40882         <xsd:element
40883 name="GeopoliticalContextCategory" type="ccts:GeopoliticalContextCategoryType"
40884 minOccurs="0" maxOccurs="unbounded" />
40885     </xsd:element>
40886     <xsd:element
40887 name="OfficialConstraintsContextCategory"
40888 type="ccts:OfficialConstraintsContextCategoryType" minOccurs="0"
40889 maxOccurs="unbounded" />
40890     </xsd:element>
40891     <xsd:element
40892 name="SystemCapabilitiesContextCategory"
40893 type="ccts:SystemCapabilitiesContextCategoryType" minOccurs="0"
40894 maxOccurs="unbounded" />
40895     </xsd:element>
40896 </xsd:sequence>
40897 </xsd:complexType>
40898 </xsd:element>
40899 </xsd:sequence>
40900 </xsd:complexType>
40901 </xsd:element>
40902 <xsd:attribute name="id" type="EntityUniqueIdentifierType" />
40903 <xsd:attribute name="versionID" type="VersionIdentifierType" />
40904 </xsd:complexType>
40905 </xsd:element>
40906 <xsd:complexType name="BusinessInformationContextCategoryType">
40907 <xsd:sequence>
40908 <xsd:element name="BusinessInformationEntityID" type="IDType"
40909 maxOccurs="unbounded" />
40910 <xsd:element name="ContextExclusion" minOccurs="0">
40911 <xsd:complexType>
40912 <xsd:sequence>
40913 <xsd:element
40914 name="BusinessInformationEntityID" type="IDType" maxOccurs="unbounded" />
40915 </xsd:sequence>
40916 </xsd:complexType>
40917 </xsd:element>
40918 </xsd:sequence>
40919 <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean" />
40920 </xsd:complexType>
40921 <xsd:complexType name="BusinessProcessContextCategoryType">
40922 <xsd:sequence>
40923 <xsd:element name="BusinessProcessCode" minOccurs="0"
40924 maxOccurs="unbounded">
40925 <xsd:complexType>
40926 <xsd:complexContent>
40927 <xsd:extension base="CodeType" />
40928 </xsd:complexContent>
40929 </xsd:complexType>
40930 </xsd:element>
40931 <xsd:element name="ContextExclusion" minOccurs="0">
40932 <xsd:complexType>
40933 <xsd:sequence>
40934 <xsd:element name="BusinessProcessTypeCode"
40935 type="CodeType" maxOccurs="unbounded" />
40936 </xsd:sequence>
40937 </xsd:complexType>
40938 </xsd:element>
40939 </xsd:sequence>
40940 <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean" />
40941 </xsd:complexType>
40942 <xsd:complexType name="BusinessProcessRoleContextCategoryType">
40943 <xsd:sequence>
40944 <xsd:element name="BusinessProcessRoleCode" type="CodeType"
40945 minOccurs="0" maxOccurs="unbounded" />
40946 <xsd:element name="ContextExclusion" minOccurs="0">
40947 <xsd:complexType>
40948 <xsd:sequence>
40949 <xsd:element name="PartyFunctionCode"
40950 type="CodeType" maxOccurs="unbounded" />
40951 </xsd:sequence>
40952 </xsd:complexType>
40953 </xsd:element>
40954 </xsd:sequence>
40955 <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean" />
40956 </xsd:complexType>
40957 <xsd:complexType name="SupportingRoleContextCategoryType">
40958 <xsd:sequence>
40959 <xsd:element name="SupporterFunctionCode" minOccurs="0"
40960 maxOccurs="unbounded">
40961 <xsd:complexType>
40962 <xsd:complexContent>

```

4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168
4169
4170
4171
4172
4173
4174
4175
4176
4177
4178
4179
4180
4181
4182
4183
4184
4185
4186
4187
4188
4189
4190
4191
4192
4193
4194
4195
4196
4197
4198
4199
4200
4201
4202
4203
4204
4205
4206
4207
4208
4209
4210
4211
4212
4213
4214
4215
4216
4217
4218
4219
4220
4221
4222
4223
4224
4225
4226
4227
4228
4229
4230
4231
4232
4233

```
        <xsd:extension base="CodeType" />
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="ContextExclusion" minOccurs="0">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="SupporterFunctionCode"
type="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="CodeType"
minOccurs="0" maxOccurs="unbounded" />
    <xsd:element name="ContextExclusion" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="IndustryTypeCode"
type="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="CodeType"
minOccurs="0" maxOccurs="unbounded" />
    <xsd:element name="ContextExclusion" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="ProductTypeCode"
type="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>
    <xsd:element name="GeopoliticalCode" minOccurs="0"
maxOccurs="unbounded" />
    <xsd:element name="ContextExclusion" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element ref="clm54217:CurrencyCode"
maxOccurs="unbounded" />
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean" />
</xsd:complexType>
<xsd:complexType name="OfficialConstraintsContextCategoryType">
  <xsd:sequence>
    <xsd:element name="OfficialConstraintsCode" minOccurs="0"
maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:complexContent>
          <xsd:extension base="CodeType" />
        </xsd:complexContent>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="ContextExclusion" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="LawTypeCode"
type="CodeType" maxOccurs="unbounded" />
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
```

```
4234         </xsd:complexType>
4235     </xsd:element>
4236 </xsd:sequence>
4237 <xsd:attribute name="inAllContextsListIndicator" type="xsd:boolean"/>
4238 </xsd:complexType>
4239 <xsd:complexType name="SystemCapabilitiesContextCategoryType">
4240 <xsd:sequence>
4241 <xsd:element name="SystemCapabilitiesID" type="IDType"
4242 minOccurs="0" maxOccurs="unbounded"/>
4243 <xsd:element name="ContextExclusion" minOccurs="0">
4244 <xsd:complexType>
4245 <xsd:sequence>
4246 <xsd:element name="SoftwareSolutionID"
4247 type="IDType" maxOccurs="unbounded"/>
4248 </xsd:sequence>
4249 </xsd:complexType>
4250 </xsd:element>
4251 </xsd:sequence>
4252 <xsd:attribute name="inAllContextsIndicator" type="xsd:boolean"/>
4253 </xsd:complexType>
4254 <xsd:element name="UsageRule" type="ccts:UsageRuleType"/>
4255 <xsd:complexType name="UsageRuleType">
4256 <xsd:sequence>
4257 <xsd:element name="UniqueID" type="EntityUniqueIdentifierType"/>
4258 <xsd:element name="Constraint" type="TextType"/>
4259 <xsd:element name="ConstraintTypeCode" type="CodeType"/>
4260 <xsd:element name="ConditionTypeCode"
4261 type="ConditionTypeCodeType"/>
4262 <xsd:element name="Name" type="NameType" minOccurs="0"/>
4263 <xsd:element name="BusinessTerm" type="TextType" minOccurs="0"
4264 maxOccurs="unbounded"/>
4265 </xsd:sequence>
4266 </xsd:complexType>
4267 </xsd:schema>
```

4268 **Appendix G. UN/CEFACT Data Type Catalogue**

4269 The UN/CEFACT Data Type (DT) Catalogue Version 3.0 identifies the data types
4270 needed to support model development of core components and business information
4271 entities.

4272 Appendix H. Use Cases for Code Lists

4273 Code lists provide mechanisms for conveying data in a consistent fashion where all
 4274 parties to the information – originator, sender, receiver, processor – fully understand
 4275 the purpose, use, and meaning of the data. This specification support flexible use of
 4276 code lists. This appendix details the mechanisms for this use.

4277 The five alternative uses for code lists are:

- 4278 • Referencing a predefined standard code list, such as ISO 4217 currency
 4279 codes as a supplementary component in an BDT, such as AmountType.
- 4280 • Referencing any code list, standard or proprietary, by providing the required
 4281 identification as attributes in the BDT CodeType.
- 4282 • Referencing a predefined code list by declaring a specific BDT.
- 4283 • Choosing or combining values from several code lists.
- 4284 • Restricting the set of allowed code values from an established code list.

4285 Example H-1 is a code snippet from an XML Schema File that uses each of these.

4286 Example H-1: Code Use Example Schema

```

4287 <xsd:schema xmlns:ordman=":un:unece:cefact:data:ordermanagement:1:draft"
4288 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4289 targetNamespace="urn:un:unece:cefact:data:ordermanagement:1:draft"
4290 elementFormDefault="qualified" attributeFormDefault="unqualified">
4291 <!-- ===== Include ===== -->
4292 <xsd:include
4293 schemaLocation="http://www.unece.org/unecefact/data/ordermanagement/1/draft/Business
4294 InformationEntity_lp3p6.xsd"/>
4295 <xsd:include
4296 schemaLocation="http://www.unece.org/unecefact/data/ordermanagement/1/draft/Business
4297 DataType_lp3p6.xsd"/>
4298
4299 <!-- Root element -->
4300 <xsd:element name="Invoice" type="ordman:InvoiceType"/>
4301 <!-- Messase type declaration -->
4302 <xsd:complexType name="InvoiceType">
4303 <xsd:sequence>
4304 <xsd:element name="Product" type="ordman:ProductType"/>
4305 <xsd:element name="CustomerParty" type="ordman:PartyType"/>
4306 </xsd:sequence>
4307 </xsd:complexType>
4308 <!-- The below type declaration would normally appear in a separate schema module
4309 for all reusable components (ABIE) but is included here for completeness -->
4310 <xsd:complexType name="ProductType">
4311 <xsd:sequence>
4312 <xsd:element name="TotalAmount" type="ordman:AmountType_SADASS"/>
4313 <xsd:element name="TaxCurrencyCode"
4314 type="ordman:CodeType_ADSFSS"/>
4315 <xsd:element name="ChangeCurrencyCode"
4316 type="ordman:CurrencyCodeType_ICX874"/>
4317 <xsd:element name="CalculationCurrencyCode"
4318 type="ordman:CalculationCurrencyCodeType_09123X"/>
4319 <xsd:element name="RestrictedCurrencyCode"
4320 type="ordman:RestrictedCurrencyCodeType_23CVBS"/>
4321 </xsd:sequence>
4322 </xsd:complexType>
4323 </xsd:schema>
  
```

4324 This schema includes:

- 4325 • The BDT XML Schema File defined for the given context category (business
4326 process value which is order management).
- 4327 ○ The two specific data types `CurrencyCodeType` and
4328 `CalculationCurrencyCodeType` are defined as Business Code List that
4329 are included through the BDT XML Schema File.

- 4330 • The BIE XML Schema File defined for the given context category.

4331 The `xsd:complexType` named “ProductType” includes five local elements. Each of
4332 these elements represents one of the five different code list options.

4333 H.1 Referencing a Common Code List as a Supplementary 4334 Component in a Business Data Type

4335 In Example H-1, the element `TotalAmount` is declared as shown in Example H-2.

4336 Example H-2: Declaration of `TotalAmount` Element

4337

```
<xsd:element name="TotalAmount" type="ordman:Amountclm5ISO42173AType_090909"/>
```

4338 As shown in the element declaration, `TotalAmount` is of the generic CCT `AmountType`
4339 that is implemented in the the data package using the primitive decimal and the CCL
4340 ISO code list 42173A resulting in the BDT `Amountclm5ISO42173AType_090909` which has
4341 been defined in the BDT XML Schema File. The `Amountclm5ISO42173A` Type
4342 declaration is as show in Example H-3.

4343 Example H-3: Declaration of `Amount` DataTypes in the BDT

4344

```
<xsd:schema targetNamespace="urn:un:unece:uncefact:data:ordermanagement:1:draft"
4345       xmlns:clm54217="urn:un:unece:uncefact:odelist:common:1:draft:5:4217:2001" ...
4346       elementFormDefault="qualified" attributeFormDefault="unqualified">
4347       <!-- =====
4348       <!-- ===== Imports
4349       <!-- =====
4350       <!-- ===== Imports of Code Lists
4351       <!-- =====
4352       <xsd:import namespace="urn:un:unece:uncefact:odelist:common:1:draft:5:4217:2001"
4353       schemaLocation="
4354       http://www.unece.org/unecefact/odelist/common/1/draft/5/4217_2001_.xsd" />
4355       <!-- =====
4356       <!-- ===== Type Definitions
4357       <!-- =====
4358       <!-- ===== Amount Decimal. Type
4359       <!-- =====
4360       <xsd:complexType name="Amountclm5ISO42173AType_090909">
4361        <xsd:simpleContent>
4362         <xsd:extension base="xsd:decimal">
4363         <xsd:attribute name="currencyCode"
4364         type="clm5ISO42173A:IS03AlphaCurrencyCodeContentType" use="optional"/>
4365         </xsd:extension>
4366        </xsd:simpleContent>
4367       </xsd:complexType>
```

4369 The `AmountType` has attributes declared that represent the supplementary components
4370 defined in CCTS for this data type. These attributes include `currencyCode` for the
4371 supplementary component of `Amount`. `Currency`. `code`. This `currencyCode` attribute is
4372 declared to be of the `xsd:simpleType`

4373 `clm5ISO42173A:IS03AlphaCurrencyCodeContentType`. The
4374 `clm5ISO42173A:IS03AlphaCurrencyCodeContentType` has been declared in the code list
4375 schema module for ISO Currency Codes, and the allowed code values for the

4376 `currencyCode` attribute have been defined as enumeration facets in the
 4377 `clm5ISO42173A:ISO3AlphaCurrencyCodeContentType` type definition.

4378 An extract of the CCL XML Schema File for the ISO Currency Codes is shown in H-
 4379 4.

4380 **Example H-4: Declaration of a Currency Code List**

```

4381 <!-- ===== -->
4382 <!-- ===== Root Element Declarations ===== -->
4383 <!-- ===== -->
4384 <xsd:element name="CurrencyCode" type="clm54217:CurrencyCodeContentType" />
4385 <!-- ===== -->
4386 <!-- ===== Type Definitions ===== -->
4387 <!-- ===== -->
4388 <!-- ===== Code List Type Definition: Currency Codes ===== -->
4389 <!-- ===== -->
4390 <xsd:simpleType name="CurrencyCodeContentType">
4391   <xsd:restriction base="xsd:token">
4392     <xsd:enumeration value="AED">
4393       <xsd:annotation>
4394         <xsd:documentation>
4395           ... see the section for Code Value Documentation ...
4396         </xsd:documentation>
4397       </xsd:annotation>
4398     </xsd:enumeration>
4399     <xsd:enumeration value="AFN">
4400       <xsd:annotation>
4401         <xsd:documentation>
4402           ... see the section for Code Value Documentation ...
4403         </xsd:documentation>
4404       </xsd:annotation>
4405     </xsd:enumeration>
4406   </xsd:restriction>
4407 </xsd:simpleType>
4408 </xsd:schema>

```

4409 The `currencyCode` attribute has a fixed value of ISO 4217 Currency Code as defined
 4410 in CCTS. Only code values from this code list are allowed in a CEFACCT conformant
 4411 instance documents. The resulting instance documents conveyance currency code
 4412 values are represented as:

```

4413 <TotalAmount currencyCode="AED">3.14</TotalAmount>

```

4414 [Note:]

4415 When using this option no information about the code list used is carried in the
 4416 instance document as this is already defined in the XML Schema.

4417 **H.2 Referencing any code list using BDT CodeType**

4418 The second element in our example message – `TaxCurrencyCode` – is of the BDT
 4419 CodeType.

```

4420 <xsd:element name="TaxCurrencyCode" type="CodeType_56432" />

```

4421 This `CodeType` data type includes a number of supplementary components required
 4422 in order to uniquely identify the code list to be used for validation.

4423 The `CodeType` is declared in the BDT XML Schema File as shown in Figure H-5

4424

4425 **Example H-5: Declaration of a Code Type in the BDT XML Schema File**

```

4426 <xsd:complexType name="CodeType_56432">
4427 <xsd:simpleContent>
4428 <xsd:extension base="xsd:token">
4429 <xsd:attribute name="listID" type="xsd:token" use="optional"/>
4430 <xsd:attribute name="listAgencyID" type="xsd:token" use="optional"/>
4431 <xsd:attribute name="listVersionID" type="xsd:token" use="optional"/>
4432 </xsd:extension>
4433 </xsd:simpleContent>
4434 </xsd:complexType>

```

4435 When the `codeType` is used, either the `listID` indicates the Code List identification.
 4436 The `listAgencyID` is the Agency identification that made the code list available. The
 4437 `listVersionID` indicates the version of the code list.

4438 The association to the specific values must be made at runtime. In an instance
 4439 document this element could be represented as:

```

4440 <TaxCurrencyCode listID="ISO 4217" listVersionID="2001"
4441 listAgencyID="5">AED</TaxCurrencyCode>

```

4442 It should be noted that when applying this option, validation of code values in the
 4443 instance document will not be done by the XML parser.

4444 **H.3 Referencing a Common Code List in a BDT**

4445 The third element in our example message `ChangeCurrencyCode` is based on the
 4446 business data type `CurrencyCodeType`.

```

4447 <xsd:element name="ChangeCurrencyCode" type="CurrencyCodeType_A28945"/>

```

4448 The `CurrencyCodeType` would be defined in the BDT XML Schema File as:

```

4449 <xsd:simpleType name=" CurrencyCodeType_A28945">
4450 <xsd:restriction base="clm54217-A:CurrencyCodeContentType"/>
4451 </xsd:simpleType>

```

4452 This means that the value of the `ChangeCurrencyCode` element can only have code
 4453 values from the identified ISO 4217 code list. In an instance document this element
 4454 would be represented as:

```

4455 <ChangeCurrencyCode>AED</ChangeCurrencyCode>

```

4456 [Note:]

4457 When using this option no information about the code list used is carried in the
 4458 instance document as this is already defined in the XML Schema.

4459 **H.4 Choosing or Combining Values from Several Code Lists**

4460 The fourth option is to combine values from diverse code lists by using the
 4461 `xsd:union` element.

4462 The `xsd:union` code list approach enables multiple code lists to be used for a
 4463 single element or attribute. The element declaration in the XML Schema, the element
 4464 `CalculationCurrencyCode` is based on the namespace specific BCL type

4465 defined in the context category specific namespace BCL XML Schema File where
4466 the **ordman: CalculationCurrencyCodeclmType_D982143** is declared.

```
4467 <xsd:element name="CalculationCurrencyCode"  
4468 type="ordman:CalculationCurrencyCodeclmType_ D982143" />
```

4469 The **ordman:CalculationCurrencyCodeclmType_D982143** is defined in the
4470 BCL XML Schema File with in the context category namespace for Order
4471 Management, using an **xsd:union** element that unions the code lists together.

```
4472 <xsd:simpleType name="CalculationCurrencyCodeType_D982143">  
4473 <xsd:union memberTypes="clm54217-N:CurrencyCodeContentType  
4474 clm54217-A:CurrencyCodeContentType" />  
4475 </xsd:simpleType>
```

4476 This allows values to come from either the **clm54217-**
4477 **N:CurrencyCodeContentType** or from the **clm54217-**
4478 **A:CurrencyCodeContentType**. The CCL XML Schema File for **clm54217-**
4479 **A:CurrencyCodeContentType** is the same as the one used earlier in this
4480 Appendix. The CCL XML Schema File for **clm54217-**
4481 **N:CurrencyCodeContentType** is the same as the one used earlier in this
4482 Appendix.

4483 The **xsd:union** allows the use of code values from different pre-defined code lists
4484 in instance documents. The code lists must be imported once in the BCL XML
4485 Schema File. The specific code list will be represented by the namespace prefixes
4486 (**clm54217-A** or **clm54217-N**), the element in the instance document will not have
4487 the specific code list tokens conveyed as the first part of the element name. The
4488 recipient of the instance does not know unambiguously which code list each code
4489 value is defined. This is because a reference to the specific code lists comes from
4490 different Code List XML Schema Files, in this case, **clm54217-N** and **clm54217-A**.

4491 In an instance document this element could be represented as:

```
4492 <Invoice >  
4493 ...  
4494 <CalculationCurrencyCode>840</CalculationCurrencyCode>  
4495 ...  
4496 </Invoice>
```

4497 The advantage of the **xsd:union** is that attributes can also make use of these code
4498 lists.

4499 [Note:]

4500 When using this option no information about the code list used is carried in the
4501 instance document as this is already defined in the XML Schema.

4502 H.5 Restricting the Allowed Code Values

4503 This option is used when it is desired to reduce the number of allowed code values
4504 from an existing code list. For example, a trading partner community may only
4505 recognize certain code values from the ISO 4217 Currency Code list. To accomplish
4506 this, create a BCL XML Schema File within the specific context category namespace

4507 of the XML Schema Files that use it. This BCL XML Schema File simply contains the
4508 restricted set of values used by the context category.

4509 This is accomplished by importing the CCL XML Schema File and using
4510 **xsd:restriction** to restrict the values to the set of values required. For more
4511 please see section [8.5.3.4 Type Definitions](#).

4512 **Appendix I. Alternative Business Message Syntax** 4513 **Binding**

4514 UN/CEFACT will create the XML syntax binding of its CCTS conformant BIE data
4515 models directly from the associations and hierarchies expressed in the Business
4516 Message Template for each business message exchange. This approach is based
4517 on traditional nesting of all components of the data model.

4518 The XML Schema Specification also supports an alternative to nesting. This
4519 alternative, using schema identity constraints (`xsd:key/xsd:unique/xsd:keyRef`),
4520 enables referencing and reuse of a given element in instance documents.
4521 UN/CEFACT is currently evaluating this alternative for future use to include a method
4522 for application at the data model level. In anticipation that the data model issues will
4523 be resolved, UN/CEFACT has already developed a set of rules for its XML
4524 implementation. These rules and the supporting narrative are presented in this
4525 Appendix. Organizations using this Alternative Method will still be considered
4526 conformant to this specification, if they adhere to all other conformance requirements
4527 and use the rules defined in this Appendix.

4528 **I.1 XML Schema Architecture**

4529 **I.1.1 Message Assembly Considerations**

4530 If referencing between specific ABIE's is required in the scope of the root Message
4531 Assembly (MA) or of a lower level ABIE, the Business Message Template must
4532 specify the list of ABIE's that are implemented as referenced rather than nested
4533 properties. This will allow the identity constraints to be generated in the message
4534 schema.

4535 **I.1.2. Requirements for XML Element Referencing**

4536 **I.1.2.1 Implementation of Aggregations – Nesting or Referencing**

4537 Since aggregations relate ABIEs that have independent life cycles, the same
4538 instance of a particular ABIE may be referenced more than once within a message.

4539 The ClaimNotify message shown below, taken from the Insurance Industry, illustrate
4540 this.

4541 In Example I-1 and Example I-2 the same Person *John Smith* can play the role of
4542 *Insured* in the Policy ABIE and the role of *Claimant* in the Claim ABIE. In order to
4543 reduce redundancy in the message, it is possible to use XML referencing to relate
4544 one Person instance to the Policy and Claim instances as an alternate method to
4545 nesting information about Person within Policy and Claim.

4546 In general, when the level of reuse of an instance ABIE in a message is significant it
4547 becomes adequate to use XML referencing for the purpose of removing redundancy
4548 from the message and increasing information integrity.

4549

4550 **Example I-1: XML Instance of ClaimNotify using nesting**

```

4551 <ClaimNotify>
4552 .....
4553 <Claim>
4554   <ClaimantParty>
4555     <Name>John Smith</Name>
4556   </ClaimantParty>
4557 </Claim>
4558 .....
4559 <Policy>
4560   <InsuredParty>
4561     <Name>John Smith</Name>
4562   </InsuredParty>
4563 </Policy>
4564 </ClaimNotify>

```

4565 **Example I-2: XML Instance of ClaimNotify using referencing**

```

4566 <ClaimNotify>
4567 .....
4568 <Party key="P1">
4569   <Name>John Smith</Name>
4570 </Party>
4571 <Claim>
4572 <ClaimantParty partyReference="P1"/>
4573 </Claim>
4574 .....
4575 <Policy>
4576   <InsuredParty partyReference="P1"/>
4577 </Policy>
4578 .....
4579 </ClaimNotify>

```

4580 **I.1.2.2 Other Usages of XML Referencing**

4581 Another requirement for XML element referencing is *Dynamic Referencing*.

4582 The requirement is that any element composing a message is potentially the target
 4583 of a reference for the purpose of building dynamic relationships between elements
 4584 within the message. An important use case is identification of faulty elements for
 4585 error reporting.

4586 **I.1.2.3 Schema Validation Requirements for XML References**4587 **I.1.2.3.1 Structural References between Aggregated ABIEs**

4588 For structural references between ABIEs, the level of validation performed by the
 4589 XML Schema definition of a message should be as strong as if the referenced
 4590 element would have been defined as a nested child of the element that references it.
 4591 Thus, the schema must strictly enforce identity constraints, i.e.:

- 4592 1. Check uniqueness of the identifiers of the referenced elements
- 4593 2. Check that the references match the identifiers of the corresponding
 4594 referenced elements.

4595 Due to its more robust identity constraints, this specification mandates **key/keyRef**
 4596 as the XML referencing technique to be used instead of **Id/IdRef**. See sections
 4597 [7.1.5 Constraints on Schema Construction](#), [I.2.1.1 Constraints on Schema](#)
 4598 [Construction](#) and [I.3.1.1 Declaration of the Referencing Constraints](#).

4599 Referencing between ABIEs occur in the boundaries of a particular 'scope element'
4600 in the XML document. The scope element is the container of all the elements that
4601 can be involved in the identity constraints. These identity constraints act as follows:

- 4602 • The uniqueness (xsd:unique) or key (xsd:key) constraints define the keys and
4603 enforce that a value is unique within the scope element.

4604 The key reference (xsd:keyRef) constraints define the key references and enforce
4605 that a value corresponds to a value represented by a uniqueness (xsd:unique) or key
4606 (xsd:key) constraint.

4607 Most often the scope element will be the message root element but it can also be
4608 another element lower in the hierarchy. The XML Schema language requires that the
4609 key-keyref constraints be defined within a scope element.

4610 **I.1.2.3.2 Dynamic References**

4611 For dynamic references schema validation is not required. Since dynamic
4612 referencing is only used for ancillary purposes, it is not deemed essential to enforce
4613 uniqueness of identifiers in the schema when they are not involved in structural
4614 referencing. Uniqueness of such identifiers should be granted by use of adequate
4615 algorithms for the generation of the identifiers. This will avoid unnecessary
4616 complexity of the identity constraints.

4617 **I.2 General XML Schema Language Conventions**

4618 **I.2.1 Overall XML Schema Structure and Rules**

4619 **I.2.1.1 Constraints on Schema Construction**

4620 The XML Schema **xsd:key**, **xsd:keyref** or **xsd:unique** identity constraints
4621 have the following characteristics that make them preferable to the
4622 **xsd:ID/xsd>IDREF** technique.

- 4623 • The keys and relationships between objects are strongly typed. They are
4624 declared explicitly in the schema. Each relationship is distinctly defined and
4625 specifies exactly which object has a key, what is the key, which other objects
4626 can link to this object and through which element or attribute. You can prevent
4627 an object to point to an arbitrary object that has an identifier attribute, as it is
4628 the case with the ID/IDREF method.
- 4629 • The scope of key uniqueness is precisely defined among one or several
4630 objects within a particular instance of an XML element. It is not more
4631 necessary to ensure uniqueness of id attributes across the whole XML
4632 document.
- 4633 • The elements or attributes used as keys or key references can be of any data
4634 type, not only ID or IDRef (implying the NMTOKEN format). This allows any
4635 element or attribute to be used for linking.

4636 The following principles are taken into account for the implementation of schema
4637 identity constraints:

- 4638 1. Identifiers and references used in schema identity constraints will be
 4639 attributes. This has the advantage that the data element content of the XML
 4640 complex types derived from ABIEs is kept unchanged
- 4641 2. For maximum element and type reuse and to stay away from forward
 4642 compatibility problems, attributes used as identifiers or references will be
 4643 optional. This means that no key (`xsd:key`) constraints should be defined on
 4644 identifiers, which would make the identifiers mandatory in the context of a
 4645 message; only uniqueness (`xsd:unique`) constraints must be used.
- 4646 3. Only the ABIEs that are part of a logical aggregation implemented by XML
 4647 referencing will be subject to explicit schema identity constraints. For all other
 4648 ABIEs - which may only be involved in dynamic references - uniqueness of
 4649 identifiers should be granted by use of adequate algorithms for the generation
 4650 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

4651 **I.2.2 Attribute and Element Declarations**

4652 **I.2.2.1 Attributes**

4653 Attributes are only used in two cases:

- 4654
- 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.
- 4655
4656
4657

[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

4658 **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
----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

4659 **I.3 XML Schema Files**4660 **I.3.1 Root XML Schema Files**4661 **I.3.1.1 Declaration of the Referencing Constraints**

4662 Referencing between ABIEs occurs within the limits defined by a particular 'scope'
4663 element in the XML document tree.

4664

4665 The scope element is the container of all the elements that can be involved in the
4666 identity constraints. The schema language requires that the identity constraints be
4667 contained in the schema declaration of the scope element.

4668 Most often the scope element will be the message root element, but it can also be
4669 another element lower in the hierarchy.

4670 The identifier attribute of each ABIE that is part of a logical aggregation implemented
4671 by XML referencing will be subject to a uniqueness (**xsd:unique**) constraint
4672 defined in the scope element. The name of the **xsd:unique** constraint must be
4673 unique in the schema.

4674 The uniqueness (**xsd:unique**) constraints define the keys and enforce that a value
4675 is unique within the scope element.

4676 The key reference (**xsd:keyRef**) constraints define the key references and enforce
4677 that a value corresponds to a value represented by a uniqueness (**xsd:unique**)
4678 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

4679 This declaration will guarantee uniqueness of the identifier attribute values across all
4680 referenced elements of the same name, in the given scope.

4681 [Note:]
 4682 The value of `xsd:selector/@xpath` identifies instances of one element in one
 4683 namespace (by default the namespace of the XML Schema File in which the
 4684 `xsd:selector` is declared.).

4685 In Example I-3 the declaration under the message root element will guarantee
 4686 uniqueness of the `@key` attribute values across all `bie:Party` elements, in the
 4687 scope of the `rsm:ClaimNotify` message.

4688 Example I-3: Unique Declaration

```
4689 <xsd:unique name="ClaimNotifyPartyKey">
4690 <xsd:selector xpath="bie:Party"/>
4691 <xsd:field xpath="@key"/>
4692 </xsd:unique>
```

4693 For each referenced ABIE used in a given scope, corresponding key reference
 4694 (`xsd:keyRef`) declarations must be made. Naming conventions used for key
 4695 reference attributes, as exposed in I.3.2.2, are such that only one key reference
 4696 (`xsd:keyRef`) declaration is needed for all the elements where the key reference
 4697 attribute appears.

[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: <code><Scope Element><Referenced Element>Reference</code> 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

4698 In Example I-4 the declaration under the message root element will enforce
 4699 referencing between all the elements that have the `@PartyReference` attribute
 4700 and instances of `bie:Party`, in the scope of the `rsm:ClaimNotify` message.

4701 Example I-4: Key Reference Declaration

```
4702 <xsd:keyref name="ClaimNotifyPartyReference" refer="ClaimNotifyPartyKey">
4703 <xsd:selector xpath="."/>
4704 <xsd:field xpath="@partyReference"/>
4705 </xsd:keyref>
```

4706 [Note:]
 4707 The value of `xsd:selector/@xpath` allows for any element in any namespace to
 4708 be the parent element of the reference attribute in the `xsd:keyref` constraint.

4709 Dynamic referencing does not require the schema to enforce uniqueness of `@key`
 4710 attributes when they are not involved in structural referencing. This will avoid
 4711 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
----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---

4712 **I.3.2 Business Information Entities XML Schema Files**

4713 **I.3.2.1 Type Definitions**

4714 Every aggregate business information entity (ABIE) `xsd:complexType` definition
 4715 will include an optional identifier attribute that may be used for both dynamic and
 4716 structural referencing. It will be defined as a local attribute named “key” to avoid any
 4717 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, <code>key</code> attribute that MAY be used as the complex element identifier in the XML document where it appears.	1
[R 92C0]	<code>key</code> MUST be a reserved attribute name.	1
[R 8A37]	Every <code>key</code> local attribute declaration MUST be of the type <code>xsd:token</code> .	1

4718 **I.3.2.2 Element Declarations and References**

4719 **9.5.1.1.1 I.3.2.2.1 ASBIE Elements**

4720 For each ASBIE who's `ccts:AggregationKind` value=`shared`, there are two
 4721 mutually exclusive cases, one of which needs to be selected on the base of the
 4722 applicable Message Assembly definition.

- 4723 • The globally declared element for the associated ABIE is included in the
 4724 content model of the parent ABIE as a nested complex property.
- 4725 • An equivalent referencing element pointing to the associated ABIE is included
 4726 in the content model of the parent ABIE.

4727 See section [5.4 Reusability Schema](#) and [I.1.1 Message Assembly Considerations](#)
 4728 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

4729

4730 Example I-5 shows the schema definition of an ASBIE specified as a referencing
 4731 element.

4732 **Example I-5: Element and type definition of an ASBIE, specified as a referencing element**

4733
 4734
 4735
 4736

```
<xsd:complexType name="PartyReferenceType">
  <xsd:attribute name="partyReference" type="xs:token"/>
</xsd:complexType>
<xsd:element name="ClaimantParty" type="PartyReferenceType"/>
```

4737 **Appendix J. Date. Type, DateTime. Type and Time. Type**
 4738 **Data Type Representations and Their Translation to XML**
 4739 **Schema Types**

4740 Editors Note: Section maybe updated based on discrepancies found by Serge.

4741 The value domain and representation of Date. Type, DateTime. Type and Time.

4742 Type are based on a single **TimePoint** primitive.

4743 These types are provided inorder to support the ISO 8601 formats. These types are
 4744 expressed in the BDT XML Schema File using the appropriate XML Schema base
 4745 type from the common XBT XML Schema File that is applicable to the BDT using the
 4746 translation tables Table J-1 for Date. Type, Table J-2 for Time. Type and Table J-3
 4747 for DateTime. Type.

4748 Table J-1 shows the Date. Type and the corresponding ISO 8601 formats.

ISO 8601 Format Code	Default Indicator	xbt: Base Type	xsd: Base Type	Pattern restriction
YYYY-MM-DD	true	DateType	xsd:date	[0-9]{4}-[0-1][0-9]-[0-3][0-9]
YYYY-MM		YearMonthType	xsd:gYearMonth	[0-9]{4}-[0-1][0-9]
YYYY		N/A	xsd:gYear	N/A
--MM-DD		MonthDayType	xsd:gMonthDay	--[0-1][0-9]-[0-3][0-9]
--MM--		MonthType	xsd:gMonth	--[0-1][0-9]--
---DD		DayType	xsd:gDay	---[0-3][0-9]
YYYY-DDD		YearDayType	xsd:token	[0-9]{4}-[0-3] [0-9]{2}
-DDD		DayOfYearType	xsd:token	-[0-3] [0-9]{2}
YYYY-Www-D		YearWeekDayType	xsd:token	[0-9]{4}-W[0-5] [0-9]-[1-7]
-Www-D		WeekDayType	xsd:token	-W[0-5] [0-9] -[1-7]
YYYY-Www		YearWeekType	xsd:token	[0-9]{4}-W[0-5] [0-9]
-Www		WeekType	xsd:token	-W[0-5] [0-9]
-W-D		DayOfWeekType	xsd:token	-W-[1-7]

4749 **Table J-1: Date. Type**

4750 When more than one format is allowed (e.g. variable precision), the base types will
 4751 be **unioned** within the given BDT.

4752 Variable precision is needed for example in the case of a Birth Date, where the full
 4753 date may not be know or may not be shared, Example J-1 shows the types that
 4754 maybe unioned to accomplish this.

4755

4756 **Example J-1 XML Schema of a BDT that unions two date XBT.**

```

4757 ...
4758 <xsd:simpleType name="BirthDateType_192837">
4759   <xsd:union memberTypes="xbt:DateType xbt:YearMonthType" />
4760   ...
4761   <xsd:attribute name="formatCode" type="
4762   clm6XXXX1:DateFormatCodeContentType" use="optional">
4763   ...
4764   </xsd:attribute
4765   </xsd:simpleType>
4766 ...
    
```

4767 **XML instance of the type using the default base type:**

```

4768 <BirthDate>1985-04-12</BirthDate>
    
```

4769 or

4770 **XML instance of the type not using the default base type:**

```

4771 <BirthDate formatCode="YYYY-MM">1985-04</BirthDate>
    
```

4772 Table J-2 shows the Time. Type and the corresponding ISO 8601 formats.

ISO 8601 Format	Default Indicator	xbt: Base Type	xsd: Base Type	Pattern restriction
hh:mm:ss	true	TimeType	xsd:time	[0-2] [0-9]: [0-5] [0-9]: [0-5] [0-9].[0-9]*
hh:mm:ss+hh:mm		TimeUTCType	xsd:time	[0-2] [0-9]: [0-5] [0-9]: [0-5] [0-9].[0-9]*[+ -] [0-2] [0-9]: [0-6] [0-9]
hh:mm:ssZ		TimeZuluType	xsd:time	[0-2] [0-9]: [0-5] [0-9]: [0-5] [0-9].[0-9]*Z
hh:mm		HourMinuteType	xsd:token	[0-2] [0-9]:[0-5] [0-9]
hh		HourType	xsd:token	[0-2] [0-9]
-mm:ss		MinuteSecondType	xsd:token	-[0-5] [0-9]:[0-5] [0-9].[0-9]*
-mm		MinuteType	xsd:token	-[0-5] [0-9]
--ss		SecondType	xsd:token	--[0-5] [0-9]

4773 **Table J-2: Time. Type**

- 4774 [Note:] - Conventions on time formats:
 4775 Second decimals are allowed and optional
 4776 UTC and Zulu time are only available for hh:mm:ss format.

4777 Table J-3 shows the DateTime Data Type using the ISO 8601 formats.

ISO 8601 Format Code	Default Indicator	xbt: Base Type	xsd: Base Type	Pattern restriction
YYYY-MM-DDThh:mm:ss	true	DateTimeType	xsd:dateTime	[0-9]{4}-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*
YYYY-MM-DDThh:mm:ss+hh:mm		DateTimeUTCType	xsd:dateTime	[0-9]{4}-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*[+ -][0-2][0-9]:[0-6][0-9]
YYYY-MM-DDThh:mm:ssZ		DateTimeZuluType	xsd:dateTime	[0-9]{4}-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*Z
YYYY-MM-DDThh:mm		DateHourMinuteType	xsd:token	[0-9]{4}-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]:[0-5][0-9]
YYYY-MM-DDThh		DateHourType	xsd:token	[0-9]{4}-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]
--MM-DDThh:mm:ss		MonthDayTimeType	xsd:token	--[0-1][0-9]-[0-3][0-9]T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*
--MM-DDThh:mm:ss+hh:mm		MonthDayTimeUTCType	xsd:token	--[0-1][0-9]-[0-3][0-9]T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*[+ -][0-2][0-9]:[0-6][0-9]
--MM-DDThh:mm:ssZ		MonthDayTimeZuluType	xsd:token	--[0-1][0-9]-[0-3][0-9]T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*Z
--MM-DDThh:mm		MonthDayHourMinuteType	xsd:token	--[0-1][0-9]-[0-3][0-9]T[0-2][0-9]:[0-5][0-9]
--MM-DDThh		MonthDayHourType	xsd:token	--[0-1][0-9]-[0-3][0-9]T[0-2][0-9]
---DDThh:mm:ss		DayTimeType	xsd:token	---[0-3][0-9]T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*
---DDThh:mm:ss+hh:mm		DayTimeUTCType	xsd:token	---[0-3][0-9]T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*[+ -][0-2][0-9]:[0-6][0-9]

ISO 8601 Format Code	Default Indicator	xbt: Base Type	xsd: Base Type	Pattern restriction
---DDThh:mm:ssZ		DayTimeZuluType	xsd:token	---[0-3][0-9]T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*Z
---DDThh:mm		DayHourMinuteType	xsd:token	---[0-3][0-9]T[0-2][0-9]:[0-5][0-9]
---DDThh		DayHourType	xsd:token	---[0-3][0-9]T[0-2][0-9]
YYYY-DDDThh:mm:ss		YearDayTimeType	xsd:token	[0-9]{4}-[0-3][0-9]{2}T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*
YYYY-DDDThh:mm:ss+hh:mm		YearDayTimeUTCType	xsd:token	[0-9]{4}-[0-3][0-9]{2}T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*[+ -][0-2][0-9]:[0-6][0-9]
YYYY-DDDThh:mm:ssZ		YearDayTimeZuluType	xsd:token	[0-9]{4}-[0-3][0-9]{2}T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*Z
YYYY-DDDThh:mm		YearDayHourMinuteType	xsd:token	[0-9]{4}-[0-3][0-9]{2}T[0-2][0-9]:[0-5][0-9]
YYYY-DDDThh		YearDayHourType	xsd:token	[0-9]{4}-[0-3][0-9]{2}T[0-2][0-9]
-DDDThh:mm:ss		DayOfYearTimeType	xsd:token	-[0-3][0-9]{2}T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*
-DDDThh:mm:ss+hh:mm		DayOfYearTimeUTCType	xsd:token	-[0-3][0-9]{2}T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*[+ -][0-2][0-9]:[0-6][0-9]
-DDDThh:mm:ssZ		DayOfYearTimeZuluType	xsd:token	-[0-3][0-9]{2}T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*Z
-DDDThh:mm		DayOfYearHourMinuteType	xsd:token	-[0-3][0-9]{2}T[0-2][0-9]:[0-5][0-9]
-DDDThh		DayOfYearHourType	xsd:token	-[0-3][0-9]{2}T[0-2][0-9]
YYYY-Www-DThh:mm:ss		YearWeekDayTimeType	xsd:token	[0-9]{4}-W[0-5][0-9]-[1-7]T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*
YYYY-Www-DThh:mm:ss+hh:mm		YearWeekDayTimeUTCType	xsd:token	[0-9]{4}-W[0-5][0-9]-[1-7]T[0-2][0-9]:[0-5][0-9]:[0-5][0-9].[0-9]*[+ -][0-2][0-9]:[0-6][0-9]

ISO 8601 Format Code	Default Indicator	xbt: Base Type	xsd: Base Type	Pattern restriction
YYYY-Www-DThh:mm:ssZ		YearWeekDayTimeZuluType	xsd:token	[0-9]{4}-W[0-5] [0-9]-[1-7]T[0-2][0-9]:[0-5] [0-9]:[0-5] [0-9]*Z
YYYY-Www-DThh:mm		YearWeekDayHourMinute	xsd:token	[0-9]{4}-W[0-5] [0-9]-[1-7]T[0-2][0-9]:[0-5] [0-9]
YYYY-Www-DThh		YearWeekDayHourType	xsd:token	[0-9]{4}-W[0-5] [0-9]-[1-7]T[0-2][0-9]
-Www-DThh:mm:ss		WeekDayTimeType	xsd:token	-W[0-5][0-9]-[1-7]T[0-2][0-9]:[0-5] [0-9]:[0-5] [0-9].[0-9]*
-Www-DThh:mm:ss+hh:mm		WeekDayTimeUTCType	xsd:token	-W[0-5][0-9]-[1-7]T[0-2][0-9]:[0-5] [0-9]:[0-5] [0-9].[0-9]*[+ -][0-2][0-9]:[0-6][0-9]
-Www-DThh:mm:ssZ		WeekDayTimeZuluType	xsd:token	-W[0-5][0-9]-[1-7]T[0-2][0-9]:[0-5] [0-9]:[0-5] [0-9].[0-9]*Z
-Www-DThh:mm		WeekDayHourMinuteType	xsd:token	-W[0-5][0-9]-[1-7]T[0-2][0-9]:[0-5] [0-9]
-Www-DThh		WeekDayHourType	xsd:token	-W[0-5][0-9]-[1-7]T[0-2][0-9]
-W-DThh:mm:ss		DayOfWeekTimeType	xsd:token	-W-[1-7] T[0-2][0-9]:[0-5] [0-9]:[0-5] [0-9].[0-9]*
-W-DThh:mm:ss+hh:mm		DayOfWeekTimeUTCType	xsd:token	-W-[1-7] T[0-2][0-9]:[0-5] [0-9]:[0-5] [0-9].[0-9]*[+ -][0-2][0-9]:[0-6][0-9]
-W-DThh:mm:ssZ		DayOfWeekTimeZuluType	xsd:token	-W-[1-7] T[0-2][0-9]:[0-5] [0-9]:[0-5] [0-9].[0-9]*Z
-W-DThh:mm		DayOfWeekHourMinuteType	xsd:token	-W-[1-7] T[0-2][0-9]:[0-5] [0-9]
-W-DThh		DayOfWeekHourType	xsd:token	-W-[1-7] T[0-2][0-9]

4778 **Table J-3: DateTime. Type (combinations of Date and Time representations)**

4779 [Note:]

4780 The use of regular expressions: Regular expressions cannot validate the date-time
4781 value space to the same extent as the xsd built-in types; they can only validate the
4782 lexical space.

4783 Example J-2 shows the XBT XML Schema File that defines the types expressed in
4784 the Table J-1, Table J-2 and Table J-3.

4785

4786
4787

Example J-2 XBT XML Schema File that expresses additional built-in types to support ISO 8601.

4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800
4801
4802
4803
4804
4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- ===== -->
<!-- ===== XML Schema Builtin Type Extension XML Schema File ===== -->
<!-- ===== -->
<!--
Schema agency:      UN/CEFACT
Schema version:     1.0 Draft A
Schema date:        29 July 2009

Copyright (C) UN/CEFACT (2009). 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.

Copyright (C) UN/CEFACT (2009). All Rights Reserved.
-->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:un:unece:unefact:data:common:1:draft"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <!-- ===== -->
  <!-- ===== Type Definitions ===== -->
  <!-- ===== -->
  <!-- ===== Duration types ===== -->
  <!-- ===== -->
  <xsd:simpleType name="WeekDurationType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: nW</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="\d+W"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- ===== -->
  <!-- ===== Date types ===== -->
  <!-- ===== -->
  <xsd:simpleType name="DateType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: YYYY-MM-DD</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:date">
      <xsd:pattern value="[0-9]{4}-[0-1][0-9]-[0-3][0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="DayType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: ---DD</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:gDay">
      <xsd:pattern value="---[0-3][0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>

```


49936
49937
49938
49939
49940
49941
49942
49943
49944
49945
49946
49947
49948
49949
49950
49951
49952
49953
49954
49955
49956
49957
49958
49959
49960
49961
49962
49963
49964
49965
49966
49967
49968
49969
49970
49971
49972
49973
49974
49975
49976
49977
49978
49979
49980
49981
49982
49983
49984
49985
49986
49987
49988
49989
49990
49991
49992
49993
49994
49995
49996
49997
49998
49999
50000
50001
50002
50003
50004
50005
50006
50007
50008
50009
50010

```

    <xsd:restriction base="xsd:token">
      <xsd:pattern value="[0-9]{4}-W[0-5] [0-9]-[1-7]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!------->
  <!--===== Time types =====>
  <!------->
  <xsd:simpleType name="HourLocalType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: hh</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="[0-2] [0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="HourMinuteLocalType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: hh:mm</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="[0-2] [0-9]:[0-5] [0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="MinuteType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -mm</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="-[0-5] [0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="MinuteSecondType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -mm:ss</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="-[0-5] [0-9]:[0-5] [0-9].[0-9]*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="SecondType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: --ss</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="--[0-5] [0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="TimeLocalType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: hh:mm:ss</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:time">
      <xsd:pattern value="[0-9]{4}-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]: [0-5] [0-9]:[0-5] [0-9].[0-9]*[+|-][0-2][0-9]:[0-6] [0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="TimeUTCType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: hh:mm:ssZ</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:time">
      <xsd:pattern value="[0-2] [0-9]: [0-5] [0-9]: [0-5] [0-9].[0-9]*Z"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="TimeUTCOffsetType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format:
hh:mm:ss+hh:mm</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:time">
      <xsd:pattern value="[0-2] [0-9]: [0-5] [0-9]: [0-5] [0-9].[0-9]*[+|-] [0-2] [0-9]: [0-6] [0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!------->

```

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

```

<!--===== DateTime types =====>
<!--=====
<xsd:simpleType name="DateHourLocalType">
  <xsd:annotation>
    <xsd:documentation>ISO 8601 format: YYYY-MM-
DDTh</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:token">
    <xsd:pattern value="[0-9]{4}-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]: [0-
5] [0-9]:[0-5] [0-9].[0-9]*/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="DateHourMinuteLocalType">
  <xsd:annotation>
    <xsd:documentation>ISO 8601 format: YYYY-MM-
DDTh:mm</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:token">
    <xsd:pattern value="[0-9]{4}-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]: [0-
5] [0-9]:[0-5] [0-9].[0-9]*/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="DateTimeLocalType">
  <xsd:annotation>
    <xsd:documentation>ISO 8601 format: YYYY-MM-
DDTh:mm:ss</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:dateTime">
    <xsd:pattern value="[0-9]{4}-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]: [0-
5] [0-9]:[0-5] [0-9].[0-9]*/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="DateTimeUTCOffsetType">
  <xsd:annotation>
    <xsd:documentation>ISO 8601 format: YYYY-MM-
DDTh:mm:ss+hh:mm</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:dateTime">
    <xsd:pattern value="[0-9]{4}-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]: [0-
5] [0-9]:[0-5] [0-9]*[+|-][0-2][0-9]:[0-6] [0-9]*/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="DateTimeUCType">
  <xsd:annotation>
    <xsd:documentation>ISO 8601 format: YYYY-MM-
DDTh:mm:ssZ</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:dateTime">
    <xsd:pattern value="[0-9]{4}-[0-1][0-9]-[0-3][0-9]T[0-2][0-9]: [0-
5] [0-9]:[0-5] [0-9].[0-9]*Z"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="DayHourLocalType">
  <xsd:annotation>
    <xsd:documentation>ISO 8601 format: ---DDTh</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:token">
    <xsd:pattern value="---[0-3][0-9]T[0-2][0-9]*/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="DayHourMinuteLocalType">
  <xsd:annotation>
    <xsd:documentation>ISO 8601 format: ---
DDTh:mm</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:token">
    <xsd:pattern value="---[0-3][0-9]T[0-2][0-9]:[0-5][0-9]*/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="DayTimeLocalType">
  <xsd:annotation>
    <xsd:documentation>ISO 8601 format: ---
DDTh:mm:ss</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:token">

```


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38

```

    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="DayOfYearHourMinuteLocalType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -DDDThh:mm</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="-[0-3] [0-9]{2} T[0-2][0-9]:[0-5] [0-9]" />
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="DayOfYearTimeLocalType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -
DDDThh:mm:ss</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="-[0-3] [0-9]{2} T[0-2][0-9]: [0-5] [0-9]:[0-5]
[0-9].[0-9]*" />
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="DayOfYearTimeUTCOffsetType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -
DDDThh:mm:ss+hh:mm</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="-[0-3] [0-9]{2} T[0-2][0-9]: [0-5] [0-9]:[0-5]
[0-9].[0-9]*[\+|-][0-2][0-9]:[0-6][0-9]" />
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="DayOfYearTimeUTCType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -
DDDThh:mm:ssZ</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="-[0-3] [0-9]{2} T[0-2][0-9]: [0-5] [0-9]:[0-5]
[0-9].[0-9]*[\+|-][0-2][0-9]:[0-6][0-9]" />
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="MonthDayHourLocalType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -MM-DDThh</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="--[0-1][0-9]-[0-3][0-9]T[0-2][0-9]" />
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="MonthDayHourMinuteType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -MM-
DDThh:mm</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="--[0-1][0-9]-[0-3][0-9]T[0-2][0-9]: [0-5] [0-
9]" />
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="MonthDayTimeLocalType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -MM-
DDThh:mm:ss</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="--[0-1][0-9]-[0-3][0-9]T[0-2][0-9]: [0-5] [0-
9]:[0-5] [0-9].[0-9]*" />
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="MonthDayTimeUTCOffsetType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -MM-
DDThh:mm:ss+hh:mm</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="--[0-1][0-9]-[0-3][0-9]T[0-2][0-9]: [0-5] [0-
9]:[0-5] [0-9].[0-9]*[\+|-][0-2][0-9]:[0-6][0-9]" />
  
```


5239
5240
5241
5242
5243
5244
5245
5246
5247
5248
5249
5250
5251
5252
5253
5254
5255
5256
5257
5258
5259
5260
5261
5262
5263
5264
5265
5266
5267
5268
5269
5270
5271
5272
5273
5274
5275
5276
5277
5278
5279
5280
5281
5282
5283
5284
5285
5286
5287
5288
5289
5290
5291
5292
5293
5294
5295
5296
5297
5298
5299
5300
5301
5302
5303
5304
5305
5306
5307
5308
5309
5310
5311
5312
5313
5314

```

    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="MonthDayTimeUTCType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -MM-
DDThh:mm:ssZ</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="--[0-1][0-9]-[0-3][0-9]T[0-2][0-9]: [0-5] [0-
9]:[0-5] [0-9].[0-9]*Z"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="WeekDayHourLocalType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -Www-DThh</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="-W[0-5][0-9]-[1-7]T[0-2][0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="WeekDayHourMinuteType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -Www-
DThh:mm</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="-W[0-5][0-9]-[1-7]T[0-2][0-9]:[0-5] [0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="WeekDayTimeLocalType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -Www-
DThh:mm:ss</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="-W[0-5][0-9]-[1-7]T[0-2][0-9]:[0-5] [0-9]:[0-
5] [0-9].[0-9]*"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="WeekDayTimeUTCOffsetType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -Www-
DThh:mm:ss+hh:mm</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="-W[0-5][0-9]-[1-7]T[0-2][0-9]:[0-5] [0-9]:[0-
5] [0-9].[0-9]*[+|-][0-2][0-9]:[0-6][0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="WeekDayTimeUTCType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: -Www-
DThh:mm:ssZ</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="-W[0-5][0-9]-[1-7]T[0-2][0-9]:[0-5] [0-9]:[0-
5] [0-9].[0-9]*Z"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="YearDayHourLocalType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: YYYY-
DDDThh</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="[0-9]{4}-[0-3] [0-9]{2}T[0-2][0-9]"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="YearDayHourMinuteLocalType">
    <xsd:annotation>
      <xsd:documentation>ISO 8601 format: YYYY-
DDDThh:mm</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:pattern value="[0-9]{4}-[0-3] [0-9]{2}T[0-2][0-9]: [0-5] [0-
9]"/>

```


5391
5392
5393
5394
5395
5396

```
<xsd:restriction base="xsd:token">  
  <xsd:pattern value="[0-9]{4}-W[0-5] [0-9]-[1-7]T[0-2][0-9]:[0-5]  
[0-9]:[0-5] [0-9].[0-9]*Z"/>  
</xsd:restriction>  
</xsd:simpleType>  
</xsd:schema>
```

5397

Appendix K. Naming and Design Rules List

Rule Number	Rule Description	Category																		
[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="418 583 1320 1682"> <thead> <tr> <th colspan="2" data-bbox="418 583 1320 653">Rule Categorization</th> </tr> <tr> <th data-bbox="418 657 456 726">ID</th> <th data-bbox="459 657 1320 726">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="418 730 456 869">1</td> <td data-bbox="459 730 1320 869">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="418 873 456 1012">2</td> <td data-bbox="459 873 1320 1012">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="418 1016 456 1184">3</td> <td data-bbox="459 1016 1320 1184">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="418 1188 456 1327">4</td> <td data-bbox="459 1188 1320 1327">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="418 1331 456 1470">5</td> <td data-bbox="459 1331 1320 1470">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="418 1474 456 1570">6</td> <td data-bbox="459 1474 1320 1570">Rules that relate to extension that are determined by specific organizations.</td> </tr> <tr> <td data-bbox="418 1575 456 1682">7</td> <td data-bbox="459 1575 1320 1682">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]	<p>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.</p>	1																		

Rule Number	Rule Description	Category
[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 8EC9]	UN/CEFACT MA <code>xsd:complexType</code> definitions MUST locally declare all ASMA.	3
[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 only use lowercase alphabetic characters [a-z] , uppercase alphabetic characters [A-Z] , digit characters [0-9] or the underscore character [<code>_</code>] as 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

Rule Number	Rule Description	Category				
[R 984C]	Each organization's XML Schema components MUST be assigned to a namespace for that organization.	1				
[R 8CED]	UN/CEFACT namespaces MUST be defined as Uniform Resource Names.	3				
[R 8E2D]	<p>The XML Schema namespaces MUST use the following pattern:</p> <table border="1" data-bbox="427 621 1300 953"> <tr> <td data-bbox="427 621 537 789">URN:</td> <td data-bbox="537 621 1300 789">urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:<schema type>[:<package>]+:<major>:<status></td> </tr> <tr> <td data-bbox="427 789 537 953">URL:</td> <td data-bbox="537 789 1300 953">http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/<schema type>[/<package>]+/<major>/<status></td> </tr> </table> <p>Where:</p> <ul style="list-style-type: none"> • organization – An identifier of the organization providing the standard. • organization hierarchy – The first level of the hierarchy within the organization providing the standard. • organization hierarchy level – Zero to n level hierarchy of the organization providing the standard. • schematype – A token identifying the type of schema module: data codelist identifierscheme documentation. • package – One to n level of the packages expressed in the associated CCTS v3.0 complaint model in which the XML Schema Files expressed. Additionally, a common location is used by each of the schema types for common content. • major – The major version number. • status – The status of the schema as: draft standard. 	URN:	urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:<schema type>[:<package>]+:<major>:<status>	URL:	http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/<schema type>[/<package>]+/<major>/<status>	3
URN:	urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:<schema type>[:<package>]+:<major>:<status>					
URL:	http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/<schema type>[/<package>]+/<major>/<status>					
[R B56B]	Published namespace content MUST only be changed by the publishing organization of the namespace or its successor.	1				

Rule Number	Rule Description	Category
[R 92B8]	<p>The XML Schema File name for files other than code lists and identifier schemes MUST be of the form: <Schema Module Name>_<Version Identifier>.xsd, with periods, spaces, other separators and the words XML Schema File removed.</p> <p>Where:</p> <ul style="list-style-type: none"> • Schema Module Name – Is the name of the Schema Module. • Version Identifier – Is the major and minor version identifier. 	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 xsd:targetNamespace attribute.	1
[R 9C85]	Every XML Schema File within a single namespace version MUST also be assigned to a single file version number.	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 in the Header comment of the file after the <BusinessInformationPayload> that is expressed in the XML Schema File by using the value of the <BusinessInformationPayload> followed by the words XML Schema File .	1
[R 9961]	A Root XML Schema File MUST NOT replicate reusable constructs available in XML Schema Files that can be referenced through xsd:include or xsd:import .	1
[R 8238]	A BIE XML Schema File MUST be created within each namespace that is defined for a package.	1
[R 8252]	The BIE XML Schema Files MUST be named <i>Business Information Entity XML Schema File</i> by placing the name within the Header documentation section of the file.	1
[R A2F0]	A Reference BDT XML Schema File MUST be created in the data common namespace to represent the set of unrestricted BDTs using default value domains.	1

Rule Number	Rule Description	Category
[R AA56]	A BDT XML Schema File MUST be created within each package namespace.	1
[R 847C]	The BDT XML Schema Files MUST be named <i>Business Data Type XML Schema File</i> by placing the name within the header documentation section of the file.	1
[R 9CDD]	An XBT XML Schema File MUST be created in the data common namespace to represent the additional types not defined by XML Schema that are needed to implement the BDT equivalents of the CDTs defined in the <i>UN/CEFACT Data Type Catalogue Version 3.0</i>	1
[R 96ED]	The XBT XML Schema Files MUST be named <i>CCTS XML Builtin Types XML Schema File</i> 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 B443]	<p>A Code List XML Schema File MUST be given a file name that represents the name of the code list and is unique within the namespace to which it belongs using the form:</p> <pre><Code List Agency Identifier>_<Code List Identifier>_<Code List Version Identifier>.xsd</pre> <p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Identifier – Identifies the agency that maintains the code list. • Code List Identifier – Identifies a list of the respective corresponding codes. • Code List Version Identifier – Identifies the version of the code list. 	1

Rule Number	Rule Description	Category
[R B0AD]	<p>The semantic name of each Code List XML Schema File as defined in the comment section within the XML Schema File MUST be of the form:</p> <p><Code List Agency Name> <Code List Name> - Code List XML Schema File</p> <p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Name – Agency that maintains the code list. • 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. • The creation of a new code list that is only used within the package namespace. 	1
[R AB90]	An Identifier Scheme XML Schema File MUST be created to convey identifier scheme metadata for each scheme being used.	1
[R AD8C]	<p>An Identifier Scheme XML Schema File MUST be given a file name that represents the name of the Identifier Scheme and is unique within the namespace to which it belongs using the form:</p> <p><Identifier Scheme Agency Identifier>_<Identifier Scheme Identifier>_<Identifier Scheme Version Identifier>.xsd</p> <p>Where:</p> <ul style="list-style-type: none"> • Identifier Scheme Agency Identifier – Identifies the agency that maintains the identifier scheme. • Identifier Scheme Identifier – Identifies the scheme. • Identifier Scheme Version Identifier – Identifier of the version of the identifier scheme 	1

Rule Number	Rule Description	Category
[R A154]	<p>The semantic name of each Identifier Scheme XML Schema File as defined in the comment section within the XML Schema File MUST be of the form:</p> <p><Identifier Scheme Agency Name> < Identifier Scheme Name> - Identifier Scheme XML Schema File</p> <p>Where:</p> <ul style="list-style-type: none"> • Identifier Scheme Agency Name – Agency that maintains the identifier 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 artefacts 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> <p><xsd:schema ... version="<major>p<minor>[p<revision>]"></p> <p>Where:</p> <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. 	2

Rule Number	Rule Description	Category
[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 artefacts.	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 9EDA]	Every UN/CEFACT XML Schema File must start with an XML declaration that specifies the xml version and encoding being used.	1
[R AABF]	Every UN/CEFACT XML Schema File must use XML version 1.0.	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 xsd 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

Rule Number	Rule Description	Category
[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 <i>Core Components Technical Specification Schema File</i> 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 ccts .	1
[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
[R B221]	Supplementary Component information 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
[R B8B6]	Empty elements MUST NOT be used.	3

Rule Number	Rule Description	Category
[R 8337]	The xsd:nillable attribute MUST NOT be used.	3
[R 8608]	Anonymous types MUST NOT be used.	1
[R A4CE]	An xsd:complexType MUST be defined for each CCTS BIE.	1
[R BC3C]	An xsd:complexType MUST be defined for each CCTS BDT whose value domain cannot be fully expressed using an xsd:simpleType .	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, BCL XML Schema Files, and BIS 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 xsd:element declaration, and each xsd:complexType and xsd:simpleType definition MUST have an xsd:annotation xsd:appInfo declared that includes one or more ccts:UsageRule and one or more ccts:BusinessContext .which are used to communicate the specific usage and context that the artifact applies.	1
[R 88DE]	Usage rules MUST be expressed within the appropriate BDT, Content Component or Supplementary Component xsd:annotation xsd:appInfo ccts:UsageRule element.	1

Rule Number	Rule Description	Category
[R B851]	<p>The structure of the ccts:UsageRule element MUST be:</p> <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:ConstraintTypeCode [1..1] – The type of constraint E.g. unstructured, OCL. • ccts:ConditionTypeCode [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 MUST be created.	1
[R A538]	Each defined or declared XML Schema component MUST use an xsd:annotation and xsd:appInfo element to communicate the context of the component.	1
[R B96F]	Each Root, BIE, BDT and BCL XML Schema File MUST be defined in a unique namespace that is derived from the corresponding package within the CCTS conformant model.	1
[R B698]	The Root XML Schema File MUST include the BIE and BDT XML Schema Files that reside in its namespace.	1
[R B71D]	If a Root XML Schema File in a namespace reuses artefacts defined in another namespace, it MUST import a Root Schema File that resides in the other 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 A445]	Each ASMA component MUST be realized as a local element that is defined using the type (xsd:complexType) definition of the top level ABIE for that component.	3

Rule Number	Rule Description	Category
[R 9CC0]	The name of the local element defined for the ASMA Component MUST consist of an optional property term followed by the name of the ABIE to which it is associated.	3
[R 8837]	Each Root XML Schema File MUST define a 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 annotation documentation (xsd:annotation xsd:documentation) that contains:</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. • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the business information payload. • Definition (mandatory): The semantic meaning of 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. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the Object class. It is the basis for the DEN. • BusinessTermName (optional, repeating): A synonym term under which the payload object is known by in industry. 	1

Rule Number	Rule Description	Category
[R A86D]	<p>For every ASMA Copoment local xsd:element declaration definition, a structured set of annotations MUST contain:</p> <ul style="list-style-type: none"> • UniqueID (mandatory): The unique identifier that identifies an ASMA instance in a unique and unambiguous way. • VersionID (mandatory): An unique identifier that identifies the version of an ASMA. • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the ASMA. • Definition (mandatory): The semantic meaning of the ASMA or the underling ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differeniates the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE. • BusinessTermName (optional, repeating): A synonym term in which the ABIE is commonly known. 	1
[R 8FE2]	The BIE XML Schema File MUST use xsd:include to include the BDT XML Schema File that resides in the same namespace.	1
[R AF95]	For every object class (ABIE) identified in a data package, 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

Rule Number	Rule Description	Category
[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 its containing ABIE MUST be of an xsd:simpleType or xsd:complexType that represents the BDT that defines it.	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 each BBIE identified in an ABIE, a named xsd:element MUST be locally declared within the xsd:complexType that represents the 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 from the property term.	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]	For each BBIE element name declaration where the word Text is the representation term, the word Text MUST be removed from the name of the element or type definition.	1
[R BCD6]	Every BBIE element declaration MUST be of the BDT xsd:simpleType or xsd:complexType that represents the source BBIE business data type.	1

Rule Number	Rule Description	Category
[R 9025]	Every ASBIE whose <code>ccts:AggregationKind</code> value = composite , a local element for the associated ABIE MUST be declared in the associating ABIE <code>xsd:complexType</code> content model.	1
[R 9241]	Every ASBIE whose <code>ccts:AggregationKind</code> 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 <code>xsd:complexType</code> that represents its associated ABIE.	1
[R ACB9]	<p>For every ABIE <code>xsd:complexType</code> definition a structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements MUST contain:</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. • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the ABIE. • Definition (mandatory): The semantic meaning of the ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE. • BusinessTermName (optional, repeating): A synonym term in which the ABIE is commonly known. 	1
[R B0BA]	For every ABIE <code>xsd:complexType</code> definition a structured set of <code>xsd:annotation</code> <code>xsd:appInfo</code> elements MUST be present that fully declare its context.	1

Rule Number	Rule Description	Category
[R BCE9]	<p>For every ABIE usage rule, the ABIE <code>xsd:complexType</code> definition 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
[R 88B6]	<p>For every ABIE <code>xsd:element</code> declaration definition, a structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements MUST contain:</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. • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the ABIE. • Definition (mandatory): The semantic meaning of the ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE. • BusinessTermName (optional, repeating): A synonym term in which the ABIE is commonly known. 	1

Rule Number	Rule Description	Category
[R B8BE]	<p>For every BBIE xsd:element declaration a structured set of xsd:annotation xsd:documentation elements MUST contain:</p> <ul style="list-style-type: none"> • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the BBIE. • Definition (mandatory): The semantic meaning of the associated BBIE. • Cardinality (mandatory): Indicates the cardinality of the BBIE within the containing ABIE. • SequencingKey (mandatory): Indicates the sequence of the BBIE within the containing ABIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ABIE • PropertyTermName (mandatory): Represents a distinguishing characteristic of the BBIE. • PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the BBIE. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • RepresentationTermName (mandatory): An element of the component name that describes the form in which the BBIE is represented. • BusinessTermName (optional, repeating): A synonym term in which the BBIE is commonly known. 	1
[R 95EB]	<p>For every BBIE xsd:element declaration a structured set of xsd:annotation xsd:appInfo elements MUST be present that fully declare its context.</p>	1

Rule Number	Rule Description	Category
[R 8BF6]	<p>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:</p> <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]	<p>Every ASBIE global element declaration MUST have a structured set of <code>xsd:annotation</code> <code>xsd:documentation</code> elements that contain:</p> <ul style="list-style-type: none"> • UniqueID (mandatory): The unique identifier that identifies an ASBIE instance in a unique and unambiguous way. • VersionID (mandatory): An unique identifier that identifies the version of an ASBIE. • DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the ASBIE. • Definition (mandatory): The semantic meaning of the associated ASBIE. • ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ASBIE • PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the ASBIE. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • PropertyTermName (mandatory): Represents a distinguishing characteristic of the ASBIE. • AssociationType (mandatory): Indicates the UML AssociationKind value of <code>shared</code> or <code>composite</code> of the associated ABIE. • AssociatedObjectClassQualifierName (optional, repeating): a name or names that qualify the associated object class. 	1

Rule Number	Rule Description	Category
	<p>The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier.</p> <ul style="list-style-type: none"> AssociatedObjectClassName (Mandatory): The name of the associated object class. BusinessTermName (optional, repeating): A synonym term in which the ASBIE is commonly known. 	
[R 926A]	<p>Every ASBIE xsd:element declaration or xsd:ref occurrence within the containing ABIE MUST have a structured set of xsd:annotation xsd:documentation elements that contain:</p> <ul style="list-style-type: none"> UniqueID (mandatory): The unique identifier that identifies an ASBIE instance in a unique and unambiguous way. VersionID (mandatory): An unique identifier that identifies the version of an ASBIE. DictionaryEntryName (mandatory): The Dictionary Entry Name (DEN) of the ASBIE. Definition (mandatory): The semantic meaning of the associated ASBIE. Cardinality (mandatory): Indicates the cardinality of the ASBIE within the containing ABIE. SequencingKey (mandatory): Indicates the sequence of the ASBIE within the containing ABIE. ObjectClassQualifierName (optional, repeating): Is a word or ordered words which help define and differentiate the associated ABIE from its CC. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. ObjectClassTermName (mandatory): Is a semantically meaningful name of the object class of the ASBIE PropertyQualifierName (optional repeating): Is a word or words which help define and differentiate the ASBIE. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. PropertyTermName (mandatory): Represents a distinguishing characteristic of the ASBIE. AssociationType (mandatory): Indicates the UML AssociationKind value of shared or composite of the 	1

Rule Number	Rule Description	Category
	<p>associated ABIE.</p> <ul style="list-style-type: none"> • AssociatedObjectClassQualifierName (optional, repeating): a name or names that qualify the associated object class. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. • AssociatedObjectClassName (Mandatory): The name of the associated object class. • BusinessTermName (optional, repeating): A synonym term in which the ASBIE is commonly known. 	
[R 9D87]	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.	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 • ccts:ConditionType. 	1
[R 8E0D]	Each BDT XML Schema File MUST include (xsd:include) all BCL XML Schema Files and BIS XML Schema Files that are defined in the same namespace.	1
[R B4C0]	Each BDT XML Schema File MUST import (xsd:import) the XBT XML Schema File, and each CCL XML Schema File and CIS XML Schema File that is used by BDTs contained within the BDT XML Schema File.	1
[R AE00]	Each BDT 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 for that namespace.	1

Rule Number	Rule Description	Category
[R A7B8]	<p>The name of a BDT MUST be the:</p> <ul style="list-style-type: none"> • BDT <code>ccts:DataTypeQualifierTerm(s)</code> if any, plus. • The <code>ccts:DataTypeTerm</code>, plus. • The word <code>Type</code>, plus. • The underscore character [<code>_</code>], plus. • A six character unique identifier, unique within the given namespace, consisting of lowercase alphabetic characters [a-z], uppercase alphabetic characters [A-Z], and digit characters [0-9]. <p>All separators are removed and approved abbreviations and acronyms are applied.</p>	1
[R 8437]	<p>The six character unique identifier used for the BDT Type name MUST be unique within the namespace in which it is defined.</p>	1
[R B43E]	<p>When a BDT for Date, Time, and DateTime needs to support variable precision beyond what is possible within the XML Schema types, the BDT MUST use a <code>formatCode</code> attribute to indicate the format of the content, if and only if the format is not the default.</p>	1
[R 9B37]	<p>All <code>formatCode</code> attributes for the BDTs Date, Time or DateTime MUST define the formats allowed for the BDT.</p>	1
[R 9908]	<p>Every BDT devoid of <code>ccts:supplementaryComponents</code>, or whose <code>ccts:supplementaryComponents</code> BVD facets map directly to the facets of an XML Schema built-in data type, MUST be defined as a named <code>xsd:simpleType</code>.</p>	1
[R B91F]	<p>The <code>xsd:simpleType</code> definition of a BDT whose content component BVD is defined by a primitive whose facets map directly to the facets of an XML Schema built-in datatype MUST contain an <code>xsd:restriction</code> element with the <code>base</code> attribute set to the XML Schema built-in data type that represents the primitive.</p>	1
[R AA60]	<p>The <code>xsd:simpleType</code> definition of a BDT whose content component BVD is defined as a single code list MUST contain an <code>xsd:restriction</code> element with the <code>base</code> attribute set to the code list's defined <code>xsd:simpleType</code>.</p>	1

Rule Number	Rule Description	Category
[R A861]	The xsd:simpleType definition of a BDT whose content component BVD is defined by an identifier scheme MUST contain an xsd:restriction element with the base attribute set to the identifier scheme's defined xsd:simpleType .	1
[R AB05]	Every BDT that includes one or more Supplementary Components that do not map directly to the facets of an XSD built-in datatype MUST be defined as an xsd:complexType .	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 the Supplementary Component Property Term Name and Representation Term Name with periods, spaces, and other separators removed.	1
[R BBCB]	The xsd:complexType definition of a BDT whose Content Component BVD is defined by a primitive whose facets do not map directly to the facets of an XML Schema built-in datatype MUST contain an xsd:simpleContent element that contains an xsd:extension whose base attribute is set to the XML Schema built-in data type that represents the primitive.	1
[R BD8E]	The xsd:complexType definition of a BDT whose Content Component BVD is defined as a single code list MUST contain an xsd:simpleContent element that contains an xsd:extension element whose base attribute is set to the defined xsd:simpleType for the code list.	1
[R 91E8]	The xsd:complexType definition of a BDT whose Content Component BVD is defined by an identifier scheme MUST contain an xsd:simpleContent element that contains an xsd:extension whose base attribute is set to the identifier scheme's defined xsd:simpleType .	1
[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

Rule Number	Rule Description	Category
[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 or the formatCode attribute for Date , Type , Date Time Type , and Time Type .	1
[R BFE5]	<p>Every BDT XML Schema type definition MUST contain a structured set of xsd:annotation xsd:documentation elements that contain:</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. • DataTypeTermName (mandatory): The name of the DataType. The possible values for the DataType are defined in the Data Type Catalogue. • DataTypeQualifierTerm Name (optional, repeating): Is a word or words which help define and differentiate a Data Type. It further enhances the semantic meaning of the DataType. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. 	1

Rule Number	Rule Description	Category
[R 8095]	<p>Every BDT <code>xsd:simpleContent</code> element MUST contain a structured set of <code>ContentComponentValueDomain</code> <code>xsd:annotation</code> <code>xsd:documentation</code> elements that contain:</p> <ul style="list-style-type: none"> • <code>Definition</code> (mandatory): The semantic meaning of the BDT. • <code>DefaultIndicator</code> (mandatory): Indicates if the primitive, scheme or list is the default BVD for the data type. • <code>PrimitiveTypeName</code> (optional): The primitive type of the BDT Content Component. One of <code>PrimitiveTypeName</code>, or <code>SchemeOrListID</code> must be present. • <code>SchemeOrListID</code> (optional): The unique identifier assigned to the scheme or list that uniquely identifies it. One of <code>PrimitiveTypeName</code> or <code>SchemeOrListID</code> must be present. • <code>SchemeOrListVersionID</code>: The version of the scheme or list. Must be present if <code>SchemeOrListID</code> is present. • <code>SchemeOrListAgencyID</code> (optional): The unique identifier assigned to the Agency that owns or is responsible for the Scheme or Code List being referenced. Must be present if <code>SchemeOrListID</code> is present. • <code>SchemeOrListModificationAllowedIndicator</code> (optional): Indicates whether the Identifier Scheme or Code List can be modified. • <code>DefaultValue</code> (optional): The default value for the BDT Content Component. 	1

Rule Number	Rule Description	Category
[R 9C95]	<p>Every BDT Supplementary Component xsd:attribute declaration MUST contain a structured set of xsd:annotation xsd:documentation elements that contain:</p> <ul style="list-style-type: none"> • Cardinality (mandatory): Indicates the cardinality of the SC within the containing BDT. • DictionaryEntryName (mandatory): The Data Dictionary Entry Name (DEN) of the BDT SC. • Definition (mandatory): The semantic meaning of the BDT SC. • 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. • DataTypeTermName (mandatory): The name of the DataType Term. The possible values for the DataType Term are defined in the Data Type Catalogue. • DataTypeQualifierTermName (mandatory): A word or words which help define and differentiate a Data Type. It further enhances the semantic meaning of the DataType. The order in which the qualifiers are expressed indicate the order to be used, where the first one is to be the first order qualifier. 	1

Rule Number	Rule Description	Category
[R 91C3]	<p>Every Supplementary Component xsd:attribute declaration MUST contain within the structured set of xsd:annotation xsd:documentation elements containing SupplementaryComponentValueDomain element that contains:</p> <ul style="list-style-type: none"> • DefaultIndicator (mandatory): Indicates if the primitive, scheme or list is the default BVD for the data type. • PrimitiveTypeName (optional): The primitive type of the BDT Supplementary Component. One of PrimitiveTypeName or SchemeOrListID must be present. • SchemeOrListID (optional): The unique identifier assigned to the scheme or list that uniquely identifies it. One of PrimitiveTypeName or SchemeOrListID must be present. • SchemeOrListVersionID (optional): The version of the scheme or list. Must be present if SchemeOrListID is present. • SchemeOrListAgencyID (optional): The unique identifier assigned to the Agency that owns or is responsible for the Scheme or Code List being referenced. Must be present if SchemeOrListID is present. • SchemeOrListModificationAllowedIndicator (optional): Indicates whether the Identifier Scheme or Code List can be modified. • DefaultValue (optional): Is the default value. 	1
[R 8866]	The XML Schema Built-in Type Extension XML Schema File (XBT) MUST be defined in the data common namespace.	1
[R 9E40]	Each code list used by a BDT or BBIE MUST be defined in its own XML Schema File.	2
[R 89D1]	Agencies that do not have an Agency Identifier assigned by UN/CEFACT MUST use the Agency Name in CamelCase as the Agency Identifier.	1
[R AD5F]	Agencies that do not have a Scheme or List Identifier assigned MUST use the Scheme or List Name in CamelCase as the SchemeOrList Identifier.	1

Rule Number	Rule Description	Category
[R 849E]	<p>Code List XML Schema File names MUST be of the form: <List Agency Identifier>_<List Identifier>_<List 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"> • List 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. • List Identifier – Identifies a list of the respective corresponding ids. • List 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.	1
[R B5EC]	The Code List XML Schema File global element name MUST be the formal name of the code list with the word Code appended if not present in the code list name.	
[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 with the word code appended if it is not part of the code list name, and with the word ContentType appended.	1
[R 962C]	Each code in a Code List XML Schema File MUST be expressed as an xsd:enumeration , where the xsd:value for the enumeration is the actual code value.	1

Rule Number	Rule Description	Category
[R A142]	<p>Every Code List MUST contain a structured set of xsd:annotation xsd:documentation elements that contain:</p> <ul style="list-style-type: none"> • SchemeOrListID (mandatory): The unique identifier assigned to the code list. • SchemeOrListAgencyID (mandatory): The unique identifier assigned to the Agency that owns or is responsible for the code list being referenced. • SchemeOrListVersionID (mandatory): The version of the scheme or list. • SchemeOrListModificationAllowedIndicator (mandatory): Indicates whether the values being validated can be outside the enumerations specified by the code list. 	1
[R A814]	<p>Each code list xsd:enumeration MUST contain a structured set of xsd:annotation xsd:documentation elements that contain:</p> <ul style="list-style-type: none"> • Name (mandatory): The name of the code. • Description (optional): Descriptive information concerning the code. 	1

Rule Number	Rule Description	Category				
[R 992A]	<p>Code list XML Schema File namespaces MUST use the following pattern:</p> <table border="1" data-bbox="425 432 1292 764"> <tr> <td data-bbox="425 432 537 600">URN:</td> <td data-bbox="537 432 1292 600">urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:codelist:common:<major>:<status>:<name></td> </tr> <tr> <td data-bbox="425 600 537 764">URL:</td> <td data-bbox="537 600 1292 764">http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/codelist/common/<major>/<status>/<name></td> </tr> </table> <p>Where:</p> <ul style="list-style-type: none"> • organization – Identifier of the organization providing the standard. • organization hierarchy – The first level of the hierarchy within the organization providing the standard. • organization 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. <p>Code list names are further defined as: <Code List Agency Identifier><divider><Code List Identifier></p> <p>Where:</p> <ul style="list-style-type: none"> ▪ Code List Agency Identifier – The identifier for the agency that the code list is from. ▪ Divider – The divider character for URN is ‘:’ the divider character for URL is ‘/’. ▪ Code List Identifier – The identifier for the given code list. 	URN:	urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:codelist:common:<major>:<status>:<name>	URL:	http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/codelist/common/<major>/<status>/<name>	3
URN:	urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:codelist:common:<major>:<status>:<name>					
URL:	http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/codelist/common/<major>/<status>/<name>					

Rule Number	Rule Description	Category
[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 Identifier><Code List Version Identifier></p> <p>Such that any repeated words are eliminated.</p> <p>Where:</p> <ul style="list-style-type: none"> • Code List Agency Identifier – The identifier for the agency that the code list is from. • Code List Identifier – The identifier for the given code list. • Code List Version Identifier – The identifier for the version of the given code list. 	2
[R 86C8]	CCL XML Schema Files MUST NOT import or include any other XML Schema Files.	1
[R B40B]	Each CCL XML Schema File xsd:simpleType MUST use an xsd:restriction element whose base attribute is xsd:token and contains one xsd:enumeration element for each value expressed in the code list.	1
[R 8F2D]	<p>BCL XML Schema file MUST be used to</p> <ul style="list-style-type: none"> • Define a codelist where one does not exist or • Restrict the value of an existing code list or • Combine several individual code lists using xsd:union. 	1
[R 87A9]	BCL XML Schema Files MUST import only CCL XML Schema Files it uses directly.	1
[R 8104]	The xsd:simpleType definition for each BCL XML Schema File that defines a new code list MUST use an xsd:restriction element whose base attribute is xsd:token which contains one xsd:enumeration element for each value expressed for the code list.	1
[R 882D]	The xsd:simpleType definition for each BCL XML Schema File that restricts an existing code list MUST use an xsd:restriction element whose base attribute is the xsd:simpleType of the code list being restricted which contains one xsd:enumeration element for each value expressed in the restricted code list.	1

Rule Number	Rule Description	Category
[R 9A22]	The xsd:simpleType definition for each BCL XML Schema File that combines the values of multiple code lists MUST use an xsd:union element whose memberTypes attribute contains the xsd:simpleTypes of the code lists being unioned together.	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: <Scheme Agency Identifier>_<Scheme Identifier>_<Scheme 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"> • Scheme Agency Identifier – Identifies the agency that manages the identifier scheme. The default agency IDs used are those from DE 3055, however, roles defined in DE 3055 cannot be used. • Scheme Identifier – Identifies the identifier scheme. • 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.	1
[R 9B48]	The Identifier Scheme XML Schema File global element name MUST be the formal name of the Identifier Scheme with the word identifier appended if not present in the Identifier Scheme name.	1
[R 9451]	Each Identifier Scheme XML Schema File MUST define one, and only one, named xsd:simpleType for the content component.	1
[R B79A]	The Identifier Scheme XML Schema File xsd:simpleType name MUST be the name of the identifier scheme with the word Identifier appended if not part of the identifier scheme name and the word ContentType appended.	1

Rule Number	Rule Description	Category
[R B30A]	<p>Every Identifier Scheme MUST contain a structured set of xsd:annotation xsd:documentation elements that contain:</p> <ul style="list-style-type: none">• SchemeOrListID (mandatory): The unique identifier assigned to the Identifier Scheme.• SchemeOrListVersionID (mandatory): Identifies the version of the scheme.• SchemeOrListAgencyID (mandatory): The unique identifier assigned to the Agency that owns or is responsible for the identifier scheme being referenced.• SchemeOrListModificationAllowedIndicator (mandatory): Indicates whether the values being validated can be outside the pattern specified by the scheme.	1

Rule Number	Rule Description	Category				
[R 9CCF]	<p>Identifier scheme XML Schema File namespaces MUST use the following pattern:</p> <table border="1" data-bbox="427 432 1292 764"> <tr> <td data-bbox="427 432 537 600">URN:</td> <td data-bbox="540 432 1292 600"><code>urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:identifierscheme:common:<major>:<status>:<name></code></td> </tr> <tr> <td data-bbox="427 604 537 764">URL:</td> <td data-bbox="540 604 1292 764"><code>http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/identifierscheme/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. • 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 <i>XML Schema File</i> removed. <ul style="list-style-type: none"> ○ Identifier scheme names are further defined as: <code><Identifier Scheme Agency Identifier></code> <code><divider><Identifier Scheme Identifier></code> <p>Where:</p> <ul style="list-style-type: none"> ▪ Identifier Scheme Agency Identifier – The identifier for the agency that identifier scheme is from. ▪ Divider – The divider character for URN is : the divider character for URL is /. ▪ Identifier Scheme Identifier – The identifier for the given identifier scheme. 	URN:	<code>urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:identifierscheme:common:<major>:<status>:<name></code>	URL:	<code>http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/identifierscheme/common/<major>/<status>/<name></code>	1
URN:	<code>urn:<organization>:<organization hierarchy>[:<organization hierarchy level>]*:identifierscheme:common:<major>:<status>:<name></code>					
URL:	<code>http://<organization>/<organization hierarchy>[/<organization hierarchy level>]*/identifierscheme/common/<major>/<status>/<name></code>					

Rule Number	Rule Description	Category
[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 Identifier><Identifier Scheme Version Identifier></pre> <p>Such that any repeated words are eliminated.</p> <p>Where:</p> <ul style="list-style-type: none"> • Identifier Scheme Agency Identifier – The identifier for the agency that the identifier scheme is from. • Identifier Scheme Identifier – The identifier for the given identifier scheme. • Identifier Scheme Version Identifier – The version identifier 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 <code>base</code> attribute value = <code>xsd:token</code> .	1
[R A1E3]	<p>BIS XML Schema file MUST be used to:</p> <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 <code>base</code> 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

Rule Number	Rule Description	Category
[R 9277]	The xsi:nil attribute MUST NOT appear in any conforming instance.	3
[R B4D1]	If used by other than UN/CEFACT organizations, the xsi:nil attribute MUST only be used to signal the intentional removal of a previously communicated value.	1
[R 8250]	The xsi:type 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

5398

5399
5400**K.1 Naming and Design Rules for the Alternative Business Message Syntax in Appendix I**

Rule Number	Rule Description	Category
[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 (xsd:unique) constraint MUST be used rather than the key (xsd:key) constraint to define the keys and enforce that their values are unique within their scope of application.	1
[R 8EE7]	Identifiers used in schema identity 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
[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 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
[R AC2D]	For each referenced element in a given scope one xsd:keyref constraint involving the reference attribute that point to the referenced element MUST be declared in the XML Schema, under the scope element.	1

Rule Number	Rule Description	Category
[R 9BE8]	The xsd:keyref/xsd:selector/@xpath 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 xsd:keyref 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
[R 886A]	Uniqueness of @key attributes that are not involved in structural referencing MUST NOT be enforced by the schema via identity constraints. Uniqueness of @key attributes should be assured by use of adequate algorithms for the generation of the identifiers (e.g. UUIDs).	1
[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 cts: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

Rule Number	Rule Description	Category
[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

5401

5402 **Appendix L. Glossary**

- 5403 **Aggregate Business Information Entity (ABIE)** – A collection of related pieces of
5404 business information that together convey a distinct business meaning in a specific
5405 business context. Expressed in modelling terms, it is the representation of an object
5406 class, in a specific business context.
- 5407 **Aggregate Core Component (ACC)** – A collection of related pieces of business
5408 information that together convey a distinct business meaning, independent of any
5409 specific business context. Expressed in modelling terms, it is the representation of
5410 an object class, independent of any specific business context.
- 5411 **Aggregation** – An Aggregation is a special form of Association that specifies a
5412 whole-part relationship between the aggregate (whole) and a component part.
- 5413 **Artefact** – A piece of information that is produced, modified, or used by a process.
5414 An artefact can be a model, a model element, or a document. A document can
5415 include other documents. CCTS artefacts include all registry classes as specified in
5416 Section 9 of the *CCTS Technical Specification* and all subordinate named constructs
5417 of a CCTS registry class.
- 5418 **Assembly Rules** – Assembly Rules group sets of unrefined business information
5419 entities into larger artefacts suitable for expressing complete business information
5420 exchange concepts.
- 5421 **Association Business Information Entity (ASBIE)** – A business information entity
5422 that represents a complex business characteristic of a specific object class in a
5423 specific business context. It has a unique business semantic definition. An
5424 Association Business Information Entity represents an Association Business
5425 Information Entity property and is therefore associated to an Aggregate Business
5426 Information Entity, which describes its structure. An Association Business
5427 Information Entity is derived from an Association Core Component.
- 5428 **Association Business Information Entity Property** – A business information entity
5429 property for which the permissible values are expressed as a complex structure,
5430 represented by an Aggregate Business Information Entity.
- 5431 **Association Core Component (ASCC)** – A core component which constitutes a
5432 complex business characteristic of a specific Aggregate Core Component that
5433 represents an object class. It has a unique business semantic definition. An
5434 Association Core Component represents an Association Core Component Property
5435 and is associated to an Aggregate Core Component, which describes its structure.
- 5436 **Association Core Component Property** – A core component property for which the
5437 permissible values are expressed as a complex structure, represented by an
5438 Aggregate Core Component.
- 5439 **Attribute** – A named value or relationship that exists for some or all instances of
5440 some entity and is directly associated with that instance.
- 5441 **Backward Compatibility** – Any XML instance that is valid against one schema
5442 version will also validate against the previous schema version.
- 5443 **Basic Business Information Entity (BBIE)** – A business information entity that
5444 represents a singular business characteristic of a specific object class in a specific

- 5445 business context. It has a unique business semantic definition. A Basic Business
5446 Information Entity represents a Basic Business Information Entity property and is
5447 therefore linked to a data type, which describes its values. A Basic Business
5448 Information Entity is derived from a Basic Core Component.
- 5449 **Basic Business Information Entity Property** – A business information entity
5450 property for which the permissible values are expressed by simple values,
5451 represented by a data type.
- 5452 **Basic Core Component (BCC)** – A core component which constitutes a singular
5453 business characteristic of a specific Aggregate Core component that represents a
5454 object class. It has a unique business semantic definition. A Basic Core Component
5455 represents a Basic Core Component property and is therefore of a data type, which
5456 defines its set of values. Basic core components function as the properties of
5457 Aggregate Core components.
- 5458 **Basic Core Component (BCC) Property** – A core component property for which
5459 the permissible values are expressed by simple values, represented by a data type.
- 5460 **Business Context** – The formal description of a specific business circumstance as
5461 identified by the values of a set of context categories, allowing different business
5462 circumstances to be uniquely distinguished.
- 5463 **Business Data Type** – A business data type is a data type, which consists of one
5464 and only one BDT content component, that carries the actual content plus one or
5465 more BDT supplementary components giving essential extra definition to the BDT
5466 content component.
- 5467 **Business Data Type Content Component** – Defines the primitive type or scheme
5468 or list used to express the content of a business data type.
- 5469 **Business Data Type Content Component Restriction** – The formal definition of a
5470 format restriction that applies to the possible values of a business data type content
5471 component.
- 5472 **Business Data Type Supplementary Component** – Gives additional meaning to
5473 the business data type content component.
- 5474 **Business Data Type Supplementary Component Restrictions** – The formal
5475 definition of a format restriction that applies to the possible values of a business data
5476 type supplementary component.
- 5477 **Business Information Entity (BIE)** – A piece of business data or a group of pieces
5478 of business data with a unique business semantic definition. A business information
5479 entity can be a Basic Business Information Entity (BBIE), an Association Business
5480 Information Entity (ASBIE), or an Aggregate Business Information Entity (ABIE).
- 5481 **Business Information Entity (BIE) Property** – A business characteristic belonging
5482 to the Object Class in its specific business context that is represented by an
5483 Aggregate Business Information Entity.
- 5484 **Business Libraries** – A collection of approved process models specific to a line of
5485 business (e.g., shipping, insurance).
- 5486 **Business Process** – The business process as described using the UN/CEFACT
5487 Catalogue of Common business processes.

- 5488 **Business Process Context** – The business process name(s) as described using
5489 the *UN/CEFACT Catalogue of Common Business Processes* as extended by the
5490 user.
- 5491 **Business Process Role Context** – The actors conducting a particular business
5492 process, as identified in the *UN/CEFACT Catalogue of Common Business*
5493 *Processes*.
- 5494 **Business Semantic(s)** – A precise meaning of words from a business perspective.
- 5495 **Business Term** – This is a synonym of the dictionary entry name under which the
5496 artefact is commonly known and used in business. A CCTS artefact may have
5497 several business terms or synonyms.
- 5498 **Business Value Domain** – The set of allowed values.
- 5499 **Cardinality** – An indication of the minimum and maximum occurrences for a
5500 characteristic: not applicable (0..0), optional (0..1), optional repetitive (0..*)
5501 mandatory (1..1), mandatory repetitive (1..*), fixed (n..n) where n is a non-zero
5502 positive integer.
- 5503 **Catalogue of Business Information Entities** – This represents the approved set of
5504 Business Information Entities from which to choose when applying the Core
5505 Component discovery process
- 5506 **Classification Scheme** – This is an officially supported scheme to describe a given
5507 context category.
- 5508 **Composite** – A form of aggregation which requires that a part instance be included
5509 in at most one composite at a time, and that the composite object is responsible for
5510 the creation and destruction of the parts. Composite may be recursive.
- 5511 **Context** – Defines the circumstances in which a business process may be used.
5512 This is specified by a set of context categories known as business context.
- 5513 **Context Category** – A group of one or more related values used to express a
5514 characteristic of a business circumstance.
- 5515 **Controlled Vocabulary** – A supplemental vocabulary used to uniquely define
5516 potentially ambiguous words or business terms. This ensures that every word within
5517 any of the core component names and definitions is used consistently,
5518 unambiguously and accurately.
- 5519 **Core Component (CC)** – A building block for the creation of a semantically correct
5520 and meaningful information exchange package. It contains only the information
5521 pieces necessary to describe a specific concept.
- 5522 **Core Component Library (CCL)** – The Core Component Library is the part of the
5523 registry/repository in which Core Components shall be stored as registry classes.
5524 The Core Component Library will contain all the registry classes.
- 5525 **Core Component Property** – A business characteristic belonging to the object class
5526 represented by an Basic Core Component property or an Association Core
5527 Component property.
- 5528 **Core Data Type (CDT)** – The Core Data Type is the data type that constitutes the
5529 value space for the allowed values for a property.

- 5530 **Data Package** – Is a division of content based upon the requirements of a
5531 conformation CCTS model. Data packages may be hierarchical. Data packages may
5532 share information.
- 5533 **Definition** – This is the unique semantic meaning of a core component, business
5534 information entity, business context or data type.
- 5535 **Dictionary Entry Name** – This is the official name of a CCTS-conformant artefact.
- 5536 **Facet** – A facet is a constraining value that represents a component restriction of a
5537 Business Data Type content or supplementary component so as to define its allowed
5538 value space.
- 5539 **Geopolitical Context** – Geographic factors that influence business semantics (e.g.,
5540 the structure of an address).
- 5541 **Industry Classification Context** – Semantic influences related to the industry or
5542 industries of the trading partners (e.g., product identification schemes used in
5543 different industries).
- 5544 **Information Entity** – A reusable semantic building block for the exchange of
5545 business-related information.
- 5546 **LowerCamelCase (LCC)** – LowerCamelCase is a lexical representation of
5547 compound words or phrases in which the words are joined without spaces and all but
5548 the first word are capitalized within the resulting compound.
- 5549 **Message Assembly** – The process whereby Business Information Entities are
5550 assembled into a usable message for exchanging business information.
- 5551 **Naming Convention** – The set of rules that together comprise how the dictionary
5552 entry name for CCTS artefacts are constructed.
- 5553 **Object Class** – The logical data grouping (in a logical data model) to which a data
5554 element belongs (ISO11179). The object class is the part of a core component or
5555 business information entity dictionary entry name that represents an activity or
5556 object.
- 5557 **Object Class Term** – A component of the name of a core component or business
5558 information entity which represents the object class to which it belongs.
- 5559 **Official Constraints Context** – Legal and governmental influences on semantics
5560 (e.g. hazardous materials information required by law when shipping goods).
- 5561 **Primitive Type** – A primitive type, also known as a base type or built-in type, is the
5562 basic building block for the representation of a value as expressed by more complex
5563 data types.
- 5564 **Product Classification Context** – Factors influencing semantics that are the result
5565 of the goods or services being exchanged, handled, or paid for, etc. (e.g. the buying
5566 of consulting services as opposed to materials).
- 5567 **Property Term** – A semantically meaningful name for the characteristic of the Object
5568 Class that is represented by the property.
- 5569 **Qualified Business Data Type** – A qualified business data type contains restrictions
5570 on a business data type content or business data type supplementary component(s).

- 5571 **Qualifier Term** – A word or group of words that help define and differentiate an item
5572 (e.g. a business information entity or a business data type) from its associated items
5573 (e.g. from a core component, a core data type, another business information entity or
5574 another business data type).
- 5575 **Registry** – An information system that manages and references artefacts that are
5576 stored in a repository. The term registry implies a combination of registry/repository.
- 5577 **Registry Class** – The formal definition of all the common information necessary to
5578 be recorded in the registry by a registry artefact – core component, a business
5579 information entity, a data type or a business context.
- 5580 **Repository** – an information system that stores artefacts.
- 5581 **Representation Term** – A semantic expression of the value domain.
- 5582 **Scope element** – (for identity constraints) – The element whose schema declaration
5583 contains the identity constraints.
- 5584 **Supporting Role Context** – Semantic influences related to non-partner roles (e.g.,
5585 data required by a third-party shipper in an order response going from seller to
5586 buyer.).
- 5587 **Syntax Binding** – The process of expressing a Business Information Entity in a
5588 specific syntax.
- 5589 **System Capabilities Context** – This context category exists to capture the
5590 limitations of systems (e.g. an existing back office can only support an address in a
5591 certain form).
- 5592 **UMM Information Entity** – A UN/CEFACT Modelling Methodology (UMM)
5593 information entity realizes structured business information that is exchanged by
5594 partner roles performing activities in a business transaction. Information entities
5595 include or reference other information entities through associations.”
- 5596 **Unique Identifier** – The identifier that references a registry class instance in a
5597 universally unique and unambiguous way.
- 5598 **UpperCamelCase (UCC)** – UpperCamelCase is a lexical representation of
5599 compound words or phrases in which the words are joined without spaces and are
5600 capitalized within the resulting compound.
- 5601 **Usage Rules** – Usage rules describe a constraint that describes specific conditions
5602 that are applicable to a component in the model.
- 5603 **User Community** – A user community is a group of practitioners, with a publicized
5604 contact address, who may define Context profiles relevant to their area of business.
5605 Users within the community do not create, define or manage their individual context
5606 needs but conform to the community’s standard. Such a community should liaise
5607 closely with other communities and with general standards-making bodies to avoid
5608 overlapping work. A community may be as small as two consenting organizations.
- 5609 **Version** – An indication of the evolution over time of an instance of a core
5610 component, data type, business context, or business information entity.
- 5611 **XML Schema** – A generic term used to identify the family of grammar based XML
5612 document structure validation languages to include the more formal W3C XML
5613 Schema Definition Language, ISO 8601 Document Type Definition, or Schematron.
5614 An XML Schema is a collection of schema components.

- 5615 **XML Schema Definition Language Component** –The 13 building blocks that
5616 comprise the abstract data model of the schema, consisting of simple type
5617 definitions, complex type definitions, attribute declarations, element declarations,
5618 attribute group definitions, identity-constraint definitions, model group definitions,
5619 notation declarations, annotations, model groups, particles, wildcards, and attribute
5620 uses.
- 5621 **XML Schema Definition Language** – The World Wide Web Consortiums official
5622 recommendation for describing the structure and constraining the contents of XML
5623 documents.
- 5624 **XML Schema Document** – An XML conformant document expression of an XML
5625 schema.

5626 Disclaimer

5627 The views and specification expressed in this document are those of the authors and
5628 are not necessarily those of their employers. The authors and their employers
5629 specifically disclaim responsibility for any problems arising from correct or incorrect
5630 implementation or use of this design.

5631 Copyright Statement

5632

5633 Copyright © UN/CEFACT 2009. All Rights Reserved.

5634

5635 This document and translations of it may be copied and furnished to others, and
5636 derivative works that comment on or otherwise explain it or assist in its
5637 implementation may be prepared, copied, published and distributed, in whole or in
5638 part, without restriction of any kind, provided that the above copyright notice and this
5639 paragraph are included on all such copies and derivative works. However, this
5640 document itself may not be modified in any way, such as by removing the copyright
5641 notice or references to UN/CEFACT except as required to translate it into languages
5642 other than English.

5643 The limited permissions granted above are perpetual and will not be revoked by
5644 UN/CEFACT or its successors or assigns.

5645 This document and the information contained herein is provided on an "AS IS" basis
5646 and UN/CEFACT DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED,
5647 INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE
5648 INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED
5649 WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR
5650 PURPOSE.