UN/CEFACT/UBL XML Naming and Design Rules Analysis

03 August 2007 [Source: http://www.oasis-open.org/committees/download.php/25133/UBL_vs_CEFACT_XML_NDR_Analysis_2007-08-03.doc]

Legend

🟩	Same as UBL
🟩	No corresponding rule in UBL, Concur (pending acceptance by the TC)
🟨	Different than UBL: wording
🟥	No corresponding rule in UBL : needs discussion OR Different than UBL: intent (will include UBL Rule text)
🟪	Code Lists and Versioning

General Comments:

1. Need to be able to distinguish between those rules that are UN/CEFACT specific (e.g. creation of CCT and UDT schemas, namespace names) and those that are meant to be of wider applicability.

Rule	UN/CEFACT XML Naming and Design Rules Version 2.0, 17 February 2006	UBL Rule	UBL XML Design Rules	UBL Disposition
[R 1]	Conformance shall be determined through adherence to the content of normative sections, rules and definitions.			This is not a naming and design rule. Concur (pending acceptance by the TC)
[R 2]	All UN/CEFACT XSD Schema design rules MUST be based on the W3C XML Schema Recommendations: XML Schema Part 1: Structures and XML Schema Part 2: Data Types.			This is not a naming and design rule. This is an instruction to the *authors* of the NDR. Concur (pending acceptance by the TC)
[R 3]	All UN/CEFACT XSD Schema and UN/CEFACT conformant XML instance documents MUST be based on the W3C suite of technical specifications holding recommendation status.			Concur (pending acceptance by the TC)
[R 4]	UN/CEFACT XSD Schema MUST follow the standard structure defined in Appendix B.	GXS1	(See NDR – too long to put here.)	Concur (pending acceptance by the TC)
[R 5]	Each element or attribute XML name MUST have one and only one fully qualified XPath (FQXP).	NMC1	Each dictionary entry name MUST define one and only one fully qualified path (FQP) for an element or attribute.	
[R 6]	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.	GNR1	UBL XML element and type names MUST be in the English language, using the primary English spellings provided in the Oxford English Dictionary.	
[R 7]	Lower camel case (LCC) MUST be used for naming			We have not defined any attributes.

	attributes.			Concur (pending acceptance by the TC)
[R 8]	Upper camel case (UCC) MUST be used for naming elements and types.	GNR8	The UpperCamelCase (UCC) convention MUST be used for naming elements and types.	
[R 9]	Element, attribute and type names MUST be in singular form unless the concept itself is plural.	GNR7	UBL XML element, and type names MUST be in singular form unless the concept itself is plural.	
[R 10]	Element, attribute and type names MUST be drawn from the following character set: a-z and A-Z.			Concur (pending acceptance by the TC)
[R 11]	XML element, attribute and type names constructed from dictionary entry names MUST NOT include periods, spaces, or other separators; or characters not allowed by W3C XML 1.0 for XML names.	GNR3	UBL XML element and type names constructed from ccts:DictionaryEntryNames MUST NOT include periods, spaces, other separators, or characters not allowed by W3C XML 1.0 for XML names	
[R 12]	XML element, attribute and type names MUST NOT use acronyms, abbreviations, or other word truncations, except those included in the UN/CEFACT controlled vocabulary or listed in Appendix C.	GNR4	UBL XML element, and simple and complex type names MUST NOT use acronyms, abbreviations, or other word truncations, except those in the list of exceptions maintained and published by the UBL TC.	
[R 13]	The acronyms and abbreviations listed in Appendix C MUST always be used.	GNR6	The acronyms and abbreviations listed in the UBL-approved list MUST always be used in place of the word or phrase they represent.	Just a technicality, but makes intent clearer.
[R 14]	Acronyms and abbreviations at the beginning of an attribute declaration MUST appear in all lower case. All other acronym and abbreviation usage in an attribute declaration must appear in upper case.	GNR10	Acronyms and abbreviations at the beginning of an attribute name MUST appear in all lower case. All other acronym and abbreviation usage in an attribute declaration MUST appear in upper case.	
[R 15]	Acronyms MUST appear in all upper case for all element declarations and type definitions.	GNR11	Acronyms and abbreviations MUST appear in all upper case for all element declarations and type definitions.	
[R 16]	The schema module file name for modules other than code lists or identifier lists MUST of the form <SchemaModuleName>_<Version>.xsd, with periods, spaces, or other separators and the words Schema Module removed.	SSM7	Each UBL internal schema module MUST be named {ParentSchemaModuleName}{InternalSchemaModuleFunction }{schema module}	Main issue is the inclusion of the version number.
[R 17]	The schema module file name for code lists and identifier lists, MUST be of the form <AgencyName>_<ListName>_<Version>.xsd, with periods, spaces, or other separators removed.			The only code list schema modules we use are the ones imported by the UN/CEFACT UDT schema module.
[R 18]	In representing versioning schemes in file names, the period MUST be represented by a lowercase p.			See [R 16]
[R 19]	A root schema MUST be created for each unique business information payload.			Concur (pending acceptance by the TC)
[R 20]	Each UN/CEFACT root schema module MUST be named <BusinessInformationPayload> Schema Module	SSM7	Each UBL internal schema module MUST be named {ParentSchemaModuleName}{InternalSchemaModuleFunction }{schema module}	Concur (pending acceptance by the TC)

[R 21]	A root schema MUST NOT replicate reusable constructs available in schema modules capable of being referenced through xsd:include or xsd:import.			Concur (pending acceptance by the TC)
[R 22]	UN/CEFACT XSD schema modules MUST either be treated as external schema modules, or as internal schema modules of the root schema.	SSM5	UBL schema modules MUST either be treated as external schema modules or as internal schema modules of the document schema.	
[R 23]	All UN/CEFACT internal schema modules MUST be in the same namespace as their corresponding rsm:RootSchema.	SSM6	All UBL internal schema modules MUST be in the same namespace as their corresponding document schema.	
[R 24]	Each UN/CEFACT internal schema module MUST be named <ParentRootSchemaModuleName><InternalSchemaModuleFunction> Schema Module	SSM7	Each UBL internal schema module MUST be named {ParentSchemaModuleName}{InternalSchemaModuleFunction}{schema module}	
[R 25]	A Core Component Type schema module MUST be created.			See General Comment 1.
[R 26]	The cct:CoreComponentType schema module MUST be named 'Core Component Type Schema Module'.			See [R 25]
[R 27]	An Unqualified Data Type schema module MUST be created.			See [R 25]
[R 28]	The udt:UnqualifiedDataType schema module MUST be named 'Unqualified Data Type Schema Module'.			See [R 25]
[R 29]	A Qualified Data Type schema module MUST be created.	SSM18	A schema module defining all UBL Qualified Datatypes MUST be created.	
[R 30]	The qdt:QualifiedDataType schema module MUST be named 'Qualified Data Type Schema Module'.	SSM19	The UBL Qualified Datatypes schema module MUST be identified as QualifiedDataTypes in the document name in the schema header.	What, exactly, does "MUST be named" mean? I don't think you are referring to the file name.
[R 31]	A Reusable Aggregate Business Information Entity schema module MUST be created.	SSM9	A schema module defining all UBL Common Aggregate Components MUST be created.	Concur (pending acceptance by the TC)
[R 32]	The ram:ReusableAggregateBusinessInformationEntity schema module MUST be named 'Reusable Aggregate Business Information Entity Schema Module'.	SSM10	The UBL Common Aggregate Components schema module MUST be identified as CommonAggregateComponents in the document name within the schema header.	See [R 30]
[R 33]	Reusable Code List schema modules MUST be created to convey code list enumerations.			UBL supports OASIS Code List Representation TC - Genericode

[R 34]	The name of each clm:CodeList schema module MUST be of the form: <Code List Agency Identifier|Code List Agency Name><Code List Identification Identifier|Code List Name> - Code List Schema Module Where: Code List Agency Identifier = Identifies the agency that maintains the code list Code List Agency Name = Agency that maintains the code list Code List Identification Identifier = Identifies a list of the respective corresponding codes Code List Name = The name of the code list as assigned by the agency that maintains the code list			UBL supports OASIS Code List Representation TC - Genericode
[R 35]	An identifier list schema module MUST be created to convey enumerated values for each identifier list that requires runtime validation.			
[R 36]	The name of each ids:IdentifierList schema module MUST be of the form: <Identifier Scheme Agency Identifier|Identifier Scheme Agency Name><Identifier Scheme Identifier|Identifier Scheme Name> - Identifier List Schema Module Where: Identifier Scheme Agency Identifier = The identification of the agency that maintains the identifier list Identifier Scheme Agency Name = Agency that maintains the identifier list Identifier Scheme Identifier = The identification of the identifier list Identification Scheme Name = Name as assigned by the agency that maintains the identifier list			UBL supports OASIS Code List Representation TC - Genericode
[R 37]	Imported schema modules MUST be fully conformant with the UN/CEFACT XML Naming and Design Rules Technical Specification and the UN/CEFACT Core Components Technical Specification.			What if someone wants to take advantage of an existing spec (e.g. XLink: attributes named 'arcrole' and 'href')?
[R 38]	Every UN/CEFACT defined or imported schema module MUST have a namespace declared, using the xsd:targetNamespace attribute.	NMS1	Every UBL-defined –or -used schema module, except internal schema modules, MUST have a namespace declared using the xsd:targetNamespace attribute.	
[R 39]	Every version of a defined or imported schema module other than internal schema modules MUST have its own unique namespace.	NMS2	Every UBL-defined-or -used major version schema set MUST have its own unique namespace.	
[R 40]	UN/CEFACT published namespace declarations MUST NOT be changed, and its contents MUST NOT be changed unless such change does not break backward compatibility.	NMS6	UBL published namespaces MUST never be changed.	
[R 41]	UN/CEFACT namespaces MUST be defined as Uniform Resource Names.			Is this rule necessary considering the following 2 rules?

[R 42]	The names for namespaces MUST have the following structure while the schema is at draft status: urn:un:unece:uncefact:<schematype>:draft:<name>:<major> Where: schematype = a token identifying the type of schema module: data|process|codelist|identifierlist|documentation name = the name of the schema module (using upper camel case) 4457 with periods, spaces, or other separators and the words 'schema module' removed. major = the major version number. Sequentially assigned, first release starting with the number 1.	NMS4	The namespace names for UBL Schemas holding committee draft status MUST be of the form: urn:oasis:names:tc:ubl:schema:<subtype>:<document-id>	Concur (pending acceptance by the TC)
[R 43]	The namespace names for schema holding specification status MUST be of the form: urn:un:unece:uncefact:<schematype>:standard:<name>:<major>. Where: schematype = a token identifying the type of schema module: data|process|codelist|identifierlist|documentation name = the name of the schema module (using upper camel case) with periods, spaces, or other separators and the words 'schema module' removed. major = the major version number, sequentially assigned, first release starting with the number 1.	NMS5	The namespace names for UBL Schemas holding OASIS Standard status MUST be of the form: urn:oasis:names:specification:ubl:schema:<subtype>:<document-id>	Concur (pending acceptance by the TC)
[R 44]	UN/CEFACT namespace values will only be assigned to UN/CEFACT developed objects.	NMS3	UBL namespaces MUST only contain UBL developed schema modules.	Concur (pending acceptance by the TC)
[R 45]	The general structure for schema location MUST be: http://www.unece.org/uncefact/<schematype>/<status>/<name>_<major>.<minor>p [<revision>].xsd Where: schematype = a token identifying the type of schema module: data|process|codelist|identifierlist|documentation status = the status of the schema as: draft|standard name = the name of the schema module (using upper camel case) with periods, spaces, or other separators and the words 'schema module' removed. major = the major version number, sequentially assigned, first release starting with the number 1. minor = the minor version number within a major release, sequentially assigned, first release starting with the number 0. revision = sequentially assigned alphanumeric character for each revision of a minor release. Only applicable where status = draft.	GXS15	Each xsd:schemaLocation attribute declaration MUST contain a system-resolvable URL, which at the time of release from OASIS shall be a relative URL referencing the location of the schema or schema module in the release package.	What if the user is not connected to the internet (whether by choice or circumstance)?

[R 46]	Each xsd:schemaLocation attribute declaration MUST contain a persistent and resolvable URL.	GXS15	Each xsd:schemaLocation attribute declaration MUST contain a system-resolvable URL,…	
[R 47]	Each xsd:schemaLocation attribute declaration URL MUST contain an absolute path.	GXS15	Each xsd:schemaLocation attribute declaration MUST contain a system-resolvable URL, which at the time of release from OASIS shall be a relative URL referencing the location of the schema or schema module in the release package.	See [R 45]
[R 48]	The xsd:schema version attribute MUST always be declared.			Implied by other rules. Concur.
[R 49]	The xsd:schema version attribute MUST use the following template: <xsd:schema ... version=”<major>.<minor>”>			Dependent on our versioning strategy.
[R 50]	Every schema version namespace declaration MUST have the URI of: urn:un:unece:uncefact:<schematype>:<status>:<name>:<major>			Dependent on our versioning strategy.
[R 51]	Every UN/CEFACT XSD Schema and schema module major version number MUST be a sequentially assigned incremental integer greater than zero.	VER6	Every UBL Schema and schema module major version number MUST be a sequentially assigned, incremental number greater than zero.	
[R 52]	Minor versioning MUST be limited to declaring new optional XSD constructs, extending existing XSD constructs, or refinements of an optional nature.			Dependent on our versioning strategy.
[R 53]	For UN/CEFACT minor version changes, the name of the schema construct MUST NOT change.			Clarification needed. What schema construct? Is this necessary considering [R 52]?
[R 54]	Changes in minor versions MUST NOT break semantic compatibility with prior versions having the same major version number.	VER10	UBL Schema and schema module minor version changes MUST not break semantic compatibility with prior versions.	Concur
[R 55]	UN/CEFACT minor version schema MUST incorporate all XML constructs from the immediately preceding major or minor version schema.			Concur (pending acceptance by the TC)
[R 56]	The xsd:elementFormDefault attribute MUST be declared and its value set to qualified.	GXS1	… Attribute Declarations – elementFormDefault=””qualified”” attributeFormDefault=””unqualified”” …	
[R 57]	The xsd:attributeFormDefault attribute MUST be declared and its value set to unqualified.	GXS1	… Attribute Declarations – elementFormDefault=””qualified”” attributeFormDefault=””unqualified”” …	

[R 58]	The xsd prefix MUST be used in all cases when referring to http://www.w3.org/2001/XMLSchema as follows: xmlns:xsd=http://www.w3.org/2001/XMLSchema.	GXS4	All W3C XML Schema constructs in UBL Schema and schema modules MUST contain the following namespace declaration on the xsd schema element: xmlns:xsd="http://www.w3.org/2001/XMLSchema"	Proposal: The namespace prefix "xsd" MUST be used in all cases when referring to the namespace "http://www.w3.org/2001/XMLSchema": xmlns:xsd=http://www.w3.org/2001/XMLSchem a.
[R 59]	xsd:appInfo MUST NOT be used.	GXS12	UBL designed schema SHOULD NOT use xsd:appinfo. If used, xsd:appinfo MUST only be used to convey non-normative information.	Discuss.
[R 60]	xsd:notation MUST NOT be used.	GXS7	xsd:notation MUST NOT be used.	
[R 61]	xsd:wildcard MUST NOT be used.			Clarification needed. Although there are 'wildcard' schema components, I don't believe that the Schema spec identifies an element or attribute named "wildcard". Maybe "Wildcard schema components MUST NOT be used". Either way, we do allow them in our exrtension mechanism.
[R 62]	The xsd:any element MUST NOT be used.	GXS14	The xsd:any element MUST NOT be used except within the 'ExtensionContentType' type definition, and with xsd:processContents= "skip" for non-UBL namespaces.	Limited use in our localization/customization methodology.
[R 63]	The xsd:any attribute MUST NOT be used.	GXS17	The xsd:anyAttribute MUST NOT be used.	Is there an "xsd:any" attribute?
[R 64]	Mixed content MUST NOT be used (excluding documentation).	MDC2	Mixed content MUST NOT be used except where contained in an xsd:documentation element.	Concur.
[R 65]	xsd:substitutionGroup MUST NOT be used.	GXS5	The xsd:substitutionGroup feature MUST NOT be used.	
[R 66]	xsd:ID/xsd:IDREF MUST NOT be used.			Recommend 'SHOULD NOT'. If we want these rules to be generally useful, we should allow for this more basic construct that is more consistently implemented in the various parsers.
[R 67]	xsd:key/xsd:keyref MUST be used for information association.			Recommend 'SHOULD'.
[R 68]	The absence of a construct or data MUST NOT carry meaning.	IND6	The absence of a construct or data in a UBL instance document MUST NOT carry meaning.	
[R 69]	User declared attributes MUST only be used to convey core component type (CCT) supplementary component information.	NA	From NDR: "UBL, as a transactional based XML exchange format, has chosen to significantly restrict the use of attributes. This restriction is in keeping with the fact that attribute usage is relegated to supplementary components only; all "primary" business data appears exclusively in element content. These attributes are defined in the UN/CEFACT Unqualified Datatype schema module."	However, if these rules are hoped to be applied to something other than "transactional based XML exchange", we may want to change this to "SHOULD".

[R 70]	A xsd:attribute that represents a supplementary component with variable information MUST be based on the appropriate XSD built-in data type.			Probably could use some clarification though.
[R 71]	A xsd:attribute that represents a supplementary component which represents codes MUST be based on the xsd:simpleType of the appropriate code list.			See [R 33]
[R 72]	A xsd:attribute that represents a supplementary component which represents identifiers MUST be based on the xsd:simpleType of the appropriate identifier scheme.			Concur
[R 73]	The xsd:nillable attribute MUST NOT be used.	GXS16	The built in xsd:nillable attribute MUST NOT be used for any UBL declared element.	
[R 74]	Empty elements MUST NOT be used.	ELD7	Empty elements MUST not be declared, except in the case of extension, where the 'UBLExtensions' element is used.	By "MUST NOT be used", do you mean "MUST NOT be declared"? Is part of our localization/customization methodology.
[R 75]	Every BBIE leaf element declaration MUST be of the udt:UnqualifiedDataType or qdt:QualifiedDataType that represents the source basic business information entity (BBIE) data type.	ELD3	For every class and property identified in the UBL model, a global element bound to the corresponding xsd:complexType MUST be declared.	e.g. <xsd:element name="TaxableAmount" type="TaxableAmountType"/>
[R 76]	The xsd:all element MUST NOT be used.	GXS8	The xsd:all element MUST NOT be used.	
[R 77]	All type definitions MUST be named.	GTD1	All types MUST be named.	Concur
[R 78]	Data type definitions with the same semantic meaning MUST NOT have an identical set of facet restrictions.			
[R 79]	xsd:extension MUST only be used in the cct:CoreComponentType schema module and the udt:UnqualifiedDataType schema module. When used it MUST only be used for declaring xsd:attributes to accommodate relevant supplementary components..	CTD4	Every ccts:BBIEProperty xsd:complexType content model xsd:simpleContent element MUST consist of an xsd:extension element.	
[R 80]	When xsd:restriction is applied to a xsd:simpleType or xsd:complexType that represents a data type the derived construct MUST use a different name.			Concur (pending acceptance by the TC)
[R 81]	Each UN/CEFACT defined or declared construct MUST use the xsd:annotation element for required CCTS documentation.			
[R 82]	The root schema module MUST be represented by a unique token.			Clarification needed. Are you talking about namespace prefixes?
[R 83]	The rsm:RootSchema MUST import the following schema modules: – ram:ReusableABIE Schema Module – udt:UnqualifiedDataType Schema Module –			

	qdt:QualifiedDataType Schema Module			
[R 84]	A rsm:RootSchema in one UN/CEFACT namespace that is dependent upon type definitions or element declaration defined in another namespace MUST import the rsm:RootSchema from that namespace.	SSM2	A document schema in one UBL namespace that is dependent upon type definitions or element declarations defined in another namespace MUST only import the document schema from that namespace.	Concur
[R 85]	A rsm:RootSchema in one UN/CEFACT namespace that is dependent upon type definitions or element declarations defined in another namespace MUST NOT import Schema Modules from that namespace other than the rsm:RootSchema.	SSM3	A document schema in one UBL namespace that is dependant upon type definitions or element declarations defined in another namespace MUST NOT import internal schema modules from that namespace.	Concur
[R 86]	The rsm:RootSchema MUST include any internal schema modules that reside in the root schema namespace.			To be consistent, shouldn't this be "…MUST xsd:include any internal schema modules…"
[R 87]	A single global element known as the root element, representing the business information payload, MUST be declared in a rsm:RootSchema.	RED2	The root element MUST be the only global element declared in document schemas.	Concur (pending acceptance by the TC)
[R 88]	The name of the root element MUST be the name of the business information payload with separators and spaces removed.			Isn't it named like any other ABIE?
[R 89]	The root element declaration must be of xsd:complexType that represents the business information payload.			Could be reworded a bit.
[R 90]	Root schema MUST define a single xsd:complexType that fully describes the business information payload.			
[R 91]	The name of the root schema xsd:complexType MUST be the name of the root element with the word 'Type' appended.			Why is this the reverse of other ABIEs, where the type comes first then the element name is derived from the type by removing the word 'Type'?

[R 92]	The rsm:RootSchema root element declaration MUST have a structured set of annotations present in the following pattern: • UniqueID (mandatory): The identifier that references the business information payload instance in a unique and unambiguous way. • Acronym (mandatory): The abbreviation of the type of component. In this case the value will always be RSM. • Name (mandatory): The name of the business information payload. • Version (mandatory): An indication of the evolution over time of a business information payload. • Definition (mandatory): A brief description of the business information payload. • BusinessProcessContextValue (mandatory, repetitive): The business process with which this business information is associated. • GeopoliticalorRegionContextValue (optional, repetitive): The geopolitical/region contexts for this business information payload. • OfficialConstraintContextValue (optional, repetitive): The official constraint context for this business information payload. • ProductContextValue (optional, repetitive): The product context for this business information payload. • IndustryContextValue (optional, repetitive): The industry context for this business information payload. • BusinessProcessRoleContextValue (optional, repetitive): The role context for this business information payload. • SupportingRoleContextValue (optional, repetitive): The supporting role context for this business information payload. • SystemCapabilitiesContextValue (optional, repetitive): The system capabilities context for this business information payload.			
[R 93]	All UN/CEFACT internal schema modules MUST be in the same namespace as their corresponding rsm:RootSchema.	SSM6	All UBL internal schema modules MUST be in the same namespace as their corresponding document schema.	Isn’t this the same as [R 23]?

[R 94]	The internal schema module MUST be represented by the same token as its rsm:RootSchema.			Clarification needed. Are you talking about namespace prefixes?
[R 95]	The Reusable Aggregate Business Information Entity schema module MUST be represented by the token ram.			Clarification needed. Are you talking about namespace prefixes?
[R 96]	The ram:ReusableAggregateBusinessInformationEntity schema MUST import the following schema modules: – udt:UnqualifiedDataType Schema Module – qdt:QualifiedDataType Schema Module			Do you have some sort of ABIE schema module that *isn't* reusable? [Assumes locally declared elements for BBIEs.]
[R 97]	For every object class (ABIE) identified in the UN/CEFACT syntax-neutral model, a named xsd:complexType MUST be defined.	CTD1	For every class identified in the UBL model, a named xsd:complexType MUST be defined.	Concur (pending acceptance by the TC)
[R 98]	The name of the ABIE xsd:complexType MUST be the ccts:DictionaryEntryName with the spaces and separators removed, approved abbreviations and acronyms applied, and with the 'Details' suffix replaced with 'Type'.	CTN1	A UBL xsd:complexType name based on an ccts:Aggregate BusinessInformationEntity MUST be the ccts:DictionaryEntryName with the separators removed and with the "Details" suffix replaced with "Type".	Concur
[R 99]	Every aggregate business information entity (ABIE) xsd:complexType definition content model MUST use the xsd:sequence and/or xsd:choice elements to reflect each property (BBIE or ASBIE) of its class.	CTD2	Every ccts:ABIE xsd:complexType definition content model MUST use the xsd:sequence element containing references to the appropriate global element declarations.	[MJG: I have no objection to this rule. The 'references' reference in our rule goes away if we accept the Hybrid approach (local BBIEs).]
[R 100]	Recursion of xsd:sequence and/or xsd:choice MUST NOT occur.			Why? Can you have a sequence that contains a choice between two sequences? Or a sequence that contains another sequence that has maxOccurs >1?
[R 101]	The order and cardinality of the elements within an ABIE xsd:complexType MUST be according to the structure of the ABIE as defined in the model.			
[R 102]	For each ABIE, a named xsd:element MUST be globally declared.	ELD3	For every class and property identified in the UBL model, a global element bound to the corresponding xsd:complexType MUST be declared.	Difference goes away if we accept Hybrid approach.
[R 103]	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.	ELN1	A UBL global element name based on a ccts:ABIE MUST be the same as the name of the corresponding xsd:complexType to which it is bound, with the word "Type" removed.	Concur.
[R 104]	Every ABIE global element declaration MUST be of the xsd:complexType that represents the ABIE.	ELD3	For every class and property identified in the UBL model, a global element bound to the corresponding xsd:complexType MUST be declared.	

[R 105]	For every attribute of an object class (BBIE) identified in an ABIE, a named xsd:element MUST be locally declared within the xsd:complexType representing that ABIE.			Dependent on acceptance of Hybrid approach.
[R 106]	Each BBIE element name declaration MUST be the property term and qualifiers and the representation term of the basic business information entity (BBIE). Where the word 'identification' is the final word of the property term and the representation term is 'identifier', the term 'identification' MUST be removed. 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.	CTN2 ELN2	A UBL xsd:complexType name based on a ccts:BasicBusiness InformationEntityProperty MUST be the ccts:Dictionary EntryName shared property term and its qualifiers and representation term of the ccts:BasicBusinessInformationEntity, with the separators removed and with the "Type" suffix appended after the representation term. A UBL global element name based on a ccts:BBIEProperty MUST be the same as the name of the corresponding xsd:complexType to which it is bound, with the word "Type" removed.	
[R 107]	If the representation term of a BBIE is 'text', 'text' MUST be removed.	CTN6	A UBL xsd:complexType name based on a ccts:BasicBusiness InformationEntityProperty and with a ccts:BasicBusiness InformationEntityRepresentationTerm of 'Text' MUST have the word "Text" removed from the end of its name.	Although "from the name" probably should be added to the end of the rule.
[R 108]	The BBIE element MUST be based on an appropriate data type that is defined in the UN/CEFACT qdt:QualifiedDataType or udt:UnqualifiedDataType schema modules.	ELD3	For every class and property identified in the UBL model, a global element bound to the corresponding xsd:complexType MUST be declared.	e.g. <xsd:element name="TaxableAmount" type="TaxableAmountType"/> **Also, how does this differ from [R 75].**
[R 109]	For every ASBIE whose ccts:AssociationType is a composition, a named xsd:element MUST be locally declared.			Hybrid approach.
[R 110]	For each locally declared ASBIE, the 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.	ELN3	A UBL global element name based on a ccts:ASBIE MUST be the ccts:ASBIE dictionary entry name property term and its qualifiers; and the object class term and qualifiers of its associated ccts:ABIE. All ccts:DictionaryEntryName separators MUST be removed.	
[R 111]	For each locally declared ASBIE, the element declaration MUST be of the xsd:complexType that represents its associated ABIE.			Hybrid Approach
[R 112]	For every ASBIE whose ccts:AssociationType is not a composition, the globally declared element for the associated ABIE must be referenced using xsd:ref.	ELD4	When a ccts:ASBIE is unqualified, it is bound via reference to the global ccts:ABIE element to which it is associated.	

[R 113]	For every ABIE xsd:complexType definition a structured set of annotations MUST be present in the following pattern: • UniqueID (mandatory): The identifier that references an ABIE instance in a unique and unambiguous way. • Acronym (mandatory): The abbreviation of the type of component. In this case the value will always be ABIE. • DictionaryEntryName (mandatory): The official name of an ABIE. • Version (mandatory): An indication of the evolution over time of an ABIE instance. • Definition (mandatory): The semantic meaning of an ABIE. • ObjectClassTerm (mandatory): The Object Class Term of the ABIE. • ObjectClassQualifierTerm (optional): Qualifies the Object Class Term of the ABIE. • UsageRule (optional, repetitive): A constraint that describes specific conditions that are applicable to the ABIE. • BusinessTerm (optional, repetitive): A synonym term under which the ABIE is commonly known and used in the business. • BusinessProcessContextValue (optional, repetitive): The business process with which this ABIE is associated. • GeopoliticalorRegionContexValuet (optional, repetitive): The geopolitical/region contexts for this ABIE. • OfficialConstraintContextValue (optional, repetitive): The official constraint context for this ABIE. • ProductContextValue (optional, repetitive): The product context for this ABIE. • IndustryContextValue (optional, repetitive): The industry context for this ABIE. • BusinessProcessRoleContextValue (optional, repetitive): The role context for this ABIE. • SupportingRoleContextValue (optional, repetitive): The supporting role context for this ABIE. • SystemCapabilitiesContextValue (optional, repetitive): The system capabilities context for this ABIE. • Example (optional, repetitive): Example of a possible value of an ABIE.	DOC5	The xsd:documentation element for every Aggregate Business Information Entity MUST contain a structured set of annotations in the following sequence and pattern: ComponentType (mandatory): The type of component to which the object belongs. For Aggregate Business Information Entities this must be “ABIE”. DictionaryEntryName (mandatory): The official name of the Aggregate Business Information Entity . Version (optional): An indication of the evolution over time of the Aggregate Business Information Entity. Definition(mandatory): The semantic meaning of the Aggregate Business Information Entity. ObjectClassQualifier (optional): The qualifier for the object class. ObjectClass(mandatory): The Object Class represented by the Aggregate Business Information Entity. AlternativeBusinessTerms (optional): Any synonym terms under which the Aggregate Business Information Entity is commonly known and used in the business.	Concur (pending acceptance by the TC)

[R 114]	For every ABIE xsd:element declaration definition, a structured set of annotations MUST be present in the following pattern: • UniqueID (mandatory): The identifier that references an ABIE instance in a unique and unambiguous way. • Acronym (mandatory): The abbreviation of the type of component. . In this case the value will always be ABIE. • DictionaryEntryName (mandatory): The official name of an ABIE. • Version (mandatory): An indication of the evolution over time of an ABIE instance. • Definition (mandatory): The semantic meaning of an ABIE. • ObjectClassTerm (mandatory): The Object Class Term of the ABIE. • ObjectClassQualifierTerm (optional): Qualifies the Object Class Term of the ABIE. • UsageRule (optional, repetitive): A constraint that describes specific conditions that are applicable to the ABIE. • BusinessTerm (optional, repetitive): A synonym term under which the ABIE is commonly known and used in the business. • BusinessProcessContextValue (optional, repetitive): The business process with which this ABIE is associated. • GeopoliticalorRegionContextValue (optional, repetitive): The geopolitical/region contexts for this ABIE. • OfficialConstraintContextValue (optional, repetitive): The official constraint context for this ABIE. • ProductContextValue (optional, repetitive): The product context for this ABIE. • IndustryContextValue (optional, repetitive): The industry context for this ABIE. • BusinessProcessRoleContextValue (optional, repetitive): The role context for this ABIE. • SupportingRoleContextValue (op 4685 tional, repetitive): The supporting role context for this ABIE. • SystemCapabilitiesContext Value(optional, repetitive): The system capabilities context for this ABIE. • Example (optional, repetitive): Example of a possible value of an ABIE.			

DOC6	For every BBIE xsd:element declaration a structured set of annotations MUST be present in the following pattern: • UniqueID (mandatory): The identifier that references a BBIE instance in a unique and unambiguous way. • Acronym (mandatory): The abbreviation of the type of component. In this case the value will always be BBIE. • DictionaryEntryName (mandatory): The official name of the BBIE. • VersionID (mandatory): An indication of the evolution over time of a BBIE instance. • Definition (mandatory): The semantic meaning of the BBIE. • Cardinality (mandatory): Indication whether the BIE Property represents a not-applicable, optional, mandatory and/or repetitive characteristic of the ABIE. • ObjectClassTerm (mandatory): The Object Class Term of the parent ABIE. • ObjectClassQualifierTerm (optional): Qualifies the Object Class Term of the parent ABIE. • PropertyTerm (mandatory): The Property Term of the BBIE. • PropertyQualifierTerm (optional): Qualifies the Property Term of the BBIE. • PrimaryRepresentationTerm (mandatory): The Primary Representation Term of the BBIE. • UsageRule (optional, repetitive): A constraint that describes specific conditions that are applicable to the BBIE. • BusinessProcessContextValue (optional, repetitive): The business process with which this BBIE is associated. • GeopoliticalorRegionContextValue (optional, repetitive): The geopolitical/region contexts for this BBIE. • OfficialConstraintContextValue (optional, repetitive): The official constraint context for this BBIE. • ProductContextValue (optional, repetitive): The product context for this BBIE. • IndustryContextValue (optional, repetitive): The industry context for this BBIE. • BusinessProcessRoleContextValue (optional, repetitive): The role context for this BBIE. • SupportingRoleContextValue (optional, repetitive): The supporting role context for this BBIE.			

[R 116]	For every ASBIE xsd:element declaration a structured set of annotations MUST be present in the following pattern: • UniqueID (mandatory): The identifier that references an ASBIE instance in a unique and unambiguous way. • Acronym (mandatory): The abbreviation of the type of component. In this case the value will always be ASBIE. • DictionaryEntryName (mandatory): The official name of the ASBIE. • Version (mandatory): An indication of the evolution over time of the ASBIE instance. • Definition (mandatory): The semantic meaning of the ASBIE. • Cardinality (mandatory): Indication whether the ASBIE Property represents a not applicable, optional, mandatory and/or repetitive characteristic of the ABIE. • ObjectClassTerm (mandatory): The Object Class Term of the associating ABIE. • ObjectClassQualifierTerm (optional): A term that qualifies the Object Class Term of the associating ABIE. • AssociationType (mandatory): The Association Type of the ASBIE. • PropertyTerm (mandatory): The Property Term of the ASBIE. • PropertyQualifierTerm (Optional): A term that qualifies the Property Term of the ASBIE. • AssociatedObjectClassTerm (mandatory): The Object Class Term of the associated ABIE. • AssociatedObjectClassQualifierTerm (optional): Qualifies the Object Class Term of the associated ABIE. • BusinessProcessContextValue (optional, repetitive): The business process with which this ASBIE is associated. • GeopoliticalorRegionContextValue (optional, repetitive): The geopolitical/region contexts for this ASBIE. • OfficialConstraintContextValue (optional, repetitive): The official constraint context for this ASBIE. • ProductContextValue (optional, repetitive): The product context for this ASBIE. • IndustryContextValue (optional, repetitive): The industry context for this ASBIE. • BusinessProcessRoleContextValue (optional,	DOC6	The xsd:documentation element for every Association Business Information Entity element declaration MUST contain a structured set of annotations in the following sequence and pattern: ComponentType (mandatory): The type of component to which the object belongs. For Association Business Information Entities this must be "ASBIE". DictionaryEntryName (mandatory): The official name of the Association Business Information Entity. Version (optional): An indication of the evolution over time of the Association Business Information Entity. Definition(mandatory): The semantic meaning of the Association Business Information Entity. Cardinality(mandatory): Indication whether the Association Business Information Entity represents an optional, mandatory and/or repetitive assocation. ObjectClass(mandatory): The Object Class containing the Association Business Information Entity. PropertyTermQualifier (optional): A qualifier is a word or words which help define and differentiate the Association Business Information Entity. PropertyTerm(mandatory): Property Term represents the Aggregate Business Information Entity contained by the Association Business Information Entity. AssociatedObjectClassQualifier (optional): Associated Object Class Qualifiers describe the 'context' of the relationship with another ABIE. That is, it is the role the contained Aggregate Business Information Entity plays within its association with the containing Aggregate Business Information Entity. AssociatedObjectClass (mandatory): Associated Object Class	Concur (pending acceptance by the TC)

[R 117]	The core component type (CCT) schema module MUST be represented by the token cct.			Clarification needed. Are you talking about namespace prefixes?
[R 118]	The cct:CoreCoreComponentType schema module MUST NOT include or import any other schema modules.			
[R 119]	Every core component type MUST be defined as a named xsd:complexType in the cct:CoreComponentType schema module.			
[R 120]	The name of each xsd:complexType based on a core component type MUST be the dictionary entry name of the core component type (CCT), with the separators and spaces removed and approved abbreviations applied.			
[R 121]	Each core component type xsd:complexType definition MUST contain one xsd:simpleContent element.			
[R 122]	The core component type xsd:complexType definition xsd:simpleContent element MUST contain one xsd:extension element. This xsd:extension element must include an XSD based attribute that defines the specific XSD built-in data type required for the CCT content component.			
[R 123]	Within the core component type xsd:extension element a xsd:attribute MUST be declared for each supplementary component pertaining to that core component type.			
[R 124]	Each core component type supplementary component xsd:attribute name MUST be the CCTS supplementary component dictionary entry name with the separators and spaces removed.			
[R 125]	If the object class of the supplementary component dictionary entry name contains the name of the representation term of the parent CCT, the duplicated object class word or words MUST be removed from the supplementary component xsd:attribute name.			
[R 126]	If the object class of the supplementary component dictionary entry name contains the term ‘identification’, the term ‘identification’ MUST be removed from the supplementary component xsd:attribute name.			

[R 127]	If the representation term of the supplementary component dictionary entry name is ‘text’, the representation term MUST be removed from the supplementary component xsd:attribute name.			
[R 128]	The attribute representing the supplementary component MUST be based on the appropriate XSD built-in data type.			
[R 129]	For every core component type xsd:complexType definition a structured set of annotations MUST be present in the following pattern: • UniqueID (mandatory): The identifier that references the Core Component Type instance in a unique and unambiguous way. • Acronym (mandatory): The abbreviation of the type of component. . In this case the value will always be CCT. • DictionaryEntryName (mandatory): The official name of a Core Component Type. • Version (mandatory): An indication of the evolution over time of a Core Component Type instance. • Definition (mandatory): The semantic meaning of a Core Component Type. • PrimaryRepresentationTerm (mandatory): The primary representation term of the Core Component Type. • PrimitiveType (mandatory): The primitive data type of the Core Component Type. • UsageRule (optional, repetitive): A constraint that describes specific conditions that are applicable to the Core Component Type. • BusinessTerm (optional, repetitive): A synonym term under which the Core Component Type is commonly known and used in the business. • Example (optional, repetitive): Example of a possible value of a Core Component Type.			

[R 130]	For every supplementary component xsd:attribute declaration a structured set of annotations MUST be present in the following pattern: • Name (mandatory): The official name of the Supplementary Component. • Definition (mandatory): The semantic meaning of the Supplementary Component. • ObjectClassTerm (mandatory): The Object Class of the Supplementary Component. • PropertyTerm (mandatory): The Property Term of the Supplementary Component. • PrimitiveType (mandatory): The primitive data type of the Supplementary Component. • UsageRule (optional, repetitive): A constraint that describes specific conditions that are applicable to the Supplementary Core Component. • Example (optional, repetitive): Example of a possible value of a Basic Core Component.	DOC8	The xsd:documentation element for every Supplementary Component attribute declarationMUST contain a structured set of annotations in the following sequence and pattern: Name (mandatory): Name in the Registry of a Supplementary Component of a Core Component Type. Definition (mandatory): A clear, unambiguous and complete explanation of the meaning of a Supplementary Component and its relevance for the related Core Component Type. Primitive type (mandatory): PrimitiveType to be used for the representation of the value of a Supplementary Component. Possible Value(s) (optional): one possible value of a Supplementary Component.	Concur (pending acceptance by the TC)
[R 131]	The Unqualified Data Type schema module namespace MUST be represented by the token udt.	NMS17	The ccts:UnqualifiedDatatypes schema module namespace MUST be represented by the token “udt”when referenced in other schemas.	Actually, they both should be modified to read something like: “The Unqualified Data Type schema module namespace name MUST be assigned the namespace prefix “udt”.
[R 132]	The udt:UnqualifiedDataType schema MUST only import the following schema modules: – ids:IdentifierList schema modules – clm:CodeList schema modules			
[R 133]	An unqualified data type MUST be defined for each approved primary and secondary representation terms identified in the CCTS Permissible Representation Terms table.			
[R 134]	The name of each unqualified data type MUST be the dictionary entry name of the primary or secondary representation term, with the word ‘Type’ appended, the separators and spaces removed and approved abbreviations applied.			

[R 135]	For every unqualified data type whose supplementary components map directly to the properties of a XSD built-in data type, the unqualified data type MUST be defined as a named xsd:simpleType in the udt:UnqualifiedDataType schema module.			
[R 136]	Every unqualified data type xsd:simpleType MUST contain one xsd:restriction element. This xsd:restriction element MUST include an xsd:base attribute that defines the specific XSD built-in data type required for the content component.			
[R 137]	For every unqualified data type whose supplementary components are not equivalent to the properties of a XSD built-in data type, the unqualified data type MUST be defined as an xsd:complexType in the udt:UnqualifiedDataType schema module.			
[R 138]	Every unqualified data type xsd:complexType definition MUST contain one xsd:simpleContent element.			
[R 139]	Every unqualified data type xsd:complexType xsd:simpleContent element MUST contain one xsd:extension element. This xsd:extension element must include an xsd:base attribute that defines the specific XSD built-in data type required for the content component.			
[R 140]	Within the unqualified data type xsd:complextype xsd:extension element an xsd:attribute MUST be declared for each supplementary component pertaining to the underlying CCT.			
[R 141]	Each supplementary component xsd:attribute name MUST be the supplementary component name with the separators and spaces removed, and approved abbreviations and acronyms applied.			
[R 142]	If the object class of the supplementary component dictionary entry name contains the name of the representation term, the duplicated object class word or words MUST be removed from the supplementary component xsd:attribute name.			
[R 143]	If the object class of the supplementary component dictionary entry name contains the term ‘identification’, the term ‘identification’ MUST be removed from the supplementary component xsd:attribute name.			

[R 144]	If the representation term of the supplementary component dictionary entry name is ‘text’, the representation term MUST be removed from the supplementary component xsd:attribute name.			
[R 145]	If the representation term of the supplementary component is ’Code’ and validation is required, then the attribute representing this supplementary component MUST be based on the defined xsd:simpleType of the appropriate external imported code list.			
[R 146]	If the representation term of the supplementary component is ’Identifier’ and validation is required, then the attribute representing this supplementary component MUST be based on the defined xsd:simpleType of the appropriate external imported identifier list.			
[R 147]	If the representation term of the supplementary component is other than ‘Code’ or ‘Identifier’, then the attribute representing this supplementary component MUST be based on the appropriate XSD built-in data type.			
[R 148]	For every unqualified data type xsd:complexType or xsd:simpleType definition a structured set of annotations MUST be present in the following pattern: • UniqueID (mandatory): The identifier that references an Unqualified Data Type instance in a unique and unambiguous way. • Acronym (mandatory): The abbreviation of the type of component. In this case the value will always be UDT. • DictionaryEntryName (mandatory): The official name of the Unqualified Data Type. • Version (mandatory): An indication of the evolution over time of the Unqualified Data Type instance. • Definition (mandatory): The semantic meaning of the Unqualified Data Type. • PrimitiveType (mandatory): The primitive data type of the Unqualified Data Type. • UsageRule (optional, repetitive): A constraint that describes specific conditions that are applicable to the Unqualified Data Type. • Example (optional, repetitive): Example of a possible value of an Unqualified Data Type.			

[R 149]	For every supplementary component xsd:attribute declaration a structured set of annotations MUST be present in the following pattern: • UniqueID (mandatory): The identifier that references a Supplementary Component instance in a unique and unambiguous way. • Acronym (mandatory): The abbreviation of the type of component. In this case the value will always be SC. • Dictionary Entry Name (mandatory): The official name of the Supplementary Component. • Definition (mandatory): The semantic meaning of the Supplementary Component. • ObjectClassTermName (mandatory): The Object Class of the Supplementary Component. • PropertyTermName (mandatory): The Property Term of the Supplementary Component. • Example (optional, repetitive): Example of a possible value of a Basic Core Component.	DOC8	The xsd:documentation element for every Supplementary Component attribute declarationMUST contain a structured set of annotations in the following sequence and pattern: Name (mandatory): Name in the Registry of a Supplementary Component of a Core Component Type. Definition (mandatory): A clear, unambiguous and complete explanation of the meaning of a Supplementary Component and its relevance for the related Core Component Type. Primitive type (mandatory): PrimitiveType to be used for the representation of the value of a Supplementary Component. Possible Value(s) (optional): one possible value of a Supplementary Component.	Concur (pending acceptance by the TC) **Also, how is this different than [R 130]?**
[R 150]	The Qualified Data Type schema module namespace MUST be represented by the token qdt.	NMS16	The ubl:QualifiedDatatypes schema module namespace MUST be represented by the namespace prefix “qdt” when referenced in other schemas.	How about: “The Qualified Data Type schema module namespace MUST be represented by the namespace prefix “qdt”.”
[R 151]	The qdt:QualifiedDataType schema module MUST import the udt:UnqualifiedDataType schema module.	SSM20	The UBL Qualified Datatypes schema module MUST import the ccts:UnQualifiedDatatypes schema module.	
[R 152]	Where required to change facets of an existing unqualified data type, a new data type MUST be defined in the qdt:QualifiedDataType schema module.			
[R 153]	A qualified data type MUST be based on an unqualified data type and add some semantic and/or technical restriction to the unqualified data type.	CTD20	A ccts:QualifiedDataType MUST be based on an unqualified data type and add some semantic and/or technical restriction to the unqualified data type.	
[R 154]	The name of a qualified data type MUST be the name of its base unqualified data type with separators and spaces removed and with its qualifier term added.	CTD21	The name of a ccts:QualifiedDataType MUST be the name of its base ccts:UnqualifiedDataType with separators and spaces removed and with its qualifier term added.	

[R 155]	Every qualified data type based on an unqualified data type xsd:complexType whose supplementary components map directly to the properties of a XSD built-in data type MUST be defined as a xsd:simpleType MUST contain one xsd:restrictionelement MUST include a xsd:base attribute that defines the specific XSD built-in data type required for the content component.	CTD22	Every qualified datatype based on an unqualified datatype xsd:complexType whose supplementary components map directly to the properties of an XSD built-in data type MUST be defined as an xsd:simpleType MUST contain one xsd:restriction element MUST include an xsd:base attribute that defines the specific XSD built-in data type required for the content component	
[R 156]	Every qualified data type based on an unqualified data type xsd:complexType whose supplementary components do not map directly to the properties of a XSD built-in data type MUST be defined as a xsd:complexType MUST contain one xsd:simpleContent element MUST contain one xsd:restriction element MUST include the unqualified data type as its xsd:base attribute.	CTD23	Every qualified datatype based on an unqualified datatype xsd:complexType whose supplementary components do not map directly to the properties of an XSD built-in data type MUST be defined as an xsd:complexType MUST contain one xsd:simpleContent element MUST contain one xsd:restriction element MUST include the unqualified datatype as its xsd:base attribute	
[R 157]	Every qualified data type based on an unqualified data type xsd:simpleType MUST contain one xsd:restriction element MUST include the unqualified data type as its xsd:base attribute or if the facet restrictions can be achieved by use of a XSD built-in data type, then that XSD built-in data type may be used as the xsd:base attribute.	CTD24	Every qualified datatype based on an unqualified datatype xsd:simpleType MUST contain one xsd:restriction element MUST include the unqualified datatype as its xsd:base attribute	Concur (pending acceptance by the TC) [MJG: I'm not sure I support the use of the XSD built-in type because you lose the 'connection' with the UDT.]
[R 158]	Every qualified data type based on a single codelist or identifier list xsd:simpleType MUST contain one xsd:restriction element or xsd:union element. When using the xsd:restriction element, the xsd:base attribute MUST be set to the code list or identifier list schema module defined simple type with appropriate namespace qualification. When using the xsd:union element, the xsd:member type attribute MUST be set to the code list or identifier list schema module defined simple types with appropriate namespace qualification.			

[R 159]	Every qualified data type that has a choice of two or more code lists or identifier lists MUST be defined as an xsd:complexType MUST contain the xsd:choice element whose content model must consist of element references for the alternative code lists or identifier lists to be included with appropriate namespace qualification.			
[R 160]	The qualified data type xsd:complexType definition xsd:simpleContent element MUST only restrict attributes declared in its base type, or MUST only restrict facets equivalent to allowed supplementary components.			Cannot restrict content?

[R 161]	Every qualified data type definition MUST contain a structured set of annotations in the following sequence and pattern: • UniqueID (mandatory): The identifier that references a Qualified Data Type instance in a unique and unambiguous way. • Acronym (mandatory): The abbreviation of the type of component. In this case the value will always be QDT. • DictionaryEntryName (mandatory): The official name of the Qualified Data Type. • Version (mandatory): An indication of the evolution over time of the Qualified Data Type instance. • Definition (mandatory): The semantic meaning of the Qualified Data Type. • PrimaryRepresentationTerm (mandatory): The Primary Representation Term of the Qualified Data Type. • PrimitiveType (mandatory): The primitive data type of the Qualified Data Type. • DataTypeQualifierTerm (mandatory): A term that qualifies the Representation Term in order to differentiate it from its underlying Unqualified Data Type and other Qualified Data Type. • BusinessProcessContextValue (optional, repetitive): The business process context for this Qualified Data Type is associated. • GeopoliticalorRegionContextValue (optional, repetitive): The geopolitical/region contexts for this Qualified Data Type. • OfficialConstraintContextValue (optional, repetitive): The official constraint context for this Qualified Data Type. • ProductContextValue (optional, repetitive): The product context for this Qualified Data Type. • IndustryContextValue (optional, repetitive): The industry context for this Qualified Data Type. • BusinessProcessRoleContextValue (optional, repetitive): The role context for this Qualified Data Type. • SupportingRoleContextValue (optional, repetitive): The supporting role context for this Qualified Data Type. • SystemCapabilitiesContextValue (optional, repetitive): The system capabilities context for this Qualified Data Type. • UsageRule (optional, repetitive): A constraint that	DOC1 DOC2 DOC3	The xsd:documentation element for every Datatype MUST contain a structured set of annotations in the following sequence and pattern (as defined in CCTS Section 7): DictionaryEntryName (mandatory) Version (mandatory): Definition(mandatory) RepresentationTerm (mandatory) QualifierTerm(s) (mandatory, where used) UniqueIdentifier (mandatory) Usage Rule(s) (optional) Content Component Restriction (optional) A Datatype definition MAY contain one or more Content Component Restrictions to provide additional information on the relationship between the Datatype and its corresponding Core Component Type. If used the Content Component Restrictions must contain a structured set of annotations in the following patterns: RestrictionType (mandatory): Defines the type of format restriction that applies to the Content Component. RestrictionValue (mandatory): The actual value of the format restriction that applies to the Content Component. ExpressionType (optional): Defines the type of the regular expression of the restriction value. A Datatype definition MAY contain one or more Supplementary Component Restrictions to provide additional information on the relationship between the Datatype and its corresponding Core Component Type. If used the Supplementary Component Restrictions must contain a structured set of annotations in the following patterns: SupplementaryComponentName (mandatory): Identifies the Supplementary Component on which the restriction applies. RestrictionValue (mandatory, repetitive): The actual value(s) that is (are) valid for the Supplementary Component	Concur (pending acceptance by the TC)

[R 162]	For every supplementary component xsd:attribute declaration a structured set of annotations MUST be present in the following pattern: • UniqueID (mandatory): The identifier that references a Supplementary Component of a Core Component Type instance in a unique and unambiguous way. • Acronym (mandatory): The abbreviation of the type of component. In this case the value will always be QDT. • Name (mandatory): The official name of a Supplementary Component. • Definition (mandatory): The semantic meaning of a Supplementary Component. • Cardinality (mandatory): Indication whether the Supplementary Component Property represents a not-applicable, optional, mandatory and/or repetitive characteristic of the Core Component Type. • PropertyTerm (optional): The Property Term of the associated Supplementary Component. • UsageRule (optional, repetitive): A constraint that describes specific conditions that are applicable to the Supplementary Component. • Example (optional, repetitive): Example of a possible value of a Supplementary Component.	DOC8	The xsd:documentation element for every Supplementary Component attribute declarationMUST contain a structured set of annotations in the following sequence and pattern: Name (mandatory): Name in the Registry of a Supplementary Component of a Core Component Type. Definition (mandatory): A clear, unambiguous and complete explanation of the meaning of a Supplementary Component and its relevance for the related Core Component Type. Primitive type (mandatory): PrimitiveType to be used for the representation of the value of a Supplementary Component. Possible Value(s) (optional): one possible value of a Supplementary Component.	Concur (pending acceptance by the TC) **Also, how is this different than [R 130] and [R 149]?**
[R 163]	Each UN/CEFACT maintained code list MUST be defined in its own schema module.			UBL supports OASIS Code List Representation TC - Genericode
[R 164]	Internal code list schema MUST NOT duplicate existing external code list schema when the existing ones are available to be imported.			UBL supports OASIS Code List Representation TC - Genericode

[R 165]	The namespace names for code list schemas MUST have the following structure while the schema is at draft status: urn:un:unece:uncefact:codelist:draft:<Code List Agency Identifier|Code List Agency Name Text>:<Code List Identification. Identifier|Code List Name Text>:<Code List Version. Identifier> Where: codelist = this token identifying the schema as a code list Code List Agency Identifier = identifies the agency that manages a code list. The default agencies used are those from DE 3055 but roles defined in DE 3055 cannot be used. Code List Agency Name Text = the name of the agency that maintains the code list. Code List Identification Identifier = identifies a list of the respective corresponding codes. listID is only unique within the agency that manages this code list. Code List Name Text = the name of a list of codes. Code List Version Identifier = identifies the version of a code list.			UBL supports OASIS Code List Representation TC - Genericode
[R 166]	The namespace names for code list schema holding specification status MUST be of the form: urn:un:unece:uncefact:codelist:standard:<Code List. Agency Identifier|Code List Agency Name Text>:<Code List Identification. Identifier|Code List Name Text>:<Code List Version Identifier> Where: codelist = this token identifying the schema as a code list Code List Agency Identifier = identifies the agency that manages a code list. The default agencies used are those from DE 3055 but roles defined in DE 3055 cannot be used. Code List Agency Name Text = the name of the agency that maintains the code list. Code List IdentificationIdentifier = identifies a list of the respective corresponding codes. listID is only unique within the agency that manages this code list. Code List Name Text = the name of a list of codes. Code List Version Identifier = identifies the version of a code list.			UBL supports OASIS Code List Representation TC - Genericode
[R 167]	Each UN/CEFACT maintained code list schema module MUST be represented by a unique token constructed as follows: clm[Qualified data type name]<Code List Agency Identifier|Code List Agency Name Text><Code List Identification Identifier|Code List Name Text> with any repeated words eliminated.			

[R 168]	The structure for schema location of code lists MUST be: http://www.unece.org/uncefact/codelist/<status>/<Code List. Agency Identifier|Code List Agency Name Text>/<Code List Identification Identifier|Code List Name Text>_<Code List Version Identifier>.xsd Where: schematype = a token identifying the type of schema module: codelist status = the status of the schema as: draft|standard Code List Agency Identifier = identifies the agency that manages a code list. The default agencies used are those from DE 3055 but roles defined in DE 3055 cannot be used. Code List Agency Name Text = the name of the agency that maintains the code list. Code List Identification Identifier = identifies a list of the respective corresponding codes. listID is only unique within the agency that manages this code list. Code List Name Text = the name of a list of codes. Code List Version Identifier = identifies the version of a code list.			
[R 169]	Each xsd:schemaLocation attribute declaration of a code list MUST contain a persistent and resolvable URL.			
[R 170]	Each xsd:schemaLocation attribute declaration URL of a code list MUST contain an absolute path.			
[R 171]	Code List schema modules MUST not import or include any other schema modules.			
[R 172]	Within each code list module one, and only one, named xsd:simpleType MUST be defined for the content component.			
[R 173]	The name of the xsd:simpleType MUST be the name of code list root element with the word 'ContentType' appended.			
[R 174]	The xsd:restriction element base attribute value MUST be set to xsd:token.			
[R 175]	Each code in the code list MUST be expressed as an xsd:enumeration, where the xsd:value for the enumeration is the actual code value.			
[R 176]	For each code list a single root element MUST be globally declared.			

[R 177]	The name of the code list root element MUST be the name of the code list following the naming rules as defined in section 5.3.			
[R 178]	The code list root element MUST be of a type representing the actual list of code values.			
[R 179]	Each code list xsd:enumeration MUST contain a structured set of annotations in the following sequence and pattern: • Name (mandatory): The name of the code. • Description (optional): Descriptive information concerning the code.			
[R 180]	Internal identifier lists schema MUST NOT duplicate existing external identifier list schema when the existing ones are available to be imported.			
[R 181]	Each UN/CEFACT maintained identifier list MUST be defined in its own schema module.			
[R 182]	The names for namespaces MUST have the following structure while the schema is at draft status: urn:un:unece:uncefact:identifierlist:draft:<Identifier Scheme. Agency Identifier|Identifier Scheme Agency Name Text>:<Identifier Scheme Identifier|Identifier Scheme Name Text>:<Identifier Scheme Version Identifier> Where: identifierlist = this token identifying the schema as an identifier scheme Identifier Scheme Agency Identifier = the identification of the agency that maintains the identification scheme. IdentifierScheme Agency Name. Text = the name of the agency that maintains the identification list. Identifier Scheme Identifier = the identification of the identification scheme. Identifier Scheme Name. Text = the name of the identification scheme. Identifier Scheme Version. Identifier = the version of the identification scheme.			

[R 183]	The namespace names for identifier list schema holding specification status MUST be of the form: urn:un:unece:uncefact:identifierlist:standard:<Identifier Scheme. Agency Identifier|Identifier Scheme Agency Name Text>:<Identifier Scheme Identifier|Identifier Scheme Name Text>:<Identifier Scheme. Version Identifier> Where: identifierlist = this token identifying the schema as an identifier scheme Identifier Scheme Agency Identifier = the identification of the agency that maintains the identification scheme. Identifier Scheme Agency Name. Text = the name of the agency that maintains the identification scheme. Identifier Scheme Identifier = the identification of the identification scheme. Identifier Scheme Name. Text = the name of the identification scheme. Identifier Scheme Version. Identifier = the version of the identification scheme.			
[R 184]	Each UN/CEFACT maintained identifier list schema module MUST be represented by a unique token constructed as follows: ids[Qualified data type name]<Identification Scheme Agency Identifier><Identification Scheme Identifier> with any repeated words eliminated.			
[R 185]	The structure for schema location of identifier lists MUST be: http://www.unece.org/uncefact/identifierlist/<status>/<Identifier Scheme Agency Identifier|Identifier Scheme Agency Name Text>/< Identifier Scheme Identifier|Identifier Scheme Name Text>_< Identifier Scheme Version Identifier>.xsd Where: schematype = a token identifying the type of schema module: identifierlist status = the status of the schema as: draft|standard Identifier Scheme. Agency Identifier = the identification of the agency that maintains the identification scheme. Identifier Scheme. Agency Name. Text = the name of the agency that maintains the identification scheme. Identifier Scheme. Identifier = the identification of the identification scheme. Identifier Scheme. Name. Text = the name of the identification scheme. Identifier Scheme. Version. Identifier = the version of the identification scheme.			

[R 186]	Each xsd:schemaLocation attribute declaration of an identifier list schema MUST contain a persistent and resolvable URL.			
[R 187]	Each xsd:schemaLocation attribute declaration URL of an identifier list schema MUST contain an absolute path.			
[R 188]	Identifier list schema modules MUST NOT import or include any other schema modules.			
[R 189]	Within each identifier list schema module one, and only one, named xsd:simpleType MUST be defined for the content component.			
[R 190]	The name of the xsd:simpleType MUST be the name of the identifier list root element with the word 'ContentType' appended.			
[R 191]	The xsd:restriction element base attribute value MUST be set to xsd:token.			
[R 192]	Each identifier in the identifier list MUST be expressed as an xsd:enumeration, where the xsd:value for the enumeration is the actual identifier value.			
[R 193]	Facets other than xsd:enumeration MUST NOT be used in the identifier list schema module.			
[R 194]	For each identifier list a single root element MUST be globally declared.			
[R 195]	The name of the identifier list root element MUST be the name of the identifier list following the naming rules as defined in section 5.3.			
[R 196]	The identifier list root element MUST be of a type representing the actual list of identifier values.			
[R 197]	Each xsd:enumeration MUST contain a structured set of annotations in the following sequence and pattern: • Name (mandatory): The name of the identifier. • Description (optional): Descriptive information concerning the identifier.			
[R 198]	All UN/CEFACT XML MUST be instantiated using UTF . UTF-8 should be used as the preferred encoding. If UTF-8 is not used, UTF-16 MUST be used.	GXS1	UBL Schema MUST conform to the following physical layout as applicable: <!-- ======= XML Declaration======== --> <?xml version="1.0" encoding="UTF-8"?> …	

[R 199]	The xsi prefix MUST be used where appropriate for referencing xsd:schemaLocation and xsd:noNamespaceLocation attributes in instance documents.			
[R 200]	UN/CEFACT conformant instance documents MUST NOT contain an element devoid of content.	IND5	UBL conformant instance documents MUST NOT contain an element devoid of content or containing null values, except in the case of extension, where theUBLExtensionContent element is used.	Needed for extension mechanism.
[R 201]	The xsi:nil attribute MUST NOT appear in any conforming instance.			
[R 202]	The xsi:type attribute MUST NOT be used.			Just thinking about more general use…