UBL Schema Naming and Design Rules Checklist

General Comments:

Make all the MUST and ~~must~~*MUST* the same. Either italics or uppercase, don't mix.

1."We formally accepted the proposal to use elements for everything, except for using attributes for supplementary components"

Find rule

~~1.~~2.

***Last modified:* 8/5/2003 9:17 PM~~7/21/2003 10:25 AM~~**

Rule Num ber	**Rule** ~~(these are not changed)~~	**Sent for email vote**	**STATUS~~LC SC~~**	**Comment** (Look for rewrites and changes to rules)
[R 1]	All UBL schema design rules MUST be based on the W3C XML Schema Recommendations: XML Schema Part 1: Structures and XML Schema Part 2: Datatypes.		ACCEPTED ~~Yes~~	ACCEPTED.
[R 2]	All UBL schema~~ta~~ and messages ~~must~~MUST be based on the W3C suite of technical specifications holding recommendation status.	7/10/03 DUE 7/17/03	ACCEPTED	ACCEPTED with change. Change wording to schemas not schemata. Tony Coates: The W3C XML Schema group have always insisted that "Schemas" is the correct plural for W3C XML Schemas, not withstanding that "schemata" is appropriate for some other kinds of schema things in the universe.
[R 3]	Each dictionary entry name ~~must~~MUST define one and only one fully qualified path (FQP) for an element or attribute.	7/10/03 DUE 7/17/03	ACCEPTED ~~Yes~~	ACCEPTED. CK: Seems to suggest that dictionary entry name use XPath, but then current usage of dictionary entry name doesn't do that. So which is which?
[R 4]	Names ~~must~~MUST be in the English language, using the primary English spellings provided in the Oxford English Dictionary.		~~Yes~~ACCEPTED	ACCEPTED.
[R 5]	XML names constructed from dictionary entry names ~~must~~MUST not include periods, spaces, or other separators. [R5a] "UBL XML element, attribute and type names MUST be taken from CCTS conformant dictionary entry names. [R5b] UBL XML element, attribute 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" [R5c] In conformance with ISO/IETF/ITU/UNCEFACT Memorandum of Understanding Management Group (MOUMG) Resolution 01/08 (MOU/MG01n83) as agreed to by OASIS, all UBL XML SHOULD be expressed using UTF-8.	7/10/03 DUE 7/17/03	HOLD	Rewritten, to be submitted 7/29/03 This is another rule LCSC has been asking for, should we clarify? The regular expression is not quite right, but we agree. We want clarification, we don't want difficulty in translation or programming within other environments. **CK**: Underscores (_) used as first character to denote "internal usage of some sort" need not be considered separators, but should presumably fall under this guidance of usage avoidance. So suggest that "separators" be replaced by "any other character than the 52 upper and lower case alphabets and the 10 digit characters. In other words, the XML names in UBL should be drawn from the regular expression set [a-zA-Z]+[a-zA-Z0-9]*" Note: this would also help mapping to the namespaces of other languages such as Java. Correction from Dan Vint: Change regular expression to: [a-zA-Z][a-zA-Z0-9]* **Dan Vint**: How about combining the two so a business person can understand as well as the programmer types? So something like this: [R 5] XML names ~~must~~MUST start with a letter (a to z or A to Z) optionally followed by any number of letters or digits (0 to 9, a to z, and A to Z). They may not include

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
				periods, spaces, or other separators. As a regular expression this is represented as: [a-zA-Z][a-zA-Z0-9]* **Chee-Kai Chin:** That's a good idea, Dan. Agree with suggested wording, except 2nd sentence which somewhat weakens it again. May I tap on your wordings to suggest: [R 5] XML names ~~must~~MUST start with a letter (a to z or A to Z) optionally followed by any number of letters or digits (0 to 9, a to z, and A to Z). Other characters not described here, such as punctuations, periods, spaces, or other separators MUST NOT appear in XML names. As a regular expression this is represented as: [a-zA-Z][a-zA-Z0-9]* Mark: This is overcomplicating a very simple rule that covers converting CCTS Dictionary Entry Names into XML names. We should not regurgitate existing XML/XSD Rules, but focus on what we are doing here. The original regular expression "XML names constructed from dictionary entry names ~~must~~MUST not include periods, spaces, or other separators." focuses exclusively on XML names taken from CCTS BIE dictionary entry names. Perhaps what is really needed here are two rules: [Rxxa] "UBL XML element, attribute and type names MUST be taken from CCTS conformant dictionary entry names. [Rxxb] UBL XML element, attribute and type names constructed from CCTS dictionary entry names ~~must~~MUST not include periods, spaces, other separators, or characters not allowed by W3C XML 1.0 for XML Names" 7/29 – We refocused rule to its original intent and split in two. To be sent to list for 5 day comment period.
[R 6]	UBL XML ~~n~~Names ~~must~~MUST ~~not~~MUST NOT use acronyms, abbreviations, or other word truncations, with the following list of exceptions:		DELETE	DELETE See R87, is this the same? Need list of exceptions. CK: Sentence incomplete. What are the exceptions? Currently, in Reusable.xsd, there are definitions of <xsd:element name="BuyersID" type="cct:IdentifierType" /> <xsd:element name="CV2" type="cct:TextType" /> <xsd:element name="UNDGCode" type="cct:CodeType" /> which contain an abbreviations. Are these elements invalid? Or should [R6] be relaxed a little? Though listing them in the list of exceptions may solve existing usage, what about cases when within individual user's context, certain abbreviations are standard usages within their industry or consortiums but not listed in the exceptions here?
[R 7]	~~Names must~~~~MUST~~ ~~not contain non-letter characters unless required by language-specific rules.~~		DELETE	DELETE See R5, we thought this should be combined with that

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
				rule to cover all naming. CK: Made redundant by [R5] modification above. Dan Vint: Based upon Rule 5, this is not a valid option. So we either need to modify rule 5 or not allow "non-letter characters" - whatever they are! 7/29 – we agree this is redundant. Delete
[R 8]	Names ~~must~~MUST-MUST be in singular form unless the concept itself is plural (example: Goods).		ACCEPTED ~~Yes~~	ACCEPTED. CK: Please define a measure of "clarity".
[R 9]	Upper-camel-case (UCC) MUST be used for naming elements and types.		~~Yes~~ACCEPTED	ACCEPTED. Standards document that describes the Upper and Lower case, we need reference there.
[R 10]	Lower-camel-case (LCC) MUST be used for naming attributes.		~~Yes~~ACCEPTED	ACCEPTED.
[R 11]	Every UBL business message ~~must~~MUST have a single corresponding top-level element.		ACCEPTED	ACCEPTED WITH CHANGE: Change to: "every UBL business document ~~must~~MUST have a single root element"
[R 12]	Every top-level element ~~must~~MUST be named according to the portion of the business process that it initiates.		ACCEPTED	ACCEPTED WITH CHANGE: Change to: "every root element in a UBL document ~~must~~MUST be named according to the portion of the business process that it …."
[R 13b]	[R13a] UBL Models MUST define classes based on UN/CEFACT Basic Business Information Entities and Aggregate Business Information Entities. [R13b] For every ~~object~~ class identified in the ~~syntax-neutral model~~UBL model, a named complex type MUST be defin~~itioned~~ed. [r13c] For every class identified in the UBL model, a ~~and a corresponding~~ global element bound to the corresponding complex type MUST be declared. ~~declaration bound to that type must~~MUST ~~be created.~~ [R13d] For every primary representation term used in the UBL model, a named complex type MUST be defined. [R13e] For every secondary representation term used in the UBL model, a named complex type MUST be defined.		HOLD	Somehow tie to the UBL model diagrams. Can we add this as a reference? Syntax-neutral model is not enough information. 7/29 – split into three rules. Send to Dave for concurrence of wording, and then send to list.
[R 14]	[R14a] The name of a complex type based on ~~an object~~a class ~~must~~MUST be the name of the object class, with the separators removed and with the "Details" suffix replaced with "Type". [R14b] A UBL complex type name based on an ABIE MUST be the CCTS dictionary entry name with the separators removed and with the "Details" suffix replaced with "Type" [R14c] A UBL complex type name based on		ACCEPTED	We recommend an appendix that groups the rules all together by type. Examples should be linked either in appendix or by reference. 7/29 – reworded. Adopted 7/30 corrected R14d and R14e and added R14f.

Rule Number	**Rule** ~~(these are not changed)~~	**Sent for email vote**	**STATUS~~LC SC~~**	**Comment** (Look for rewrites and changes to rules)
	a BBIE MUST be the CCTS dictionary entry name property term and qualifiers and representation term, with the separators removed and with the "Type" suffix appended after the representation term. [R14d] A UBL complex type name based on a primary representation term used in the UBL model MUST be the name of the corresponding CCTS:CCT, with the separators removed and with the "Type" suffix appended after the primary representation term name. [R14e] A UBL complex type name based on a secondary representation term used in tthe UBL model MUST be the name of the secondary representation term, with the separators removed and with the "Type" suffix appended after the secondary representation term name. [R14f] A UBL complex type name based on a CCTS:CCT MUST be the Dictionary Entry name of the CCTS:CCT, with the separators removed.			
[R 15a]	An element name in a global element declaration based on an object class ~~must~~MUST be the name of the object class, with the separators removed and with "Details" removed. [R15a] A UBL global element name based on an ABIE MUST be the same as the name of the corresponding complex type to which it is bound, with the word "Type" removed. [R15b] A UBL global element name based on a BBIE MUST be the same as the name of the corresponding complex type to which it is bound, with the word "Type" removed. [R15c] A UBL global element name based on an ASBIE MUST be declared and bound to the complex type of its associated ABIE. [R15d] A UBL global element name based on an ASBIE MUST be the CCTS ASBIE dictionary entry name property term and qualifiers; and the object class term and qualifiers of its associated ABIE. All CCTS Dictionary Entry Name separators MUST be removed. Redundant words in the ASBIE Property Term or Qualifiers and the associated ABIE object class term or qualifiers MUST be dropped.		ACCEPTED	See comment above. 7/29 - Reworded for clarification and correction. Will be submitted to full list for approval. [
[R 16]	~~For every complex type definition based on an object class, its content model~~ ~~must~~MUST ~~be defined such that it reflects~~		ACCEPTED	Group together like rules, Complex-types together, elements together.

Rule Num ber	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC~~ ~~SC~~	Comment (Look for rewrites and changes to rules)
	~~each property of the object class as an element declaration, with its cardinality and positioning within the content model determined by the details of the syntax-neutral model.~~ Every ABIE complex type definition content model MUST use the XSD: Sequence element with appropriate global element references, or local element declarations in the case of ID and Code, to reflect each property of its class as defined in the corresponding UBL model. [R16b] The XSD:all element MUST NOT be used. [R16c] The XSD:Choice element MUST NOT be used. [R16d] Every BBIE complex type definition content model MUST use the XSD:SimpleContent element. [R16e] Every BBIE ComplexType content model SimpleContent element MUST consist of an XSD:Extension element. [R16f] Every BBIE ComplexType content model XSD:Extension element MUST use the XSD:Base attribute to define the basis of each primary or secondary representation term. [R16g] Every BBIE ComplexType Content Model XSD:Base attribute value MUST be the CCT of the primary representation term or the datatype of the secondary representation term as appropriate.			Syntax-neutral model is not enough information, we need a better way to reference this. 7/29 – Reworded and accepted. No further action.
[R 17]	~~An element name in an element declaration [TBD: ref= or name=?] based on a property must~~MUST ~~be the full dictionary name of the property in the syntax-neutral model, with the separators and object class term removed, and with the property term removed if it is identical or similar to the representation term.~~		DELETE	Group together like rules, Complex-types together, elements together. Syntax-neutral model is not enough information, we need a better way to reference this. 7/29 – deleted. This rule is no long requred as it is duplicitive of the new rules contained in 14, 15, and 16.
[R 18]	~~If the object class term would have been helpful in the resulting XML name for clarity, or if needed to differentiate the element and allow it to have a different type association, it should be repeated in the property qualifier field.~~		DELETE	Group together like rules, Complex-types together, elements together, etc. Change to: “The object class term may be repeated in the property qualifier field, where it is deemed helpful for reasons of either XML naming clarity or if needed to differentiate the element and allow it to have a different type association.” 7/29 – Deleted as no longer necessary given the new

Rule Number	**Rule** ~~(these are not changed)~~	**Sent for email vote**	**STATUS**~~LC SC~~	**Comment** (Look for rewrites and changes to rules)
				versions of rules 14, 15, and 16.
[R 19]	~~Every element declaration corresponding to a property must~~MUST ~~be bound to a type corresponding to the property's representation term. Where the representation term corresponds to an object class (aggregate BIE), the complex type corresponding to that object class must~~MUST ~~be used. Where the representation term corresponds to a Core Component Type, the complex type corresponding to that Core Component Type must~~MUST ~~be used.~~		DELETE	Should this be broken down to different rules for each piece? 7/30 After further review, we think this rule is about binding and we have reinstated the rule pending further review. 8/1 – deleted. not necessary given the other rules we have written.
[R 20]	~~A namespace schema module dedicated to defining types corresponding to the Core Component Types must~~MUST ~~be created.~~ [R20a] A schema module defining all CCTS Core Component Types MUST be created. [R20b] The Core Component Type Schema Module MUST be named "CCTS CCT Schema Module" [R20c] The CCTS CCT Schema Module MUST reside in its own namespace. [R20d] The CCTS CCT Schema Module namespace MUST be represented by the token "cct" [R20e] The XSD Facet feature MUST not be used in the CCTS CCT Schema Module. [R20g] A schema module defining all CCTS Primary and Secondary Representation Terms MUST be created. [R20h] The Representation Term Schema Module MUST be named "CCTS Representation Term" Schema Module [R20i] The CCTS Representation Term Schema Module MUST reside in its own namespace [R20j] The CCTS Representation Term Schema Module namespace MUST be represented by the token "rt" [R20k] A schema module defining all UBL Datatypes MUST be created. [R20l] The UBL Datatypes Schema Module MUST be named "UBL Datatypes" schema module [R20m]The UBL Datatypes Schema Module MUST reside in its own namespace [R20n] The UBL Datatypes Schema Module namespace MUST be represented by the token "dt"		HOLD FOR REVIEW	Again, should this be groups with Namespace rules? 8/1/03 - Reworded for clarification and correction. Requires review by full SC
[R 21]	~~Each CCT must~~MUST ~~have at least one corresponding unique complex type and simple type, where the element's content (governed by the xs:simpleContent~~		HOLD FOR REVIEW	CK: Use of "xs:simpleContent" conflicts with [R 107] 7/30 – still working this.

Rule Num ber	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC~~ ~~SC~~	Comment (Look for rewrites and changes to rules)
	~~construct and the CCT's simple type) represents the content component of the CCT and whose attributes (defined in the complex type) each represent a supplementary component of the CCTs.~~ [R21a] User defined attributes SHOULD NOT be used. When used, user defined attributes MUST only convey CCT Supplementary Component information. [R21b] For every CCTS CCT whose supplementary components are equivalent to the properties of a built-in XSD datatype, the Supplementary Components MUST not be expressed as attributes, and the CCT MUST be defined as a named Simple Type in the CCTS CCT Schema Module. [R21c] For every CCTS CCT whose supplementary components are not equivalent to the properties of a built-in XSD:Datatype, the CCT MUST be defined as a named Complex Type in the CCTS CCT Schema Module. [R21d] CCTS:CCT simple and complex types MUST only be bound to leaf elements. [r21e] each CCTS:CCT Complex Type definition MUST contain one XSD:simpleContent element [R21f] The CCTS:CCT ComplexType definition 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 datatype required for the CCTS:ContentComponent of the CCT. [R21g] Within the CCTS:CCT XSD:extension element an XSD:attribute MUST be declared for each CCTS:SupplementaryComponent pertaining to that CCTS:CCT. [R21h] Each CCT:SupplementaryComponent XSD:attribute "name" MUST be the CCTS:SupplementaryComponent dictionary entry name property term and representation term, with the separators removed. [R21i] Each CCT:SupplementaryComponent xsd:attribute "type" MUST define the specific XSD:Built-in Datatype or the user defined simpleType for the CCTS:SupplementaryComponent of the CCT. [R21j] Each			`<xs:complexType name="BinaryObjectType">` `<xs:simpleContent>` `<xs:extension base="xs:base64Binary">` `<xs:attribute name="format" type="xs:token" use="optional"/>` `<xs:attribute name="mimeCode" type="xs:token" use="optional"/>` `<xs:attribute name="characterSetCode" type="xs:token" use="optional"/>` `<xs:attribute name="URI" type="xs:anyURI" use="optional"/>` `<xs:attribute name="filename" type="xs:token" use="optional"/>` `</xs:extension>` `</xs:simpleContent>` [`</xs:complexType>` [note for r21i – the user defined simpleType is the same simpleType enumerated list from the appropriate code list schema module for that type. Rewritten to reflect current CCT paper and XSD. We have combined other rules as appropriate to be all encompassing.

Rule Num ber	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
	CCTS:SupplementaryComponent XSD:attribute user-defined simpleType MUST only be used when the CCTS:SupplementaryComponent is based on a standardized code list for which a UBL conformant code list schema module has been created. [R21k] Each CCTS:SupplementaryComponent XSD:attribute user defined simpleType MUST be the same simpleType from the appropriate UBL conformant code list schema module for that type. [R21l] Each CCTS:Supplementary Component XSD:attribute "use" MUST define the occurance of that CCTS:SupplementaryComponent as either "required", or "optional.			
[R 22]	~~The complex type name corresponding to a CCT must~~MUST ~~be the CCT name, with the periods and spaces removed.~~		DELETE	CK: Make all name-construction guidances point to the modified [R 5]. So it reads something like: "[R 22] The complex type name corresponding to a CCT ~~must~~MUST be the CCT name, an XML name constructed following [R 5]". 31/07 Deleted in favor of rule 14f
[R 23]	~~The name of the simple type corresponding to the content component of a CCT must~~MUST ~~be the content component name, with the periods and spaces removed and with "Type" added to the end~~. [R23a] Each CCTS:CCT simpleType definition name MUST be the CCTS:CCT dictionary entry name with the separators removed. [R23b] For each CCTS:CCT simpleType, an XSD:Restriction element MUST be declared. [R23c] For each CCTS:CCT simpleType XSD:Restriction element, a base attribute MUST be declared. [R23d] Each CCTS:CCT simpleType XSD:Restriction element base attribute value MUST be set to the appropriate XSD datatype.		DELETE	CK: Guidance on "with the periods and spaces removed" should point back to [R 5]. So all name construction rules are standardized leaving no gray area to individual references to name construction. xs:simpleType name="DateTimeType"> <xs:restriction base="xs:dateTime"/> </xs:simpleType>
[R 24]	~~The name of the attribute corresponding to a supplementary component must~~MUST ~~be the name of the supplementary component, with the periods and spaces removed. The first field (the "object class" field) may be truncated, reworded, or removed as necessary for brevity and clarity. If the final field (the "representation term" field) is "Text", it must~~MUST ~~be removed. If the final field is "Identifier", it must~~MUST ~~be replaced with "ID".~~		DELETE	CK: Guidance on "with the periods and spaces removed" should point back to [R 5]. So all name construction rules are standardized leaving no gray area to individual references to name construction. 7/31 – replaced with R21h
[R 25]	~~Mixed content elements should not be used.~~		DELETE	See R97 DELETE R25, keep R97

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
	~~Note that mixed content in business documents is undesirable because white space in mixed content is difficult to handle and complicates processing, and because mixed-content models allow little useful control over the cardinality of elements.~~			
[R 26]	~~The CCT schema module may define a set of one or more common attributes that apply to all UBL elements.~~ [R26a] If a CCTS:SupplementaryComponent XSD:attribute is common to all UBL elements, it MUST be declared as part of a global attribute group in the CCTS:CCT Schema Module. [R26b] If a UBL XSD:SchemaExpression contains one or more common attributes that apply to all UBL elements contained or included or imported therein, the common attributes MUST be declared as part of a global attribute group.		ACCEPTED	This rule is unclear, should be reformulated. 7/31 – Reformatted. Since the only user defined attributes are CCTS:SupplementaryComponents, this rule MUST be focused on them. As such, the only attributes that can be declared locally are CCTS:SupplementaryComponents that are common to all UBL elements. Currently there are no CCTS:SupplementaryComponents that are common to all UBL elements however we have rewritten as r26a. There are some attributes such as xml:lang that may be common to all
[R 27]	~~A common attribute should be declared as a global attribute only in cases where the attribute's meaning is identical no matter what element it is used on, and where the attribute is useful on every UBL element. This rule applies to both external (such as xml:lang) and UBL-specific global attributes.~~ ~~Note that this rule allows for the creation of common attributes that are allowed on every element but are not globally declared, and that need documentation of their meaning in each XML environment in which they are used~~		~~Yes~~DELETE	~~ACCEPTED~~ Deleted as replaced by rules 26a and b.
[R 28]	~~The names of UBL-specific global attributes~~ ~~must~~MUST ~~be based on assigned object class property names, as is done for elements that are properties.~~ [TBD: need example.]		DELETE	Urgently need an example Does this conflict with email resolution MSG00069 from April 2002 which says attributes ~~must~~MUST be used only to represent supplementary components 7/30 – Deleted. This rule is in conflict with our rule on the use of attributes and is already covered by our CCTS:SupplementaryComponent naming rule..
[R 29]	~~Code Lists Must not be enumerated in the core schema. Code Lists Must be enumerated in a schema module using the UBL code list schema template.~~ [R29a] All UBL Codes MUST be part of a UBL or External maintained Code List [R29b] The name of each UBL Code List Schema Module MUST be of the form [{Owning Organization}[Code List Name}{Code List Schema Module} [R29c] All UBL maintained or used Code Lists MUST be enumerated using the UBL Code List Schema Module.		HOLD	Code Lists Must not be enumrated in the core schema. Code Lists ~~must~~MUST be specified using the rules laid down in the Code List paper. 7/30 – Reworded. Awaiting input from Code List group. Reworded. Awaiting input from Code List adhoc committee. Will submit to list when available. .

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
	[R29d]Each UBL maintained Code List Schema Module MUST be maintained in a separate namespace. [R29e] An XSD:Import element MUST be declared for every code list required in a UBL schema. [R29f] The code list XSD:Import element MUST contain the namespace and schema location attributes.			
[R 30]	~~Every type definition and element declaration must~~MUST ~~contain a structured set of annotations in following pattern, where the keyword is typically based on the spreadsheet column heading in the syntax-neutral model and the description is typically based on the content of the spreadsheet field:~~		DELETE	We need the pattern of the annotations for this rule. Is this the embedded documentation rule? CK: Pattern not listed after ending colon. Find a way to describe the field, rather than give it a number or title Combine with Rule 31
[R 31]	Every type definition and element declaration MUST contain a structured set of annotations in the following pattern: ~~The following sets of annotations are required in type definitions and element declarations:~~ UBL UID: The unique identifier assigned to the type in the UBL library. UBL Name: The complete name (not the tag name) of the type per the UBL library. Object Class: The Object Class represented by the type. Dictionary Entry Name: The complete name (not the tag name), which is the unique official name of the BIE or the property in the UBL library. UBL Definition: Documentation of how the type is to be used, written such that it addresses the type's function as a reusable component. Code Lists/Standards: A list of potential standard code lists or other relevant standards that could provide definition of possible values not formally expressed in the UBL structural definitions. Core Component UID: The UID of the Core Component on which the Type is based Business Process Context: A valid value describing the Business Process contexts for which this construct has been designed. Default is "In All Contexts". Geopolitical/Region Context: A valid value describing the Geopolitical/Region contexts		HOLD FOR REVIEW	rewritten We propose that this be delete. 7/30 - No justification given. Change not accepted. We propose that this be deleted, otherwise we need to add too many things. 7/30 - No justification given. Change not accepted. Add: See rule 30. It should also provide an example, to be taken from the spreadsheet's example column. 7/30 – We have combined rules 30 and 31. Example will be placed appropriately. We propose that this be deleted, since there are no approved CC UIDS. 7/30 – Use temporary until final provided per the CCTS. In all BP contexts. 7/30 CCTS defines this value as “in All Contexts”. Change not accepted. In all Geopolitical/Region context . 7/30 CCTS defines this value as “in All Contexts”. Change not accepted.

Rule Num ber	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
	for which this construct has been designed. Default is "In All Contexts". Official Constraints Context: A valid value describing the Official Constraints contexts for which this construct has been designed. Default is "None". Product Context: A valid value describing the Product contexts for which this construct has been designed. Default is "In All Contexts" Industry Context: A valid value describing the Industry contexts for which this construct has been designed. Default is "In All Contexts". Role Context: A valid value describing the Role contexts for which this construct has been designed. Default is "In All Contexts". Supporting Role Context: A valid value describing the Supporting Role contexts for which this construct has been designed. Default is "In All Contexts". System Capabilities Context: A valid value describing the Systems Capabilities contexts for which this construct has been designed. Default is "In All Contexts". 7/30 add [R31a] All relevant metadata as specified in CCTS Section 7 for the concept (BBIE, ABIE, ASBIE, CCT, Representation Term, Datatype) being conveyed.			etc. (Note: this ~~must~~MUST be aligned with the names in section 5.6.1 of CCTS v1.9 – 7/30 concur. 7/30 reworded. Awaiting internal discussion with Gunther. Note: we determined that AppInfo still creates too large of a security concern to have Code Values conveyed and will require them to be conveyed inside a documentation element.
[R 32]	~~The nillable attribute must~~MUST~~ not be used in any UBL schema. The element declaration of xsi:nil shall not appear in any UBL conforming instance.~~		DELETE	See R 94 Delete R94. Change "shall" to "~~must~~MUST" Propose a new rule: Elements that are not declared empty MUST have content. 7/30 – Deleted 32 in favor of 94.
[R 33]	~~The top-level element must~~MUST~~ be globally declared in a UBL root schema.~~ Each UBL Schema MUST declare one global element that defines the overall business process being conveyed in the Schema expression. That global element declaration MUST include an XSD:Annotation child element which MUST further contain an XSD:Documentation child element that declares "This element MUST be conveyed as the root element in any instance document based on this Schema expression."		HOLD FOR REVIEW	~~DELETE~~ Propose: delete this rule as unnecessary, add the need for documenting which elements will be root. This would be needed if local elements, 7/30. There is no way to declare which global element in a schema MUST be the root element. XSD allows any globally declared element to be the root. The rule MUST articulate that the XSD:annotation <documentation> element for the mandatory root element reflect this. written to reflect comments regarding proper identification of root element in instance.
[R 34]	~~If a definition depends on named constructs found in another namespace, then that other namespace must~~MUST~~ be imported as a namespace schema module. The referenced constructs must~~MUST~~ not be~~		ACCEPTED	Rewrite: If a definition depends on named constructs found in another namespace, then that other namespace ~~must~~MUST be imported (using the XSD IMPORT element) as a namespace schema module. The referenced constructs ~~must~~MUST not be directly

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
	~~directly included as Internal Schema Modules.~~ [R34a] Every UBL defined or used Schema module MUST have a namespace declared. [R34b] UBL namespaces MUST only contain UBL developed Schema modules.			included (with the XSD INCLUDE element) as Internal Schema Modules. 7/30 – XSD include works for schema modules that have the same namespace so restating as occures in the initial version of this rule is unnecessary. However, XSD include also works when the included schema module has no namespace. We do not want to use any schema module that has no namespace because of collision issues. Hence the rewrite to Rule 34a. Further, we want to restrict external schema developers from appropiating UBL namespaces, as this would then allow for non conforming schema modules to use the include feature, which would in turn cause the same collision issues. The result of these two rules is that the XSD:Include feature will only be used with UBL schema modules.
[R 35]	~~A namespace may be completely specified within the Root Schema. If for larger namespaces, more schema modules may be defined - call these internal modules. The root schema for that namespace then MUST include those Internal Modules.~~ [R35a] UBL Schema expressions MAY be split into multiple schema modules. [R35b]UBL Schema modules MUST either be be treated as external Schema modules or as internal Schema modules of the root Schema. [R35c] All UBL internal Schema modules MUST be in the same namespace as their corresponding root Schema. [R35d] Each UBL internal Schema module MUST be named "{ParentSchemaModuleName}{InternalSchemaModuleFunction}{Schema Module}.		ACCEPTED	OK as is. Fixed typo 7/30 – changed to more accurately reflect XSD. In the narrative of the NDR document we need to add that this rule does not preclude using the XSD:Import feature for external schema modules.
[R 36]	The namespace names for UBL namespaces MUST~~must~~MUST have the following structure while the schemas are at draft status: urn:oasis:names:tc:ubl:schema:name:major:minor		ACCEPTED	Should this go into a "how to use" section or document. Maybe just reference this?
[R 37]	The namespace names for UBL Schemas holding specification status MUST be of the form: urn:oasis:names:specification:ubl:schema:name:major:minor		ACCEPTED	Reference this in the "how to" section.
[R 38]	~~Schema location must~~MUST ~~include the complete URI which is used to identify schema modules~~. [R38a] Each XSD:SchemaLocation attribute declaration MUST contain a persistant and resolvable URL. [R38b] Each XSD:SchemaLocation attribute declaration URL MUST contain an absolute path.		ACCEPTED	Reference this in the "how to" section. Make more clear saying schema location attribute. Rewrite: Schema location ~~must~~MUST include the complete URL which is used to identify schema modules. Relative paths are not allowed (mark wordsmith).

Rule Number	**Rule** ~~(these are not changed)~~	**Sent for email vote**	**STATUS~~LC~~ ~~SC~~**	**Comment** (Look for rewrites and changes to rules)
[R 39]	UBL schema modules ~~must~~MUST be ~~located~~ hosted under the UBL committee directory: http://www.oasis-open.org/committees/ubl/schema/<schema-mod-name>.xsd		ACCEPTED	Reference this in the "how to" section. Rewrite: UBL schema modules ~~must~~MUST be hosted under the UBL committee directory:
[R 40]	Every UBL Schema and schema module Major version Must have the URI of: urn:oasis:names:tc:ubl:~~Order~~name:major-number:0		ACCEPTED	Reference this in the "how to" section. Change to: Every UBL Schema Major version Must have the URI of the form: urn:oasis:names:tc:ubl:*name*:*major-number*:*0*
[R 41]	[R41a] The first minor version release of a UBL Schema or schema module MUST~~ust~~ have the URI of: urn:oasis:names:tc:ubl:~~Order~~name:major-number.non-zero~~:1~~ [R41b] For UBL Minor version changes, the name of the version construct Must not change (short name not qualified name), unless the intent of the change is to rename the construct.		ACCEPTED	Reference this in the "how to" section. Change to: The first minor version release of a UBL Schema Must have the URI of the form: urn:oasis:names:tc:ubl:*name*:*major-number*:*non-zero* (Example: Order:1:3)
[R 43]	~~The number scheme must~~MUST~~ be the major number is a non-negative integer and the minor number is a non-negative integer.~~ [R43a] Every UBL schema and schema module major version number MUST be a non-negative integer. [R43b] Every UBL schema and schema module minor version number MUST be a non-negative integer.		ACCEPTED ~~Yes~~	ACCEPTED.
[R 44]	~~The CCT types and Reusable types and their namespace, should have a version.~~ [R44a] A UBL schema module may be created for Reusable types.		HOLD FOR REVIEW	See R-44a below. We still have some questions about how this is going to happen. Level of aggregation for this is yet to be determined. Reworded, needs to be sent out to the list.
[R 45]	~~Non-namespaced schema modules, will not have their own versions or namespaces, and thus must~~MUST~~ only be used within the context of and in conjunction with one specified parent.~~		DELETE	See R-45a below.
R-44a	**[R44b]**~~**Import Rule:**~~ A root schema in one UBL namespace ~~"A"~~ that is dependent upon type definitions or element declaration defined in another namespace ~~"B"~~MUST only import~~s~~ ~~B's~~ the RootSchema from that namespace. ~~"A" never imports other (internal) schema modules of "B".~~ [R44c] A RootSchema in one UBL namespace that is dependant upon type definitions or element declarations defined in another namespace MUST NOT import schema modules from that namespace. [R 44d] Imported Schema modules MUST		ACCEPT	This are rules from the ModNamVer v8 paper that should go into the Rules document. These two rules superscede rules 44 and 45.

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
	be fully conformant with UBL naming and design rules.			
R-45a	~~**Include Rule:** The only place XSD "include" is used is within a RootSchema. When a namespace gets large, its type definitions and element declarations may be split into multiple SchemaModules (called InternalModules) and included by the RootSchema for that namespace~~. [R45] The XSD:Include feature MUST only be used within a RootSchema.			This are rules from the ModNamVer v8 paper that should go into the Rules document. 7/30 Reworded for accuracy and alignment with Rule 35.
[R 46]	Each version Must have a namespace.		ACCEPTED ~~Yes~~	ACCEPTED. Move up above namespace rules. Leave that to the discretion of the editor.
[R 47]	Each minor version ~~must~~MUST be given a separate namespace.		ACCEPTED ~~Yes~~	ACCEPTED. Same as above.
[R 48]	A published namespace MUST never be changed.		ACCEPTED ~~Yes~~	ACCEPTED.
[R 49]	When the URN changes to reflect a change in the namespace, this change will be reflected in the version number, either major or minor.		ACCEPTED ~~Yes~~	ACCEPTED.
[R 50]	Minor versioning ~~must~~MUST be limited to declaring new optional constructs, extending existing constructs and refinements of an optional nature.		ACCEPTED ~~Yes~~	ACCEPTED.
[R 51]	Changes in minor versions ~~must~~MUST not break semantic compatibility with prior versions.		ACCEPTED ~~Yes~~	ACCEPTED.
[R 52]	Minor version namespaces ~~must~~MUST reference immediately preceding minor version root schemas.		ACCEPTED	ACCEPTED. Change to say"… ~~must~~MUST reference "immediately" proceeding minor version root schema
[R 53]	A Core Component Type without any restriction of the Content Component ~~must~~MUST be defined by a complexType. This complexType includes a simpleContent group with a extension for all relevant global and local attributes (Supplementary Components) of this Core Component Type. The base type definition of this extension ~~must~~MUST be based on one of the decided built-in datatypes (see table ###).		DELETE	DELETE rewritten under rule 21. Maybe we need clarity? Change to: A Core Component Type ~~must~~MUST be defined by a complexType. [R053a] This complexType Must include a simpleContent group with a extension for all relevant global and local attributes (Supplementary Components) of this Core Component Type. [R-53b] The base type definition of this extension ~~must~~MUST be based on one of the decided built-in datatypes (see table ###). (Gunther and Mark to rewrite this part of the rule.)
[R 54]	If the Content Component of a Core Component Type is restricted by any kind of facets, this Content Component ~~must~~MUST be a restriction of a simpleType. The name of the simpleType ~~must~~MUST be ending with the suffix "Content".	7/10/03 DUE: 7/17/03	DELETE	Replaced by rule 21 - DELETE ACCEPTED.
[R 55]	The Core Component Type with the restricted Content Component ~~must~~MUST refer to the relevant named simpleType.	7/10/03 DUE: 7/17/03	DELETE	Replaced by rule 21 - DELETE ~~ACCEPTED~~.
[R 56]	A restricted Supplementary Component	7/10/0	DELETE	Replaced by rule 21 - DELETE

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
	(local attribute) within a Core Component Type ~~must~~MUST have a restriction of its simpleType. The base type definition of the restriction ~~must~~MUST refer to one of the decided built-in datatypes (see table ###). The restriction itself should have all relevant facets.	3 DUE: 7/17/03		~~ACCEPTED~~.
[R 57]	A complexType of a Basic Core Component as well as Basic Business Information Entity without any additional restrictions ~~must~~MUST be a extension of a simpleContent. The base type definition of the extension ~~must~~MUST refer to the complexType of the relevant Core Component Type.	7/10/03 DUE: 7/17/03	DELETE	Replaced by rule 13-21 - DELETE ~~ACCEPTED~~.
[R 58]	A complexType of a Basic Core Component as well as Basic Business Information Entity with any additional restrictions ~~must~~MUST be a restriction of a simpleContent. The base type definition of the restriction ~~must~~MUST refer to the complexType of the relevant Core Component Type. The element group of the restriction includes all required facets.	7/10/03 DUE: 7/17/03	DELETE	Replaced by rule 13 - 21 - DELETE ~~ACCEPTED.~~
[R 59]	If a global attribute or a Supplementary Component (local attribute) should be restricted within a Basic Core Component as well as a Basic Business Information Entity, there ~~must~~MUST be a restriction of a simpleContent. The base type definition of the restriction ~~must~~MUST refer to the complexType of the relevant Core Component Type. This restriction includes the attribute or attributes, which should be restricted. The simpleType of each attribute ~~must~~MUST be a restriction, again. This restriction includes all relevant facets.	7/10/03 DUE: 7/17/03	DELETE	Replaced by rule 13 - 21 - DELETE ~~ACCEPTED~~.
[R 60]	If a Basic Core Component as well as a Basic Business Information Entity should have one or more restricted Supplementary Components (local attributes) and a restricted Content Component, the simpleContent of the complexType ~~must~~MUST be a restriction. This base type definition of the restriction ~~must~~MUST refer to the complexType of the relevant Core Component Type. This restriction ~~must~~MUST include all facets and restricted attributes. The simpleType of each attribute ~~must~~MUST be a restriction, too. This restriction should have all relevant facets of each restricted attribute.	7/10/03 DUE: 7/17/03	DELETE	Replaced by rule 13 - 21 - DELETE ~~ACCEPTED~~.
[R 61]	UBL Libraries and Schemas MUST only use UN/CEFACT approved Core Component Types.	7/10/03 DUE: 7/17/03	ACCEPTED	ACCEPTED.
[R 62]	The UBL Library should identify and use external standardized code lists rather than develop its own UBL-native code lists.	7/10/03 DUE: 7/17/03	ACCEPTED	ACCEPTED.
[R 63]	The UBL Library may design and use an internal code list where an existing external code list needs to be extended, or where no suitable external code list exists.	7/10/03 DUE: 7/17/03	ACCEPTED	ACCEPTED.

Rule Num ber	**Rule** ~~(these are not changed)~~	**Sent for email vote**	**STATUS~~LC~~ ~~SC~~**	**Comment** (Look for rewrites and changes to rules)
[R 64]	If a UBL code list is created, the lists should be globally scoped (designed for reuse and sharing, using named types and namespaced schema modules) rather than locally scoped (not designed for others to use and therefore hidden from their use).	7/10/03 DUE: 7/17/03	ACCEPTED	ACCEPTED.
[R 65]	For each UBL construct containing a code, the UBL documentation should identify the zero or more code lists that ~~must~~MUST be minimally supported when the construct is used.	7/10/03 DUE: 7/17/03	ACCEPTED	ACCEPTED.
[R 66]	Users of the UBL Library may identify any subset they wish from an identified code list for their own trading community conformance requirements.	7/10/03 DUE: 7/17/03	ACCEPTED	ACCEPTED.
[R 67]	Both standardized and proprietary identifiers within a message are exchanged.	7/11/03 DUE 7/18/03	DELETE	DELETE Mike Grimley: This is not a rule. Unfortunately, I don't think I really understand what the intent was, so I can't offer alternate wording. We can't find where this rule comes from. Look at section 8.2, that is where text comes from, but there is no rule.
~~[R 68]~~	~~For each specific point in time the built in datatype from XML schema (Part 2) must~~MUST ~~be used. These are xsd:time, xsd:date, xsd:dateTime.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~ACCEPTED~~
🟥	~~The expression of duration requires the use of an additional secondary Representation Term called Duration. Type.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~DELETE RULE~~ ~~MarkCrawford: Against. We have already agreed that we would not use any additional terms not in CCTS.~~
🟥	~~For the expression of the Representation Term Duration. Type the XSD built in datatype xsd:Duration must~~MUST ~~be used.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~DELETE RULE~~
~~[R 71]~~	~~A period can be expressed using the Aggregate Core Component (ACC) PeriodDetails. The ACC is divided into 3 representation types, Date, Time and DateTime. One of these must~~MUST ~~be selected. Each option has a start and end date, start and end time or start DateTime and end DateTime.~~ ~~.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~Dan Vint: This is even weaker than the BC Duration requirements. Also there is an explanation of the design that I don't think should be in the rule.~~ ~~Here is how I would change this based upon current design, if you agree we should be more specific in the requirements, then may would change to must~~MUST~~.~~ ~~[R 71] A period MAY be expressed using the ACC PeriodDetails.~~ ~~Also we are not consistent in how acronyms are spelled out. BCC used without an explanation but here we spell out ACC. There should be a standard glossary that has all these terms with maybe a pointer to control documentation for the definition.~~ ~~20030715: Rewrite but basically saying the same thing.~~ ~~ACCEPTED:~~ ~~When a Period is expressed using the ACC Period Details, one of the three representation types must~~MUST ~~be either date, time or datetime~~
~~[R 72]~~	~~For each representation term the equivalent data type must~~MUST ~~be used i.e. if the representation term Date is used, then the corresponding built in datatype xsd:date must~~MUST ~~be used.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~Dan Vint: Wouldn't it be better to have a single rule and table that lays out the relation between the representation term and its data type (maybe even the BCC involved)? If we did that there would be one place to look this information up and we could remove the~~

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
				~~need for new rule with the additions of new types and remove rules like this and #68 which seems to be a generalization of this without the specific mapping.~~ ~~If you need something specific to report against, if could be Rule #72-Date or Rule #72-Decimal.~~ ~~20030716: MC: I like the way Dan is thinking, they could be re-written and consolidated. Build a table of xsd datatypes and representations terms.~~ ~~REWRITE~~
~~[R 73]~~	~~The start and end times may be represented by the BCCs i.e. StartTime, EndTime, StartDate, EndDate etc.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~DELETE~~ ~~Duplicate of 75~~
~~[R 74]~~	~~The recurrence of these periods in time may be represented by the BCC RecurrenceValue.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~Similar to 78~~ ~~Dan Vint: I agree this duplicates 74 – I would delete this one.~~
~~[R 75]~~	~~The start and end times may be represented by the BCCs i.e. StartTime, EndTime, StartDate, EndDate etc.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~Duplicate of 73~~ ~~Dan Vint: Agreed it is a duplicate and should be deleted #75.~~ ~~Question: Is this rule getting too specific? Isn't this like saying a decimal will be represented with xsd:decimal and a string with xsd:string?~~
~~[R 76]~~	~~The intervals in a point in time should be represented by a single BCC indicated by the choice operator i.e. FrequencyDuration, FrequencyYear etc.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~Duplicate of 80~~ ~~Tony Coates: I don't see how a point in time can have intervals. I think this needs rewording, at the least.~~ ~~REWRITE~~ ~~This was written by Gunther.~~
~~[R 77]~~	~~Duration may be expressed by the BCC Duration.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~Duplicate of 79~~ ~~Dan Vint: Agreed 79 is a duplicate so delete it.~~ ~~Also is this wording a little weak? If I have a duration I have a duration and it should always be represented with the proper BCC. When or why would you allow something different?~~ ~~20030716: Change may to MUST –~~ ~~This is a consistency thing, they should all either be may or must~~~~MUST~~~~, there shouldn't really be an option here.~~
~~[R 78]~~	~~The number of recurrences may be expressed by the BCC RecurrenceValue.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~DELETE~~ ~~Similar to 74.~~
~~[R 79]~~	~~Duration may be expressed by the BCC Duration.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~DELETE~~ ~~Duplicate of 77~~
~~[R 80]~~	~~The intervals in a point in time should be represented by a single BCC indicated by the choice operator i.e. FrequencyDuration, FrequencyYear etc.~~	~~7/11/03 DUE 7/18/03~~	DELETE	~~DELETE~~ ~~Duplicate of 76~~
[R 81]	~~A UBL message set may be extended where desirable if the business function of the UBL original is retained., but the message exists within its own business context~~. If a UBL message set is extended it MUST	7/14/03 DUE 7/21/03	ACCEPTED	Mike Grimley: Recommend: A UBL message set ~~may~~ MUST only be extended if the business function of the UBL original is retained., and the message exists within its own business context.

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
	retain the business function of the original UBL message set.			
[R 82]	UBL documents ~~must~~MUST use the same legal characters in XML character data that are listed in the XML Recommendation. Including tab, carriage return, line feed, and the legal characters of Unicode and ISO/IEC 10646.	7/14/03 DUE 7/21/03	DELETE	DELETE, already covered in rule 5. HOLD. **Lisa Seaburg**: Similar to rule 5, only documents instead of just names. **Dan Vint**: I would change the rule text: [R 82] UBL documents ~~must~~MUST conform to all the requirements of the most current XML Recommendation. Why are we worried just about these characters. I don't know that this is even required, but if you want to keep something for this rule, then I would make the more general and inclusive rule as above. Otherwise its like, "Oh all I need are these characters I can forget about all the other XML requirements". **20030726**: MC: I agree, we should get rid of the second sentence. **DV:** 2 things going on here, why are we worried about just characters. **MC**: Maybe our guiding principles should be turned into rules, because then that would cover this. **MC**: recommend to put this on hold, lets decide about turning out guiding principles into rules. **Chee-Kai:** Minor editorial amendment suggestion: [R 82] UBL documents ~~must~~MUST use the same set of accepted characters in as those listed in the XML Recommendation, including tab, carriage return, line feed, and the defined characters of Unicode and ISO/IEC 10646. I'm not still happy with my own suggestion, as I think the portion saying "including tab, carriage return, ..." seems to imply that they're not part of character set accepted in XML Recommendation. The primary change here is to avoid the word "legal" in this context, and reserving "legal" to mean the real legality issues surrounding business transactions using UBL. Eduardo Gutentag: I myself am still scratching my head trying to understand why this rule is needed at all. It's like saying "UBL schemas will conform to W3C Schema, and UBL instances will conform to XML 1.0, oh, and UBL instances will use the following characters....which just happen to be prescribed in XML 1.0 but we'll prescribe too". Doesn't make sense. I believe that at some point the idea that what NDR does is draw rules for LSC to follow in the production of its normative output (i.e. schemas) has been forgotten, and has been mixed with all kinds of other things, including

Rule Number	**Rule** ~~(these are not changed)~~	**Sent for email vote**	**STATUS~~LC~~ ~~SC~~**	**Comment** (Look for rewrites and changes to rules)
				rules for instances. THis rule (and others like this) should be deleted.
[R 83]	~~Trading partners may agree on other character encodings to use among themselves. It is recommended in all case that encoding declarations be provided in the XML declarations of UBL documents.~~ [R 83] UBL documents MUST always identify their character encoding with the XML declaration.	7/14/03 DUE 7/21/03	ACCEPTED	Dan Vint: This seems to weak. I would change it to read: [R 83] UBL documents MUST always identify their character encoding with the XML declaration. It is also recommend that for portability that UTF-8 should be used; although trading partners may agree on other character encodings to use among themselves. REWRITE: Wordsmith this into agreement.
[R 84]	~~UBL messages must~~MUST ~~express semantics fully in schemas and not rely merely on well-formedness.~~ [R 84] All UBL messages MUST validate to a corresponding schema. You should not rely merely on well-formedness when defining and building a message.	7/14/03 DUE 7/21/03	ACCEPTED	Dan Vint: What are we really trying to say with this rule? Is that any UBL document should have a matching Schema? Should there also be a statement about validity based upon that schema? I would think something like this is more appropriate: [R 84] Any UBL messages MUST have a corresponding schema and the data stream ~~must~~MUST be valid based upon that schema. You should not rely merely on well-formedness when defining and building a message.. Tony Coates: I don't think Schemas could ever be said to capture semantics. They can capture some of the business rules, but rarely can they capture all of the business rules. I'm not convinced that trying to modify or adjust real business rules just to fit in with what Schemas can do is a good idea. Mike Grimley: Or: ACCEPT REWRITE: [R 84] All UBL messages MUST validate to a corresponding schema. You should not rely merely on well-formedness when defining and building a message.. **Chee-Kai:** Agree with Tony that forcing semantics into schema is odd.
[R 85]	Instances conforming to schemas should be readable and understandable, and should enable reasonably intuitive interactions.	7/14/03 DUE 7/21/03	~~START HERE ON WEDNESDAY~~8/6 - Deleted	HOLD FOR CHAIRS **Dan Vint**: I don't think the Instance has anything to do with the readability and understandability. Seems to me this should be a requirement for the schema design/generation process. This seems more appropriate: [R 85] Schemas should be designed (or generated) such that instances conforming to them are readable and understandable, and should enable reasonably intuitive interactions. **Mike Grimley**: Better, but I'm still wondering if the judgment of such things is too subjective for a rule. **Tony Coates**: My experience is that it isn't *too* subjective. Perhaps more to the point, as a general rule I insist on Schema developers doing prototypical instance documents before they start working on the Schemas, because it leads to more usable document structures. Few people can properly visualise what the instance documents will look like from a knowledge of

Rule Num ber	**Rule** ~~(these are not changed)~~	**Sent for email vote**	**STATUS**~~LC SC~~	**Comment** (Look for rewrites and changes to rules)
				the Schema, and this leads to unnecessarily awkward structures in instance documents. Since UBL has tended to focus on Schemas rather than instances until recently (from what I understand), such issues are going to exist (and Stephen in LCSC has been ferretting them out of late). So, I think something like this is needed. If not a rule, it has to be part of the process. While there are no absolutes as to when you have it "completely right", you can certainly aim to get rid of obvious things that almost everybody dislikes once they see the resulting instance documents. **Dan Vint**: True, but at least this is placing the requirement in the right place. I don't know of anything I can do in the instance to fix a problem caused in the schema. Also this is a should be, so I thin that handles the subjectiveness of the rule. I could also see deleting it if folks found it really objectionable. I don't think that it is a bad design principal. It sort of goes along with the idea of terseness not being a goal in XML - maybe this is just a friendlier way of saying that. **Chee-Kai:** For the original version, I don't understand here; can instances conforming to schemas be unreadable and non-understandable, and could possibly not enable reasonably intuitive interactions? Whether "readability" and "understandability" refer to machine or human, the rule (the original rule) doesn't say more than what is already required (verifiable against a schema). Suggest removing [R 85]. 8/6 – This is more appropriate as a guiding principle.
[R 86]	In the context of a schema, information that expresses correspondences between data elements in different classification schemes ("mappings") may be regarded as metadata. This information should be accessible in the same manner as the rest of the information in the schema.	7/14/03 DUE 7/21/03	8/6 - Deleted	ACCEPTED Bill Burcham: what does classification scheme and mappings mean here? 8/6 – We discussed this at length trying to determine what the origin of the rule is, and what it is trying to convey. General consensus that this should be deleted.
[R 87]	UBL XML Element, attribute, and Simple and Complex Type Names ~~must~~MUST not use acronyms, abbreviations, or other word truncations, except those in the list of exceptions published in Section XX. [Editor's note: Section xx to be a section in the NDR document. Currently this section only includes ID for Identifier, DUNS, and URI.]	7/14/03 DUE 7/21/03	Accepted	See R6, Delete R6, and add attributes to the list of what is named. Dan Vint: This should be combined so there is only one rule #87 [R 87] The only Abbreviations and Acronyms allowed for names used in the schemas or datastreams are listed in Section XX, ie. Element, attribute and Simple and Complex Type Names. Code list values are not controlled by this rule. [Editor's note: Section xx to be a section in the NDR document. Currently this section only includes ID for Identifier, DUNS, and URI.]

Rule Number	**Rule** ~~(these are not changed)~~	**Sent for email vote**	STATUS~~LC~~ ~~SC~~	**Comment** (Look for rewrites and changes to rules)
				Mike Grimley: That doesn't really capture Rule 90. I think two rules are necessary: [R 87] The only Abbreviations and Acronyms allowed in the naming of Element, Attribute and Simple and Complex Types are those contained in the list published in Section XX. [Editor's note: Section xx to be a section in the NDR document. Currently this section only includes ID for Identifier, DUNS, and URI.] [R90] The Abbreviations and Acronyms listed in Section XX MUST be used when referring to (or 'in the place of'?) their corresponding references. 20030716: MC: there needs to be a mechanisim for updating the controlled vocabulary. LS: Industries are not going to spend their time sending in these to some group of people, meaning most industry implementations and extensions will not be compliant. DV: these things maybe just need to be reworded alittle, editor. The other aspect is"Oops we are missing a way to handle other peoples way to handle this issue". We need to expand the section about. Maybe its just one of those things where trading partners need to agree on abbreviations. TC: Could you say the industry has to agree about a list of controlled vocabulary. LS: Maybe this is like our codelists where we don't want to be a controller of their lists, that we need to focus on our own base schema abbreviation and leave it open for outside industries.
[R 88]	Acronyms and abbreviations will only be added to the UBL approved acronym and abbreviation list after careful consideration for maximum understanding and reuse.	7/14/03 DUE 7/21/03	Accepted	See R6.
[R 89]	Acronyms and abbreviations added to the UBL approved list ~~must~~MUST only be taken from the latest version of the Pocket Oxford English Dictionary. The first occurrence listed for a word ~~will be the preferred item to~~ MUST be used.	7/14/03 DUE 7/21/03	Accepted	See R6. Dan Vint: I would change the text: [R 89] When the use of a new acronym or abbreviation is approved for use in UBL documents, the acronyms or abbreviations MUST be taken from the latest version of the Pocket Oxford English Dictionary. If more than one value is provided it should be the first occurrence listed for a word will be the preferred item to be used. Do we need something here to handle possible collisions with existing abbreviations? What about abbreviations that make words that may/may not be used elsewhere? Seems like this rule or the previous one that said they can be added should state some additional requirements (assuming you agree we should avoid these problems) Also what happens if we agree that we want an abbreviation and one is not in the dictionary, should we state a method for creating one, or should it be that we don't use the abbreviation? I have seen a general rule for creation (when they don't exist other wise) to be: "Drop all the vowels from the word."

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
				Tony Coates: Yes, that is the standard rule one uses to make acronyms accessible to native english speakers, and inaccessible to everyone else. 20030716: This is part of the discussion of the rules above. DV: Is there an abbreviation for every word that you could possibly abbreviate and what happens if there isn't one. Lets discuss in Montreal, there may be some pieces of the puzzle we need to go through. 8/6 – slight modification to align rules 87 through 89. Make sure narrative addresses that the latest version of the POED will be used each time an acronym or abbreviation is used.
[R 90]	The Abbreviations and Acronyms listed in Section XX ~~must~~MUST always be used.	7/14/03 DUE 7/21/03	Accepted	See R6 Dan Vint: This should be combined so there is only one rule #87 [R 87] The only Abbreviations and Acronyms allowed for names used in the schemas or datastreams are listed in Section XX, ie. Element, attribute and Simple and Complex Type Names. Code list values are not controlled by this rule. [Editor's note: Section xx to be a section in the NDR document. Currently this section only includes ID for Identifier, DUNS, and URI.] Mike Grimley: That doesn't really capture Rule 90. I think two rules are necessary: [R 87] The only Abbreviations and Acronyms allowed in the naming of Element, Attribute and Simple and Complex Types are those contained in the list published in Section XX. [Editor's note: Section xx to be a section in the NDR document. Currently this section only includes ID for Identifier, DUNS, and URI.] [R90] The Abbreviations and Acronyms listed in Section XX MUST be used when referring to (or 'in the place of'?) their corresponding references.
[R 91]	All types ~~declarations~~ MUST be ~~global~~namedl.	7/14/03 DUE 7/21/03	Accepted	Dan Vint: I would modify this rule to be: [R 91] For reuse and extension, all types MUST be named, which then requires their declarations be globally defined. 20030716: We need to harmonize our anonomous and global. Chee-Kai: Naming does not equate with being global. **Eve Maler**: Actually, I believe naming does equate with being global. A topLevelComplexType requires its name attribute, and a localComplexType prohibits its name attribute from being used. Roughly the same is true for simple types. (See http://www.w3.org/2001/XMLSchema.xsd) So probably the rule could be shortened to:

Rule Num ber	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
				[R 91] All types MUST be named. Explanatory text could then mention that the purpose is for reuse and extension, and that the syntactic consequence of the rule is that the types ~~must~~MUST all be declared as top-level. **Chee-Kai**: Not without loss of generality, let's just look at complexType. A complexType definition is global if and only if it is located as one of the immediate children under <xsd:schema>. A complexType is named if and only if the name attribute exists in the definition. I refer you to XML Schema Part 1: Structures, Section 4.2.2 example "v2.xsd" part of which is illustrated: <xs:redefine schemaLocation="v1.xsd"> <xs:complexType name="personName"> <xs:complexContent> <xs:extension base="personName"> ..... By definition, this <xs:complexType> definition is not global (not necessarily local), AND named. **Dan Vint**: Ah yes it is global. The redefine is working on redefining the global type defined in in the v1.xsd file as personName. Your example is using one of the mechanisms/features that creating global names in the first places allows it to be used. This example would be wrong if personName was not a global in the original schema. **Eve Maler**: Aha, so "top level" not-equal "global". Thanks for the correction. In that case, the rule should not say "...which then requires their declarations be globally defined." If we want to prohibit redefinition, we ideally should do that in a separate rule, and not (incorrectly) imply that naming a type means the definition ~~must~~MUST be directly beneath <xsd:schema>. **Chee-Kai**: Wait, I'm not saying anything about using or prohibiting <xsd:redefinition>, all I'm quoting is that a named complexType can be a non-immediate child of <xsd:schema>, thus giving a counter-example to the assertion that "naming" == "global". The original rule wordings were: [R 91] All type declarations MUST be global. which says what it wants to say already. A rule about redefinition, if there's an intention to do so, would rightly be in a separate rule as you suggested. **Eve Maler:** I checked the XSD spec part 1 (http://www.w3.org/TR/xmlschema-1/), and "global" is a concept that seems to be associated only with element and attribute declarations, and only in the sense that their namespace scope is global. It doesn't refer to the

Rule Num ber	**Rule** ~~(these are not changed)~~	**Sent for email vote**	STATUS~~LC SC~~	**Comment** (Look for rewrites and changes to rules)
				positioning of a declaration or definition within the <xsd:schema> element. So maybe it's best avoided here, since it's not very precise. On the other hand, the XSD schema itself (http://www.w3.org/2001/XMLSchema.xsd) does have a notion of topLevelComplexType versus localComplexType; these are restrictions of the complexType type and govern the definition of named vs. anonymous/locally scoped complex types. There are also topLevelSimpleType and localSimpleType. It looks as though the <redefine> element can contain the version of <complexType> that is bound to topLevelComplexType. So at the very least, maybe we should substitute "top-level" for "global" if we determine that this rule should have that additional bit of explanation on the end. (I know that redefinition was discussed at one time, but don't recall the conclusion. We should make sure that any decision on them is recorded.) **Chee-Kai**: It's funny how when I flipped through the pages of XML Schema Part I, I got reminded of lawyers ploughing through the legal printings... Anyway, thanks for englightening more aspects of XML Schema. I tend to agree with you that XML Schema Part I talks about "global" almost consistently with use of "elements" only. It is remarkably silent about talking about the concept of "global" with types. But I see a point with [R 91] to talk about a type being global because such concept is not only applicable to programming languages, it also makes it easier to talk about types (whether named or unnamed) that are allocated at the "top-level" (immediate children of <xsd:schema>) of a schema. I believe I'm not alone in talking about "global" in the sense of being an immediate child of <xsd:schema>. For elements that you mentioned, XML Schema Part I does have a paragraph saying (somewhere in the midst of Section 3.3.2): "<element>s within <schema> produce global element declarations; <element>s within <group> or <complexType> produce either particles which contain global element declarations (if there's a ref attribute) or local declarations (otherwise)." I concede that the notion of "global" being an immediate child of <xsd:schema> is an indirect conclusion. The sentence uses the notion of "depth" (within <schema>, or within <group> or <complexType> which in turn ~~must~~MUST be within <schema>) to define "global"ness. As a given <element> ~~must~~MUST be in one or the other depth location, the conclusion would be that if a given <element> has depth 1, it is global; else if it occurs within <group> or <complexType>, it ~~must~~MUST have depth > 1 and so isn't global (and not necessarily immediately concluded as being local). Either way, perhaps we need to agree on what "global"

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
				means in the context of [R 91], if there could be a chance of having "global" being interpreted as any concept, such as "naming" or "being defined" (which to me, again, are very different concepts not to be interchanged easily without knowing the implications). 8/6 – Much discussion. Gunther is concerned that non-native speakers might misinterpret. Mark to ensure ndr document defines similiarity between named types and global elements declaration properties.
[R 92]	All element declarations MUST be global with the exception of ID and Code which MUST be local.	7/14/03 DUE 7/21/03	Accepted	**Tony Coates**: I wasn't around for the original local vs. global discussion. However, rules that apply sometimes and not other times are generally a nuisance to support. I would much rather see UBL have either all global elements or all local elements, rather than an arbitrary compromise. 20030716: We need to harmonize our anononous and global. **Chee-Kai**: Agree with Tony with legs up!
[R 93]	XSD: Application Information~~Processing Instructions~~ MUST NOT be used in UBL normative Schema. UBL designed schema SHOULD NOT use XSD:ApplicationInformation. If used, XSD:ApplicationInformation MUST only be used to convey non-normative information.	7/15/03 DUE 7/22/03	Accepted.	**Eve Maler**: In schemas or instances? **Dan Vint**: I've got more of a why question on this one. Eve covered the Schema/Instance use side, but what if I want to use the Stylesheet PI to indicate how my document should be formatted? This is how they set that spec up and I would see where people might want to use that one specifically in the data stream. I would hope the surrounding text gives me a reason why, other than just because. This certainly might be an eaiser way to incorporate some trading partner agreements and still use the off the shelf UBL schema. **Chee-Kai**: I could be a little ignorant on the rationale here. I'm neither strongly for or against this rule. But SOAP folks have gone through 1.1 prohibiting it, then after implementations, have changed their mind to allow for PIs to be in. Should we relook at what new insights have been acquired by them before prohibiting PIs? **Tony Coates:** One thing that I personally haven't understood about the NDR rules. Are they supposed to apply to all users using UBL (and derivatives thereof) at any time, or are they just intended as a guide to groups like LCSC? It currently isn't clear to me whether this rule means that PIs should never be used in UBL instances by any end users, or whether it means that LCSC shouldn't explicitly use PIs in its designs, and hence explicitly force users to use PIs. So, what is it? Thanks, 8/6 – Much discussion for and against. Agreed that we would use alternative wording.
[R 94]	[R94a] Empty elements MUST not be declared [R94b]The XSD built in nillable attribute MUST NOT be used for any UBL declared element.	7/15/03 DUE 7/22/03	8/6 – need to send to list.	See R32 - **Chee-Kai**: Amendment Required because prohibiting xsd:nil does NOT equate with prohibiting empty content element. Suggested change: [R 94]: The nillable attribute MUST NOT be used. Empty content element MUST NOT be

Rule Num ber	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC~~ ~~SC~~	Comment (Look for rewrites and changes to rules)
	[R94c] An element MUST not appear without content in the document instance. [remember to add in narative that there is an issue with string length of 0.]			instantiated UNLESS it is expressly a user-intended indication to instantiate empty content for a given element. **Dan Vint**: This isn't quite right. In schemas only elements of type string can be presented as empty in a data stream without using the NIL attribute. The user indication that nil is the appropriate interpretation is to use this schema attribute (or we have to create our own method). We need a statement more like this: Any element declared to have data, ~~must~~MUST not appear in a data stream as an empty element. Elements declared as EMPTY may only appear in the data stream as an empty element. This rule then prevents the use of the nillable attribute in the schema definition and the corresponding xsi:nil attribute in the data stream. Propose a new rule: Elements that are not declared empty MUST have content. REWRITE and look at 32 to match them up together. These are two rules saying about the same thing. There are many ways to say the same thing. 3 different conditions that need to be covered. Chee-Kai: It depends on what was the intention of the rule. I took Mark's response last time when he mentioned that [R 32] (= [R 94]) already prohibits instantiating empty content (which I didn't think that [R 32] as it stands says that) to mean that [R 94]'s intent is to prohibit instantiating elements with empty content. To complete that insufficiency that I though [R 94] had, I suggested the above additional line, to be interpreted whenever empty content can be instantiated. >>Any element declared to have data, ~~must~~MUST not appear in a data stream as an >>empty element. No, for such situations, during generation, it is an invalid instance already. On the receiving end, this will cause schema validator to flag error based on, for example, a string pattern that contains no empty string or a string restriction with minLength="1" (See XML Schema Part 2 Section 4.3.2). So this doesn't say more than what is already in place. 7/30 – we concur with adding the additional rule. .
[R 95]	~~Wildcards MUST NOT be used.~~ [R95a] The XSD:Any element must not be used The XSD:anyAttribute MUST not be used The XSD:anyType MUST not be used	7/15/03 DUE 7/22/03	8/6 - Accepted	Dan Vint: The wildcard for elements is Any, the wildcard for attributes is anyAttribute and there is a wildcard for the datatype as well anyType and anySimpleType. Because the question was asked, maybe these should all be listed as an example. Suggested change: Wildcards, such as “Any” and

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC~~ ~~SC~~	Comment (Look for rewrites and changes to rules)
	The xsd:anySimpleType Must not be used			“anyAttribute”, MUST NOT be used
[R 96]	Two schemas shall be developed for each standard. One schema shall be a run-time schema devoid of documentation. One schema shall be a fully annotated schema ~~that employs XHTML for the annotations.~~ UBL MUST provide two normative schemas for each transaction. One schema shall be a run-time schema	7/15/03 DUE 7/22/03	8/6 - Accepted	Chee-Kai: Objection : Stating two versions of schemas that are otherwise the same except for documentation is a user-level optimisation/preference issue that shouldn't become a hard-rule from UBL/NDR. This is not to say that we cannot publish as a non-normative add-on, or reading aid or optimized version (assuming documented version is normative). Suggest removing it. Dan Vint: Removing it allows one to do or not do this. I believe the general request is to provide these two forms of the schema and there should be a rule/requirement that states this. Bill Burcham: I think we should remove this rule. A user can filter out annotations if they are unwanted. Dan Vint: Ok then the implication of this is that the only schema that UBL sends is one with full documentation/annotation. I suppose I can live with that, but again I would state this as a rule that the schemas are fully documented. Tony Coates: I am *completely* opposed to the suggestion that users can filter annotations out of their Schemas if required. My experience is that only the 1% of people who take part in XML committees have the confidence and experience for that. Worse, it just introduces an unnecessary, globally-distributed quality control issue. If it is so easy to filter out the documentation, then let us do it just once ourselves for everyone on the planet, and issue two equivalent normative Schemas, full-fat and low-fat. I'm happy to contribute an XSLT script or Java class to do the job, if it is too hard to do from the Schema generation tools. **Dan Vint**: I'm fine with the requirement to send both, but I think it should still be a stated requirement/rule. **Chee-Kai**: Sorry, Tony, I am VERY STRONGLY against having more than ONE normative schema. Taking a step further based on proposed "normative optimized schema without documentation", why can't people ask for "normative optimized schema without documentation AND without whitespaces" (ie, one long line of "optimized" non-fat milk?), and why can't people ask for "normative optimized schema in compressed WBXML binary format", and all sorts of other "normative" versions? Where does that stop?? So, no, please don't open the pandora box. **Chee-Kai:** Precisely! When stated as a rule, it is unnecessarily restrictive for user to accomodate this rule when situation may not call for such. If you're talking about UBL's packaging, please read my second paragraph. **Tony Coates:** There is no Pandora's Box here. People can ask for different varieties of Schema until they go

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
				blue in the face, but all that counts is what UBL decides to produce. It is quite normal for people to require Schemas without annotations for use on production systems. If UBL does not provide one, then everyone has to remove the annotations themselves, and they may get it wrong. The UBL team saves itself a bit of trouble, but the rest of the world has to suffer a distributed quality control problem. Although I understand the orthodoxy behind only wanting a single normative version of any deliverable, I strongly support UBL providing the two flavours of Schema. **Chee-Kai**: So I suppose we don't differ so much here. Is it right to say that your point being: Provide the undocumented version to help others, but ok with having only one normative form (say, the documented one)? I'm certainly not against (not that I have any influence in the final decision) supplying multiple forms within what TC members can provide. We're doing that already anyway with supplying spreadsheets-equivalent, UML-equivalent diagrams, etc. 8/6 – Chair decision. Maintain Rule.
[R 97]	Mixed content MUST NOT be used except where contained in an XSD:documentation element ~~(excluding documentation).~~	7/15/03 DUE 7/22/03	Accepted	See R25 Dan Vint: Hopefully Mixed content is defined somewhere in a glossary, if not please add a description here: An element has mixed content, when it allows both data and additional element content.
[R 98]	Built-in Simple Types SHOULD be used wherever possible.	7/15/03 DUE 7/22/03	Accepted	Dan Vint: Should we be masking the datatypes to protect against any future changes in definitions or other technologies? For instance xsd:boolean allows 0/1 and true/false - do all languages implement boolean this way? Will another XML schema language define boolean this way, or will it add yes/no? I would suggest adopting and specifying the UBL definition of the datatypes (for now they might be exactly what is in schema), but we don't use the schema types directly. So I would defined a ubl:Boolean that restricts xsd:boolean and then use ubl:Boolean everywhere instead of xsd:boolean. I belive in the long run this will be a useful "protection" from technology change.
[R 99]	XSD:Simple~~-~~Type restriction ~~MAY~~ MUST not be used for CCTS:CCTs.	7/15/03 DUE 7/22/03	8/6 accepted	Dan Vint: This rule seems a little strange to me. The implication is that we have gone through all the schema capabilities and have indicated yes/no as to what can be used - is this true? If it is true, then I would recommend one rule that has a table of all the "features" with an indication of their use. It will be easier to find and maintain.
[R 100]	The XSD:Union technique ~~MAY be used to merge data types. UBL: Not applicable. Therefore, SHOULD~~ MUST NOT be used. (Code lists are excluded from this rule.)	7/15/03 DUE 7/22/03	8/6 accepted.	Dan Vint: How about:[R 100] Union MAY be used to create new simple datatypes as long as those types are not enumerated types (code lists).
[R 101]	Complex Types MUST be named.	7/15/03 DUE 7/22/03	8/6 – deleted as duplicative of revised rule 91	
[R 102]	The absence of a construct or data MUST NOT carry meaning.	7/15/03 DUE	8/6 - Accepted	

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	Comment (Look for rewrites and changes to rules)
102]	NOT carry meaning.	7/22/03	Accepted	Tony Coates: I object to this rule on the basis that it seems unprovable. How does one guarantee that the absence of something cannot never imply something meaningful? I suspect that the idea behind this was something like "absence of an element should not be used to imply that the element has a null value", or something like that. Can someone who was around for the creation of this rule comment on what the idea behind it was? Thanks, Jim Wilson: I do seem to recall a discussion around this that is easily understood in the form of an example (totally contrived) that would violate the rule: "The CustomerID element may have the optional attribute customerIDType. If that optional attribute is missing, then the customer ID is a DUNS number." (Yes, I'm aware of W3C XML Schema support for defaults, but that's beside the point and probably not relevant at all given that the rule is not stated in the context of XML Schemas.) Dan Vint: If anything like this exists in the definitions then all the rules about EMPTY data elements would have to be removed and we would have to allow the use of nill to flag this in the data stream. I think I like the rule and would say the schema should be changed to not have an attribute or content be mutually exclusive. I think the design is wrong, not the rule
[R 103]	Substitution groups MUST NOT be used.	7/15/03 DUE 7/22/03	8/6 accepted.	Eve Maler: Does this mean that the normative UBL schemas ~~must~~MUST block element substitution so that customizations can't use it, or that UBL ~~must~~MUST not itself define substitution groups while letting customizations do it, or...? 8/6 – Will not be allowed for schema from LC. May subsequently allow in extension methodology.
[R 104]	Attribute Groups MAY be used.	7/15/03 DUE 7/22/03	8/6 – not required	This rule was superceeded by new rule 26 c and d.
[R 105]	ID/IDREF MUST NOT be used.	7/15/03 DUE 7/22/03		Dan Vint: This seems a little overboard. I guess the corresponding rule is that key/keyref will be used to implement simple ID/IDREF relationships. Has anyone verifed that this is something that the tools support? I know they will manage ID/IDREF for me, but I have doubts about any other mechanism for making relationships.
[R 106]	Key/KeyRef MAY be used.	7/16/03 DUE 7/23/03		
[R 107]	The XSD prefix MUST be used. (xmlns:xsd=http://www.w3.org/2001/xmlSchema)	7/16/03 DUE 7/23/03		Move to part of R1. **Dan Vint**: I'm a little surprised that we have a rule restricting the namespace prefix. I agree UBL should probably use one prefix for consistency, but it doesn't really matter what the actual value is other than making documentation a little easier for all UBL messages. What does it matter if my actual implementation uses something different?

Rule Num ber	**Rule** ~~(these are not changed)~~	**Sent for email vote**	STATUS~~LC~~ ~~SC~~	**Comment** (Look for rewrites and changes to rules)
				Chee-Kai: Just a minor editorial amendment suggestion to use: [R 107] The "xsd" prefix MUST be used. (xmlns:xsd=http://www.w3.org/2001/xmlSchema) because in the original version, if "XSD" is interpreted as an acronym for XML Schema, then which prefix MUST be used isn't stated. On the other hand, if "XSD" IS the prefix that ~~must~~MUST be used, then it should be lowercased. So putting it lowercase with enclosing double-quotes should help to clarify. Note: Same for XSI **Chee-Kai**: Eve's right; XML spec doesn't require a fixed prefix. And specifying a fixed prefix to associate with a given namespace value seems to go against the XML spec's spirit of having prefixes as mere "bins" to hold the namespace values. On the other hand, some implementations do have (questionable) "features" that for various reasons process fixed prefixes (e.g. "xs" or "xsd") or are not that agile in managing namespace dynamics. This situation might improve in time. But having heard Eve's point, I agree with Eve's version, but prepended with: [R 107] If a fixed prefix is used, then the prefix should be "xsd", and all W3C XML Schema constructs in UBL schema modules ~~must~~MUST use the "xsd" namespace prefix, bound to "http://www.w3.org/2001/xmlSchema". 8/1 – The point is we are creating a standard for all UBL schemas. This is not a violation of the XML specs. The purpose is to ensure consistency in the declarations, and as standardizing on the use of "xsd" as a fixed prefix for XSD is quite appropriate. Reworded Chee-Kai's proposal to read: [R107] All W3C XML Schema constructs in UBL schema and schema modules MUST contain the following regular expression: xmlns:xsd="http://www.w3.org/2001/XMLSchema"
[R 108]	The XSI prefix SHALL be used where appropriate.	7/16/03 DUE 7/23/03		**Eve Maler**: The "xsi" prefix is meant for instances, not schemas. So what does this rule mean? XSD governs the required use of xsi:type in instances, and there is a bit more going on in that namespace than just xsi:type, so I'm pretty sure we want to be specific -- e.g., if there are any cases where we might require xsi:type although XSD doesn't (e.g., to reduce reliance on an external schema file for exposing types of elements). However, I doubt we want to do anything more than allow constructs in the xsi namespace to be used as allowed/dictated by the XSD spec, which doesn't seem to need a rule. Again, if we do turn out to need a rule, it should not make the "xsi" prefix not seem like magic.

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC~~ ~~SC~~	Comment (Look for rewrites and changes to rules)
				Dan Vint: Same comment as rule 107, what does it matter what prefix I use? Related to this. We have agreed to require the use of the XML Declaration, should we also require that xsi:SchemaLocation (and maybe other attributes) are required in UBL documents? Other wise I can send you a data stream with no indication of what schema and namespace should apply. **Chee-Kai**: Suggest changing to: [R 108] The "xsi" prefix SHALL be used where appropriate. Reason: Same for "XSD" change in previous post. **Tony Coates**: >[R 108] The XSI prefix SHALL be used where appropriate. Similar to the "xsd" case, if UBL is going to have a rule about the "xsi" prefix, the rule needs to refer to the namespace URI and not just a prefix. 8/1 – same comment as rewritten Rule 107. Rewritten to read: [R108] All UBL conformant instance documents MUST contain the following regular expression: xmlns:xsii="http://www.w3.org/2001/XMLSchema-instance"
[R 109]	Abstract Complex Types MAY be used (for UBL ur-schema)..	7/16/03 DUE 7/23/03		Dan Vint: Abstract is a useful composition tool, do we also want to allow the side effect of being able to substitute elements defined from same types? The example is a Address and UKAddress and USAddress from the Schema Primer. I can use Address in the definition of an element and it allows all three of these to be used. Should we say the final derivations should only be referenced?
[R 110]	(not finalized) Complex Type extension SHOULD be used where appropriate.	7/16/03 DUE 7/23/03		Dan Vint: Is extension also restriction? Also is this for UBL base schemas or really meant for end user customization? I would think we would probably have 2 different types in the UBL schemas, and this is only a customization issue.
[R 111]	(not finalized) The final attribute SHALL be used to control extensions.	7/16/03 DUE 7/23/03		Dan Vint: Does this mean we should decide on everything if extension is allowed (one way of reading the SHALL requirement). I think this should be MAY be used, and then add some guidance as to when it is appropriate.
[R 112]	(not finalized) The block attribute SHALL be used to control extensions.	7/16/03 DUE 7/23/03		Eve Maler: This relates to substitution groups; you block elements that you don't want used as the head of such a group. So this should be coordinated with R 103. (To control complex type extension, you would use "final", not "block".) Dan Vint: Same comment on rule 111. Should ought to be MAY, Also should we be saying something about the schema level default settings for both of these as well as the local usage?
[R 113]	Complex type restriction SHOULD be used.	7/16/03 DUE 7/23/03		

Rule Number	**Rule** ~~(these are not changed)~~	**Sent for email vote**	**STATUS**~~LC SC~~	**Comment** (Look for rewrites and changes to rules)
[R 114]	Notations MUST NOT be used.	7/16/03 DUE 7/23/03		Dan Vint: Is there also a need for a statement about the use of DTD declarations in our schemas for things like entities? Eve Maler: Sigh, it would be most complete to do so -- as XML documents, schema modules could theoretically contain internal/external subsets. Dan Vint: I'm assuming this rule is here because it really requires the use of the DTD to declare the NOTATION in the first place. Eve Maler: The schema "notation" feature is supposedly not even compatible with the DTD <!NOTATION...> feature, so it's best to stay away from it for lots of reasons Mark Crawford: Remember, we already *greed our normative form is XSD, and all of our rules would be focused accordingly Dan Vint: XSD is XML and the XSD folks have even said that the only way to do some things is to combine an Internal subset/DTD in the schema to define entities and notations. This is capability and techniques that are available. The NOTATIONS issue I thought was related to this, I had not heard there was an incompatiblity in the two areas.
[R 115]	All documents shall have a container for metadata and which proceeds the body of the document and is named "Head" ______________. (anything but header)	7/16/03 DUE 7/23/03		How to name the Header: OrderHead, InvoiceHead, etc…… We are provisionally giving this advice to CheeKai to get this into the schemas for the draft versions. **Chee-Kai:** I remain critical of having to maintain such virtual structure for no apparent use. I've heard that the rules don't affect FPSC at all. By design, they should not affect LC. So who's benefiting from carrying all the empty luggages around? That said, I pointed out last time that the [R 115] should have "precedes" instead of "proceeds", unless the proponent of the rule wants Head sitting at the tail. 7/28 – Deleted by joint agreement with LCSC.
[R 116]	All elements with a cardinality of 1..n, (and lack a qualifying structure) ~~must~~MUST be contained by a list container named "(name of repeating element)List", which has a cardinality of 1..1.	7/16/03 DUE 7/23/03		This includes any element and type that has a cardinality of 1..n shall have a container. This rule should be applied at application level it is not part of the business model. List rule takes care of non-header containers **Bill Burcham**: I'm with Chee-Kai -- I think [R 116] is wrong. (I know it's probably too late -- but I'm gonna say my peace anyway :-) The two cases I've heard made in favor of it are: 1. container elements foster more readable stylesheets 2. container elements significantly improve document processing performance Argument 1 is weak. Forgive me for posting working code, but here is an instance document with superfluous

Rule Num ber	**Rule** ~~(these are not changed)~~	**Sent for email vote**	**STATUS~~LC~~ ~~SC~~**	**Comment** (Look for rewrites and changes to rules)
				containers: <?xml version="1.0" encoding="UTF-8"?> <doc> <SuperfluousContainer> <Fruit>Apple</Fruit> <Fruit>Orange</Fruit> <Fruit>Banana</Fruit> </SuperfluousContainer> </doc> And here is a stylesheet to process it: <?xml version="1.0" encoding="UTF-8"?> <xsl:transform version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"> <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/> <xsl:template match="doc"> <xsl:element name="NewDoc"> <xsl:apply-templates select="current()/*"/> </xsl:element> </xsl:template> <xsl:template match="SuperfluousContainer"> <BeforeFruit/> <xsl:apply-templates select="current()/*"/> <AfterFruit/> </xsl:template> <xsl:template match="Fruit"> <AFruit> <xsl:value-of select="text()"/> </AFruit> </xsl:template> </xsl:transform> And here is the output: <?xml version="1.0" encoding="UTF-8"?> <NewDoc> <BeforeFruit/> <AFruit>Apple</AFruit> <AFruit>Orange</AFruit> <AFruit>Banana</AFruit> <AfterFruit/> </NewDoc> The example injects an element before the first fruit and after the last one. That's the example we've been discussing for a couple years as being the bugaboo here. And here is an analogous source instance doc -- this time with no superfluous containers: <?xml version="1.0" encoding="UTF-8"?> <doc> <Fruit>Apple</Fruit> <Fruit>Orange</Fruit> <Fruit>Banana</Fruit> </doc> And here is a different stylesheet to process this one: <?xml version="1.0" encoding="UTF-8"?> <xsl:transform version="1.0"

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC~~ ~~SC~~	Comment (Look for rewrites and changes to rules)
				xmlns:xsl="http://www.w3.org/1999/XSL/Transform"> <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/> <xsl:template match="doc"> <xsl:element name="NewDoc"> <xsl:apply-templates select="current()/*"/> </xsl:element> </xsl:template> <xsl:template match="Fruit"> <xsl:if test="position() = 1"> <BeforeFruit/> </xsl:if> <AFruit> <xsl:value-of select="text()"/> </AFruit> <xsl:if test="position() = last()"> <AfterFruit/> </xsl:if> </xsl:template> </xsl:transform> Comparing the two stylesheets I note that the one for superfluous containers is 19 lines and the one for repeating elements (with no superfluous containers) is 20 lines. That's only one line of code difference. And I don't think the second stylesheet is any less readable than the first. If I look at the two source documents, and extrapolate to larger documents with more nesting I can say with certainty that superfluous containers make for larger documents and IMHO are a bit harder for humans to read -- do to the increase in indentation necessitated by the deeper hierarchy. As for point 2 (processing performance), that's just Voodoo Computer Science. So, which XML processing tools are we using for comparison? Which versions of those tools? What is the use-case/scenario/algorithm? How big is the document? Worst-case, if you tell me that the document is HUGE then I'll tell you a) the Bolivian rug-weaver using Perl as the processing tool isn't gonna see the HUGE document and b) the company (Wal*Mart) that sees the HUGE document can darn-well write a transform on the incoming document (or four or five transforms) that make it more amenable to efficient processing. But you know what -- I still haven't seen any real _evidence_ that superfluous containers provide any processing performance advantage in the first place. It's more likely they hurt performance since they _definitely_ make documents larger! So by my count, it's: Superfluous containers: they make documents bigger (inflicting a processing burden) and harder for humans to read Repeated elements (no superfluous containers): they make documents smaller and easier for humans to read, and necessitate a tiny bit more XSLT code in some situations. Down with [R 116]! **Eduardo Gutentag**: Bill, I think your argument is bogus.

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC~~ ~~SC~~	**Comment** (Look for rewrites and changes to rules)
				The alternative to (see below)

The alternative to

```
<?xml version="1.0" encoding="UTF-8"?>
<doc>
<SuperfluousContainer>
<Fruit>Apple</Fruit>
<Fruit>Orange</Fruit>
<Fruit>Banana</Fruit>
</SuperfluousContainer>
</doc>
```

is not, in real life,

```
<?xml version="1.0" encoding="UTF-8"?>
<doc>
<Fruit>Apple</Fruit>
<Fruit>Orange</Fruit>
<Fruit>Banana</Fruit>
</doc>
```

but more probably

```
<?xml version="1.0" encoding="UTF-8"?>
<doc>
<someelement>foo</somelement>
<Fruit>Apple</Fruit>
<anotherone>bar</anotherone>
<Fruit>Orange</Fruit>
<alongcontainerlikeaddress>
              <a>
                 <b>
                    <c>foo</c>
                 </b>
               </a>
         </alongcontainerlikeaddress>
<Fruit>Banana</Fruit>
</doc>
```

Also, although I don't have the time or the inclination of checking this out,
(I am on vacation after all) I believe your first stylesheet is way more
complicated than needed for dealing with the container case, I believe it
can be cut in half -- but again, I have not checked this, it's just based
on previous experience with stylesheets.

Jim Wilson: I don't have a vote but I'll throw in an opinion.

First of all, great discussion. I think Bill's analysis is right on.That said, I still like container elements (key word "like"). I feel that instance documents are slightly more intuitive and stylesheets are more intuitive (key word "feel"). Given that "what Jim likes" is not known to be a benefit to anyone but Jim, I certainly couldn't argue against reversing the rule. I hope you don't though.

Bill Burcham: Your counterexample, Eduardo, is double-bogus since in the first place your two docs carry different content, and in the second, your second doc won't validate under any scheme I've heard proposed in UBL, since the (repeated) Fruit elements have elements

Rule Num ber	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC SC~~	**Comment** (Look for rewrites and changes to rules)

				stuck between 'em.
				
				Would this pair have made my example stronger:
				
				`<?xml version="1.0" encoding="UTF-8"?>`
				` <Groceries>`
				`<SuperfluousFruitContainer>`
				`<Fruit>Apple</Fruit>`
				`<Fruit>Orange</Fruit>`
				`<Fruit>Banana</Fruit>`
				`</SuperfluousFruitContainer>`
				`      <SuperfluousVegetableContainer>`
				`            <Vegetable>Celery</Vegetable>`
				`            <Vegetable>Lettuce</Vegetable>`
				`      </SuperfluousVegetableContainer>`
				` </Groceries>`
				
				`<?xml version="1.0" encoding="UTF-8"?>`
				` <Groceries>`
				`<Fruit>Apple</Fruit>`
				`<Fruit>Orange</Fruit>`
				`<Fruit>Banana</Fruit>`
				`      <Vegetable>Celery</Vegetable>`
				`      <Vegetable>Lettuce</Vegetable>`
				` </Groceries>`
				
				All my previous arguments hold equally for these two as well. And you may be right -- that the first stylesheet could be cut in half. And I suspect that if you found such a transformation, you could pretty much apply it to the second stylesheet and cut it in half too. There just isn't much difference between the two approaches when it comes to XSLT.
				
				Chee-Kai Chin: Eduardo, I really think your counter-example is out of point to say the least.
				
				Bill's container illustated the context under which the container's proponent wishes to argue for having containers, ie to wrap those with cardinalities 1..n (and then forgetting to talk about 0..n again) with a TYPED container. This is exactly what Bill tried to illustrate with <SuperfluousContainer>.
				
				Your counter-example illustrates exactly the kind of weakness latent within the containers as proposed by the 4 rules. In the light of having contextualization, a lot more such "holes within cheese" will get generated, leaving implementors taking their own interpretation to handle them. The burden of handling those complex open issues, unfortunately, would not be suffered by the original proponents of these rules.
				
				[R 115] and [R 116] ought to be tarred, compressed and piped into /dev/null
				
				Eduardo Gutentag: Rather than continue this discussion now I propose that it be talked about in Montreal while people are speaking face to face, and if not resolved by then then I can devote some time after the first week of August.
				
				Tony Coates: I may not correctly understand what you mean here, but if you mean that Arofan, Stephen, and myself forgot to mention the 0..n case, we did cover it,

Rule Number	Rule ~~(these are not changed)~~	Sent for email vote	STATUS~~LC~~ ~~SC~~	Comment (Look for rewrites and changes to rules)
				although perhaps the wording of the proposal obscures things a bit. Just in case, I'll summarise the proposal. Suppose that the <Thing> element has cardinality M..N. Then 1. If N <= 1 (i.e. <Thing> is 0..1 or 1..1), then <Thing> does not have a container. 2. If N >= 2 (e.g. 0..2, 1..2, 2..2, 0..3, 1..3, ...) then <Thing> has a <ThingList> container. 3. If M = 0, the cardinality of <Thing> inside <ThingList> is 1..N, and the cardinality of <ThingList> is 0..1. 4. If M >= 1, the cardinality of <Thing> inside <ThingList> is M..N, and the cardinality of <ThingList> is 1..1. Chee-Kai: I ~~must~~MUST have been utterly confused with the 4-rules again, and didn't mean to imply that Arofan, Stephen nor yourself had missed out on things. These if-thens are much clearer, Tony. They handle clearly case-by-case and illustrated with examples. These should've been in the 4-rules in the first place. There was a part in the 4-rules that mentioned about containers being optional for 0..n. I'm sorry for being sloppy here as I can't dig back to point out precisely which of the 4-rules said that, but just recall from memory. The part that's not clear is that when a container for 0..n is said to be "optional", it means (or at least one interpretation can mean) when it is time to instantiate, it is optionally up to the creator to instantiate the container or not. Something like your third if-then hasn't yet said, for an example case of Thing being 0..2, whether <ThingList> should be 0 or should be 1. if-then-2 says <ThingList> should exist. if-then-3 says <ThingList> can be 0 or 1. I'm sure the answer is clear in your minds, but the rules haven't made that undisputably clear. Do the if-thens apply in order? Anyway, your if-thens still help to clear up the more fuzzy areas. Great works for the 3 of you to clarify them. And I'm still critical of even having Tops and Containers. 7/28 – Rewritten.

We need some basic context, extension, and restriction rules.

Mavis' table of decisions made from emails

http://lists.oasis-open.org/archives/ubl-ndrsc/200111/msg00068.html	New terminology to be added to the document: Well-formedness checking: Basic XML 1.0 adherence. DTD validation: Adherence to an XML 1.0 DTD. Schema validation: Adherence to an XSD schema.

	Schema processing: Schema validation checking plus provision of default values and provision of new infoset properties. Ad hoc schema processing: Doing partial schema processing, but not with official schema validator software; e.g., reading through schema to get the default values out of it. Instance constraint checking: Additional validation checking of an instance, beyond what XSD makes available, that relies only on constraints describable in terms of the instance and not additional business knowledge; e.g., checking co-occurrence constraints across elements and attributes. Such constraints might be able to be described in terms of Schematron. Application-level validation: Adherence to business requirements, such as valid account numbers.
http://lists.oasis-open.org/archives/ubl-ndrsc/200111/msg00110.html	Motion: "To use XSD as the source format for UBL business document types." Moved by Arofan and seconded by Dale. Approved by unanimous consent.
http://lists.oasis-open.org/archives/ubl-ndrsc/200201/doc00005.doc	We will not use anonymous types. We will use named types in order to build a proper dictionary that can be referenced. Named types will be top level constructs of the XSD instance. All complex types will be defined together and all simple types will be defined together so that people will know where to look for things. (Approved) To create a usable data dictionary we will document the reusable objects expressed as XSD types in the schema, document the properties of each of these objects expressed as XSD locally declared elements, and document each unique occurrence of each element within each document type. Documentation of unique occurrences of each element within each document type will be sparse but sufficient. Best efforts will be made to auto-generate as much documentation as possible. This documentation will be produced by the UBL TC. (Approved) Type name shall consist of an optional qualifier followed by the object class, followed by the suffix "Type". (Approved) Intermediate level tags (i.e. not top level and not leaf) ~~must~~MUST be comprised of the property term and may be preceded by an appropriate qualifier term as necessary to create semantic clarity at that level. The object class may be used as a qualifier. Mark Crawford has abstained and there were no further objections. (Approved) If elements share the same name they ~~must~~MUST share the same type. If they can't share a type because they are different structurally they ~~must~~MUST have different names except in the following cases. The ones currently mandated are fields containing status codes, purpose codes, action codes. (Still under discussion; add to issues list.) The initial list of representation terms shall be taken from the approved list of ebXML core component representation terms. The NDR SC proposes to be the owner of the UBL representation term list and shall liaise with UN CEFACT with regard to any changes made. (Approved) The representation term ~~must~~MUST appear on leaf elements with the following qualifications: (a) ID ~~must~~MUST be used as the substitution tokenfor the representation term Identifier. (b) The representation term "Text" will be considered the default representation term when a representation term does not appear.

	(Approved) UpperCamelCase ~~must~~MUST be used for element and type names and lowerCamelCase ~~must~~MUST be used for attribute names. (Approved)
http://lists.oasis-open.org/archives/ubl-ndrsc/200203/msg00028.html	MOTION: In those cases where it seems beneficial to have two elements that have the same tag name but are bound to different types, as is currently the case with the BIE Order.Header.Details (tag name Header), it is permissible. Motion passes with Mark C. objecting.
http://lists.oasis-open.org/archives/ubl-ndrsc/200203/msg00028.html	MOTION: Ratify the one-doctype- per-transmission principle as stated in the UBL Planning report and the modnamver paper. (Attending on 22 Mar '02): Eve, Fabrice, Mavis, Bill, Gunther, Phil, Paul, Arofan, Eduardo.) Motion passes unanimously.
http://lists.oasis-open.org/archives/ubl-ndrsc/200204/msg00069.html	We formally accepted the proposal to use elements for everything, except for using attributes for supplementary components.
http://lists.oasis-open.org/archives/ubl-ndrsc/200205/msg00011.html	Code lists We voted on accepting the "namespaced type hybrid method". Accepted with one abstention from Jessica. We agreed that the instance extension method should still be described as a (failed) contender.
http://lists.oasis-open.org/archives/ubl-ndrsc/200205/msg00011.html	Separate RT/CCT module There seems to be some interest in breaking down the UBL "core" into multiple core-ish files, for both memory management reasons (the C1 folks experimenting with Xerces report this) and for reasons of reusing only the parts one wants (some verticals seem to want to reuse the built-in ebXML CCT semantics in a neat package). There's a question about whether such a low-level module needs its own namespace, but it needs one if you are worried about memory management. There's also a question about what we would call this module: Is "CCT" incorrect, given our comments on CCTS? "Leafy things" is too informal. :-) They are sort of "built-in UBL types"; would this be a good name? But other UBL types will be built in to UBL too, by definition. We agreed on "common [UBL] leaf types" (CLTs or CULTs!) for the CCT-ish (basic) stuff, and "[UBL] common aggregate types" (CATs) for the aggregate stuff.
http://lists.oasis-open.org/archives/ubl-ndrsc/200205/msg00018.html	Modnamver Bill moved that we add a recommendation allowing two or more functional areas to share definitions common between them but not used elsewhere by creating and importing an additional RootSchema, where the criterion for creation of this additional level of namespace is that it not be used in a majority of the functional areas. Motion PASSES unanimously. (This means that we've essentially accepted Option 4.)
http://lists.oasis-open.org/archives/ubl-ndrsc/200205/msg00033.html	Code Lists We agreed that it's too circular and inconvenient to require external code list modules to use our simple types for supplementary components, so we agreed that instead they ~~must~~MUST use the attribute names that we dictate in order for us to know that they intended the CCTS semantics. We didn't agree yet on whether to recommend XSD documentation

	elements in external code list modules. We will leave this point open in the NDR document.
http://lists.oasis-open.org/archives/ubl-ndrsc/200207/msg00011.html	Nested Supplementary Components We will definitely have attributes that apply to other attributes. The worst case is codes (other than Language.Code, since the code list for that will be fixed) that are supplementary to real BIEs. - We believe that the names for these second-order attributes can be constructed automatically by applying a qualifier consisting of the name of the first-order attribute to which they apply. - However, we think the definition of the relevant XSD types could be tricky, because those additional attributes need to be specific to particular attributes defined on particular complex types.
http://lists.oasis-open.org/archives/ubl-ndrsc/200208/msg00011.html	Xsd:documentation we agreed that using xsd:documentation is most appropriate. This doesn't preclude anyone from using our documentation fields for further processing, and leaves appinfo free for (e.g.) Schematron business rules.
http://lists.oasis-open.org/archives/ubl-ndrsc/200302/msg00013.html	Our NDR document should be equally clear on which XSD features UBL schemas do and do not employ.
http://lists.oasis-open.org/archives/ubl-ndrsc/200302/msg00042.html	Rules For CCTs – 1) CCTs will be declared as elements 2) CCT Content Components will be conveyed as the value 3) CCT Supplementary Components will be conveyed as the attribute of the CCT element 4) If you use the default codeListVersionID, then you do not have to convey codeListVersionID. If however you use a different codeListVersionID, then you ~~must~~MUST convey codeListVersionID. 5)Any change to cardinality or length for any code for any CCT will only be allowed as derivations from the Ur Schema. ~~5)~~6)Binary objects will not be carried as a value for the declared element, but will be referred to through the supplementary component attribute. The element will be declared as empty.
	The following principles underpinned by Bill's document on Modularity, Namespace and Versioning have been voted upon and agreed. These principles and the prose of this document v8 will provide the basis for the rules in the NDR document. 1. UBL namespace names shall include version identifiers. 2. The version identifier that is used in the namespace name has two parts, a major number and a minor number. The major number is incremented whenever it contains any incompatible changes. The minor number is incremented with any other type of changes. 3. UBL is composed of a number of namespaces each of which has its own namespace name and, possibly and in practice, its own version identifiers. There is no one to one correspondence between the various namespace versions that make up a UBL release. 3. Once a namespsace and its associated namespace space name are published they shall not change. 5. XSD import function will be used. In all cases a minor version imports the immediately preceding minor version of the same major release.
http://lists.oasis-open.org/archives/ubl-ndrsc/200303/msg00047.html	Code Lists PROPOSED RULE: Where a code list producer has not created a conforming code list schema module, the UBL library ~~must~~MUST bind the code property to the generic code type found in the CCT module. Accepted rule. Proposed RULE: For release of 1.0 of the Code List rules we will mandate a simpleType for the CodeContentType. We will examine in future versions of the Code Lists

	rules, guidelines for using XML for expressing hierarchy in code values. Accepted. Proposed Design RULE: The NDR SC agrees to remove the codename supplementary component from our recommendations for code markup . HOwver, we recommend that for codelist schema modules chosing to do so, they may provide code expansions and definitions in an annotation element inside each enumeration element wher any natural language information should be conveyed by means of xml:lang. Accepted Design RULE: The NDR SC agrees not to use XLINK for supplementary components of code tha t involve URIs but rather to use the XSD:anyURI and to name those attributes according to our usual naming rules. Agreed.
http://lists.oasis-open.org/archives/ubl-ndrsc/200303/msg00047.html	Embedded Documentation Proposed Design Principle It is the intention of the NDR SC to use XHTML Basic as proposed in the NDR document for the purposes of documenting information other than CCTS that already has a structure. This has been voted on and agreed during this meeting which has quorum.

Agenda Items for Montreal

Containers

Redefine

I have heard this - from the London meeting right? I believe Eduardo is still planning on writing up the results of that and coming up with the "alternate implementation" which I don't think exists. Until we have a documented way of doing extensions/restrictions that works, I don't think you want to make any final determination on the use of redefine.

I'm still waiting for the notes on the final argument and solution. The one I heard is that redefine doesn't create a new namespace, if that is the only reason/problem, there is a workaround for it. Not very pretty but a workaround. This area of extension/restriction is not very pretty/complete to begin with and I thin that baby has been thrown out with the bath water on this decision.

List feature