Possibly missing constraints

However, a «Namespace» package may not be contained, directly or indirectly, in any other «Namespace» package.

The local vocabulary is defined as a stereotype of enumeration where each enumeration literal is stereotyped as a local term.

A «NIEMType» class may also be the client of a realization stereotyped as «Restriction», whose supplier is another «NIEMType» class of the same kind, the base type for the restricted type. In this case, the client class may list a subset of the attributes of the supplier class. Any attributes not so listed must have a multiplicity lower bound of 0 in the supplier class.

A «NIEMType» class may be the special class in at most one generalization, the general class of which must also represent a NIEM type.

A «NIEMType» class may be the client of at most one «Restriction» realization, and it may not be both the client of a «Restriction» realization and the special class in a generalization.

The «NIEMType» class may represent a Complex Type with Simple Content (CSC). In this case, the class may be the client of at most one «XSDSimpleContent» realization. The supplier of the «XSDSimpleContent» is a DataType. The class must not be both the special class in a generalization and the client of an «XSDSimpleContent» realization. If the class is the client of a «Restriction» realization, then the supplier DataType defines the constraining facets of the Complex Type’s xsd:restriction. If the class is not a client of a «Restriction» realization, then the supplier DataType is the base of the Complex Type’s xsd:extension.

In a PSM, a role-of property is identified by having a naming beginning with “RoleOf”. Such a property must have aggregation=none. [Maybe this constraint no longer applies in which case it should be deleted from the text and the documentation of RoleOf].

In a PIM, rather than using a «Restriction» realization, a data type that has the «ValueRestriction» stereotype applied may, equivalently, have a generalization relationship to the UML data type representing the simple type being restricted. A data type in a PIM that is *not* stereotyped as a «ValueRestriction» may still be the special type in a generalization. However, this is actually mapped to the PSM as a complex type. If the general type is a pre-defined primitive type or a «ValueRestriction» data type, then this complex type has simple content (see Subclause 7.3.2.2), otherwise it has complex content. Such a specialized data type may not be the general type for any «ValueRestriction» data type.

A union is represented as a UML data type (that is neither a primitive type nor an enumeration).

A list is represented as a UML data type (that is neither a primitive type nor an enumeration).

To resolve this difference, more than one «XSDProperty» property with the same name contained (directly or indirectly) within the same «Namespace» package (see Subclause 7.2.1) shall have the same attribute or element declaration, and the same type if any (and so must all have the same value for kind). All of the attribute uses or element particles mapped from such properties reference the same attribute or element declaration.

A UML property in a PSM that subsets another property must not have the stereotype «XSDProperty» applied with kind=attribute or have the «XSDAnyProperty» stereotype applied. It may not subset an «XSDAnyProperty».

A UML property in a PSM that is a derived union must have the «XSDProperty» applied with kind=element.

A «Choice» class must have at least one property.

A subsetted model is represented by a «Subsets» Realization from the (client) subsetted model to the (supplier) base model. Note that for backwards compatibility a «References» Realization may also be used between information models. A subsetted model is subject to the following constraints:

* The namespace must be the same as the base model.
* If the base model is a reference model, then the subsetted model must be a subset model. Otherwise, the subsetted model must be the same kind as the base model (i.e, a subset model or an extension model).
* The subset model may only include components which are defined in the base model.
* Any simple or complex restrictions in the subset model must be the same or more restrictive than those defined in the base model.
* Any “business-rules” defined in the subset model must be the same or more restrictive than those defined in the base model.
* Any abstract component in the base model must be abstract in the subset model. A concrete component in the base model may be declared abstract in the subset model.
* A Property which is not nillable in the base model must not be nillable in the subset model. A Property which is nillable in the base model may be declared not nillable in the subset model.
* Complex types within a subset model must conform with the base model. This includes:
  + A property may be removed only if the base property has a cardinality range which includes 0.
  + A subset property cardinality must be within the inclusive range of the base property cardinality.
  + Ordering of subset elements must be the same as the base element order.
  + The NIEM name/namespace of included properties must be identical between subset and base models.
  + The type of properties in a subset model must be the same type, or subtype, as the corresponding property in the reference model. Note that for a provisioned subset schema, the type of a property in a subset schema must be the same as the type of a property in a reference schema. If a subset schema is modeled with a property having a subtype of the reference model property type, then it implicitly requires provisioning of constraint schemas to represent the subtype constraint.
  + An element in the base model may have a corresponding subset element which has substitution groups in the subset model. In this case, subject to cardinality and unique particle attribution constraints, a decomposition of the base element may be defined. The decomposition allows for an ordered sequence of substitutable elements to be defined in the subset as a replacement for the single element defined in the base model. Each substitutable element may have its own cardinality bound; the sum of cardinalities must be within the bounds of the base element cardinality. The order and cardinality of the replacement sequence must conform to XML Schema constraints related to unique particle attribution.
  + Constraints on the derivation of a wildcard. A wildcard, subject to cardinality, unique particle attribution, and namespace constraints, may be decomposed in the subset model. The decomposition allows for an ordered sequence of elements to be defined in the subset as a replacement for the single wildcard defined in the base model. Each element may have its own cardinality bound; the sum of cardinalities must be within the bounds of the base element cardinality. The order and cardinality of the replacement sequence must conform to XML Schema constraints related to unique particle attribution. The namespace of each element must conform with namespace constraints specified by the wildcard (if any).

A subsetting model may be represented in the PIM as an «InformationModel» which has a «Subsets» Realization to another «InformationModel» (as supplier of the Realization). The two «InformationModel»s must have the same namespace. If the defaultPurpose of the base model is “reference”, then the subsetting model must have a defaultPurpose of “subsets”, otherwise the defaultPurpose must be the same for both models.