# OIOXML Naming and Design Rules 3.0

<table>
<thead>
<tr>
<th>Id</th>
<th>Title</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>OIO-1</td>
<td>Reuse of existing OIOXML elements and types</td>
<td>Existing elements or types belonging to the OIOXML Core Components and OIOXML Domain Components classes MUST be reused.</td>
</tr>
<tr>
<td>OIO-2</td>
<td>Reuse of existing OIOXML elements and types</td>
<td>Existing elements or types belonging to the OIOXML NDR Components class SHOULD be reused.</td>
</tr>
<tr>
<td>OIO-3</td>
<td>Reuse of an element in preference of its type</td>
<td>An element MUST be reused in preference to its type if the use of the element in a specific context is unambiguous.</td>
</tr>
<tr>
<td>OIO-4</td>
<td>Partitioning schema modules</td>
<td>Each element, and optionally its respective type, belonging to the OIOXML Core Components or OIOXML Domain Components classes MUST be placed in an individual self-contained schema module.</td>
</tr>
<tr>
<td>OIO-5</td>
<td>Partitioning schema modules</td>
<td>Each element, and optionally its respective type, belonging to the OIOXML NDR Components class SHOULD be placed in an individual self-contained schema module.</td>
</tr>
<tr>
<td>OIO-6</td>
<td>Placement of schema modules</td>
<td>All schema modules MUST be placed in the Infostructurebase.</td>
</tr>
</tbody>
</table>

**GXS: General XML Schema Rules**

<table>
<thead>
<tr>
<th>Id</th>
<th>Title</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>GXS-3</td>
<td>Choice of XML encoding scheme</td>
<td>All XML schemas MUST use UTF-8 as XML encoding scheme.</td>
</tr>
<tr>
<td>GXS-4</td>
<td>Namespace assignment</td>
<td>All XML schemas MUST be assigned a namespace.</td>
</tr>
<tr>
<td>GXS-5</td>
<td>The use of include and import</td>
<td>The include construct MUST be used when referring to a schema module under the same namespace. Otherwise the import construct MUST be used.</td>
</tr>
<tr>
<td>GXS-6</td>
<td>The use of redefine</td>
<td>The redefine construct MUST NOT be used.</td>
</tr>
<tr>
<td>GXS-7</td>
<td>The use of notation</td>
<td>The notation construct MUST NOT be used.</td>
</tr>
<tr>
<td>GXS-8</td>
<td>The use of schemaLocation</td>
<td>All schemaLocation attributtes MUST be specified with an absolute and valid URL that identifies the location of the referred schema module in the Infostructurebase.</td>
</tr>
</tbody>
</table>

**GNR: General Naming Rules**

<table>
<thead>
<tr>
<th>Id</th>
<th>Title</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>GNR-1</td>
<td>Naming in English</td>
<td>Elements, attributtes, and types MUST be named in English according to the Oxford English Dictionary or other relevant technical dictionaries.</td>
</tr>
</tbody>
</table>
Elements, attributes, and types MUST be uniquely named within the namespace of a schema and SHOULD be uniquely named across all namespaces of all approved schemas.

A noun used as part of an element, attribute, or type name MUST be in singular form unless the noun only exist in plural form (e.g. Goods).

Element and attributes MUST be named using UpperCamelCase.

Abbreviations and acronyms SHOULD NOT be used in names.

Names MUST be constructed from verbs, nouns, and adjectives.

Underscore (_), period (.), and hyphen (-) MUST NOT be used in names.

Names of elements and types SHOULD comply with the ObjectPropertyRepresentation scheme as specified in the following subrules.

The Object term of a name MUST describe the data object that the element and its type represents in a given context.

The Property term of a name MUST describe the distinguishing characteristics of the data object identified by the name's Object term.

The Representation term of a name MUST describe the representative category of the element and its type and MUST use one of the terms specified in the list of OIOXML representation terms.

A complex type MUST NOT use the Representation term in its name. The Property term MUST be "Collection" if the type contains exactly one element with 2 or more occurrences. The Property term MUST either be "Structure" or omitted completely if the type contains 2 or more different elements.

If a name uses a phrase in its Property term identical to or synonymous with a phrase in its Representation term, this phrase MUST be removed from the Property term and kept in the Representation term.

The name of a simple or complex type MUST end with the suffix "Type".

An element SHOULD have the same name as its type with the suffix "Type" omitted.

Attributes MUST be named using lowerCamelCase.
FNR-1  Naming of schema module files  The file name of a schema module MUST comply with the naming scheme: `<namespace prefix with capital letters>_<element name>.xsd`.

FNR-2  Naming of metadata files  The file name of a metadata file MUST be the concatenation of corresponding schema module file name and the suffix `.meta.xml`.

---

**GTD: General Type Definition Rules (applies to simple and complex types)**

<table>
<thead>
<tr>
<th>GTD-1</th>
<th>Strong data types</th>
<th>All types MUST be defined as strongly as possible.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GTD-2a</td>
<td>Global type definitions</td>
<td>All simple and complex types belonging to the OIOXML Core Components and OIOXML Domain Components classes MUST be globally defined.</td>
</tr>
<tr>
<td>GTD-2b</td>
<td>Global type definitions</td>
<td>All simple and complex types belonging to the OIOXML NDR Components class SHOULD be globally defined.</td>
</tr>
<tr>
<td>GTD-3</td>
<td>Renaming an existing type</td>
<td>A new simple or complex type MUST NOT be defined identical to an existing simple or complex type.</td>
</tr>
<tr>
<td>GTD-4</td>
<td>The use of anyType and anySimpleType</td>
<td>The built-in ur-types anyType and anySimpleType MUST NOT be used.</td>
</tr>
<tr>
<td>GTD-5</td>
<td>Handling binary content</td>
<td>The built-in simple types anyURI or base64Binary SHOULD be used for handling binary content.</td>
</tr>
<tr>
<td>GTD-6</td>
<td>Abstract types</td>
<td>The abstract attribute MAY be used in simple and complex type definitions.</td>
</tr>
<tr>
<td>GTD-7</td>
<td>Controlling type derivations</td>
<td>Type derivations SHOULD NOT be prevented, i.e. the attributes finalDefault and blockDefault in the root element schema including the attributes block and final in simple and complex type definitions SHOULD NOT be used.</td>
</tr>
</tbody>
</table>

---

**STD: Simple Type Definition Rules (applies only to simple types)**

<table>
<thead>
<tr>
<th>STD-1</th>
<th>The use of list</th>
<th>The list construct MUST NOT be used.</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD-2</td>
<td>The use of union</td>
<td>The union construct MUST NOT be used.</td>
</tr>
<tr>
<td>STD-3</td>
<td>Length of strings</td>
<td>The length of strings SHOULD NOT be limited.</td>
</tr>
<tr>
<td>STD-4</td>
<td>Representation of code lists</td>
<td>Code lists SHOULD be expressed using the enumeration construct.</td>
</tr>
<tr>
<td>STD-5a</td>
<td>Values in enumerations</td>
<td>The value of an enumeration construct SHOULD be expressed using lowercase letters.</td>
</tr>
<tr>
<td>STD-5b</td>
<td>Values in enumerations</td>
<td>The value of an enumeration construct MAY be expressed both in the Danish and English language.</td>
</tr>
<tr>
<td>STD-6</td>
<td>The use of the whitespace facet and related types</td>
<td>The whitespace facet and the built-in simple types token and normalizedString MUST NOT be used.</td>
</tr>
</tbody>
</table>

---

**CTD: Complex Type Definition Rules (applies only to complex types)**

| CTD-1a    | Definition of complex types by aggregation | A complex type SHOULD be defined using either the sequence or choice construct. |
**CTD-1b**  Definition of complex types by aggregation  
A complex type MUST NOT be defined using the *all* construct.

**CTD-2a**  Definition of complex types by derivation  
A complex type MAY be defined using the *extension* construct.

**CTD-2b**  Definition of complex types by derivation  
A complex type MUST NOT be defined using the *restriction* construct.

**CTD-3**  The use of a mixed content model  
The mixed content model SHOULD only be used for document-oriented structures.

**CTD-4**  The use of an empty content model  
The empty content model MUST only be used for document-oriented structures.

**CTD-5**  The use of any  
The *any* construct SHOULD only be used in content models for document-oriented structures.

**CTD-6**  The use of anyAttribute  
The *anyAttribute* construct MUST NOT be used in a content model.

---

**ELD: Element Declaration Rules (applies only to elements)**

**ELD-1a**  Global element declarations  
All elements belonging to the *OIOXML Core Components* and *OIOXML Domain Components* classes MUST be globally declared.

**ELD-1b**  Global element declarations  
All elements belonging to the *OIOXML NDR Components* class SHOULD be globally declared.

**ELD-2**  Namespace for elements  
All elements MUST be assigned a namespace, i.e. the attribute *elementFormDefault* in the root element *schema* MUST be assigned the value "qualified" and the attribute *form* MUST NOT be used in element declarations.

**ELD-3**  The use of substitution groups  
Substitution groups MUST NOT be used in element declarations.

**ELD-4**  The use of nillable  
The attribute *nillable* MUST NOT be used in element declarations.

**ELD-5**  Default values  
The attribute *default* MUST NOT be used in element declarations.

**ELD-6**  Constant values  
The attribute *fixed* MUST NOT be used in element declarations.

---

**ATD: Attribute Declaration Rules (applies only to attributes)**

**ATD-1**  The use of attributes  
Attributes MUST only be used for metadata.

**ATD-2**  Namespace for attributes  
All attributes MUST NOT be assigned a namespace, i.e. the attribute *attributeFormDefault* in the root element *schema* MUST be assigned the value "unqualified" and the attribute *form* MUST NOT be used in attribute declarations.

**ATD-3**  Default values  
The attribute *default* MUST NOT be used in attribute declarations.

**ATD-4**  Constant values  
The attribute *fixed* MUST NOT be used in attribute declarations.

---

**VER: Versioning Rules**

**VER-1**  Namespace versioning  
Versioning of schema modules MUST be specified using namespaces.
VER-2  Locking OIOXML schema modules
A schema module that is approved and accepted in one of the OIOXML classes MUST NOT be changed.

VER-3  Backward compatible changes
An existing namespace MAY be used when implementing backward compatible changes in a schema module.

VER-4  The use of version
The attribute version in the root element schema MAY be used for internal purposes.

NMS: Namespace Rules

NMS-1  Namespace representation
A namespace MUST represent a valid URL in the Infostructurebase and comply with the following naming scheme: http://rep.oio.dk/<internetdomæne>/xml/schemas/<YYYY>/<MM>/<DD>/

NMS-2  Prefix naming in the namespace declaration
A namespace prefix MUST be based on the internet domain name in the corresponding namespace without the ending period (.) and root domain (e.g. without ".dk").

DOC: Documentation Rules

DOC-1  Documentation of schema modules
Metadata MUST be assigned to all schema modules.

DOC-1a  Documentation of schema modules
Schema modules belonging to the OIOXML Core Components and OIOXML Domain Components classes MUST specify the metadata information Title, Description(En) and Description(Dk).

DOC-1b  Documentation of schema modules
Schema modules belonging to the OIOXML NDR Components class MUST specify the metadata information Title and Description(En).

DOC-2  Documentation of types, elements and attributes
All types, elements, and attributes whose metadata information is not described adequately MUST be described using the documentation construct.

DOC-3  The use of appinfo
The appinfo construct MUST NOT be used.

DOC-4  Documentation of code lists
All values in a code list MUST be completely documented either locally or by reference to another document in the Infostructurebase holding a complete description.

DOC-5  Delivery document
A schema delivery MUST include one document that describes the complete delivery.