# ST.96 - ANNEX V

IMPLEMENTATION RULES AND GUIDELINES

Version 0.6

*Proposal presented by the XML4IP Task Force for consideration at the CWS/2*

## Editorial Note

The Implementation Rules and Guidelines (Annex V) is a draft which requires implementation and further test by Offices to refine it as necessary. Subsequently, a final proposal for Annex V will be prepared and submitted for consideration and approval by the Committee on WIPO Standards.

## Table of Contents

# 1. INTRODUCTION

## *1.1 OVERVIEW*

1. WIPO Standard ST.96 provides a collection of XML resources that, for many cases, can be used “as is”. However, Intellectual Property Office (IPO) may need to address the use cases whose requirements are not met by XML Schemas defined in ST.96. This document is intended to guide how IPOs develop their schemas based on ST.96 Schemas and other XML resources.

2. One of the goals of ST.96 Implementation Guidelines is to promote implementation of ST.96 Schemas in a common way by IPOs and maintain a consistent usage of element and attribute names between specific implementations.

## *1.2 SCOPE*

3. The scope of this document is to provide a comprehensive set of rules and guidelines for the customization and implementation of ST.96 Schemas.

## *1.3 HOW TO USE THIS DOCUMENT*

4. This document is intended for use by IPOs which want to implement ST.96 Schemas. This document refers to the main body of ST.96 and its Annexes I, II and III. Therefore, those ST.96 documentations are prerequisite for use this document.

## *1.4 TERMINOLOGY*

5. In this document:

- the term “Customization” refers to the alteration of ST.96 Schemas in order to better fit specific requirements;
- the term “ST.96 compatible schema” refers to a schema consistent with ST.96 Schema Components and XML Design Rules and Conventions for Industrial Property (DRCs), i.e., Annex I of ST.96;
- the term “ST.96 conformant schema” refers to a compatible schema that has not been extended and that sustains constraints expressed by an ST.96 Schema.

## *1.5 RULE IDENTIFIERS*

6. All office implementation rules are informative. Office Implementation rules are identified through a prefix of [OI nn]. The value “nn” indicates the sequential number of the rule. For example, the rule identifier [OI-06] identifies the 6th rule of Office Implementation rules.

## *1.6 OVERVIEW OF CUSTOMIZATION METHODOLOGY*

7. The ST.96 Component Schemas have been expressed in W3C XML Schema (XSD), based on the ST.96 DRCs. The ST.96 Component Schemas are used to both specify and validate ST.96 conformance. The steps involved in ST.96 development are shown in Figure 1 below.

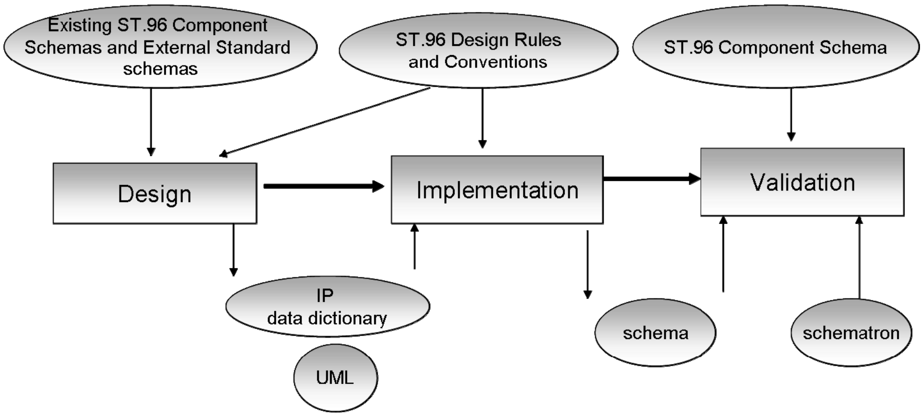
Existing ST.96 Component Schemas and External Standard schemas
ST.96 Design Rules and Conventions
ST.96 Component Schema
Design
Implementation
Validation
IP data dictionary
UML
schema
schematron

Figure1. Overview of ST.96 development methodology

It is recommended to take a similar approach when IPOs customize ST.96 Schemas for their implementation schemas.

## 2. DESIGNING FOR ST.96 CUSTOMIZATION

8. The design of the conceptual models for ST.96 and its customizations is not affected by the syntactical issues of XML, schema languages, or validation tools. Once the need to customize ST.96 Schema has been determined, IPOs SHOULD decide whether the result will be ST.96 *conformant* or ST.96 *compatible*.

9. Designing a customization may involve:

- Adding components to meet requirements of a specific business context;
- Omitting optional components not needed in a specific context;
- Refining the meaning of components;
- Creating constraints on possible values for basic components such as code lists;
- Combining (or recombining) and assembling components into new aggregate components or document components; and
- Adding business rules if an IPO needs further validation using Schematron.

### *2.1 DESIGNING FOR CONFORMANCE*

10. ST.96 conformance at the instance and schema level means that there are no constraint violations when validating the instance against an ST.96 Schema. When designing XML schema for ST.96 conformance, the key objective is to create custom models that can be used to specify and validate ST.96 conformant instances. An ST.96 conformant instance is an instance that validates against an ST.96 Schema. An ST.96 conformant schema is a schema that validates only ST.96 conformant instances. An ST.96 conformant schema is a subset of an ST.96 Schema.

Consequently, designing for conformance applies primarily to restrictions:

- Subsets of the Schema model — restricting the number of components in a Schema.
- Constraints on content — restricting the possible values a component can have

#### 2.1.1 Subsets of the Schema model

11. The use of subsets allows for the removal from a component of any optional components that are not needed to satisfy the specific business requirements of an implementation.

12. It must be noted that subsetting can only be used to remove optional elements or change cardinality in ways that do not reduce the required minimum number of occurrences or extend the permitted maximum number of occurrences of an element. The resulting range must be a subset of the original range. Thus, where 1 < m < n,

- 0..1 can become 1..1 or 0..0 (but not, for example, 1..2)
- 0..n can become 0..1, 1..m, 1..n, m..n, or 0..0
- 1..n can become 1..1, m..n, or 1..m
- 1..1 cannot be changed

#### 2.1.2 Enumeration list constraints on content

13. Constraining the values for a component to a fixed set is a common customization requirement. For example, “the Currency Code must be expressed using ISO 4217 codes” is a constraint on the possible values for Currency Code in an instance.

[OI-01] Office Implementation Schemas must not use `xsd:list` or `xsd:union` to derive simple Types defined in WIPO Standard ST.96.

#### 2.1.3 Other contraints on content

14. There are other cases in which the treatment of ST.96 instances may require customization in order to limit or restrict content values. For example, "The length of an Address Line cannot exceed 40 characters."

15. Co-occurrence constraints apply when the values of one or more components are affected by the values of one or more other components in the document content. The basis can be the presence or absence of content, or particular values of content. For example, "The Start Date must be earlier than the End Date."

16. There are many business rules a customization may require that constrain the values used in the documents. Some of these constraints cannot be specified easily using schema validation semantics. A useful syntax for the formal assertion of these type of value constraints is Schematron (ISO/IEC 19757-3).

17. Using Schematron, a customization can specify all such assertions in a declarative fashion independent of how the assertions are actually implemented as running code in a validation process.

### *2.2 DESIGNING FOR COMPATIBILITY*

18. While ST.96 conformance cannot be guaranteed, some degree of familiarity can be expected through the re-use of components defined in ST.96 and ST.96 Design Rules and Conventions. Compatibility should be a design objective when creating new components or extending existing ST.96 Components.

19. When designing for compatibility, a key objective is to re-use as much ST.96 Components as possible. Where this is not possible, the guiding principles of ST.96 should be followed.

#### 2.2.1 Re-use of ST.96 Schema Components

20. Re-using ST.96 Schema Components keeps customization as closely aligned with ST.96 as possible and prevents an unnecessary proliferation of Components requiring maintenance. Do not recreate aggregate Components that already exist in ST.96.

#### 2.2.2 Compatible extension of ST.96

21. If re-use of existing ST.96 Components is not feasible, customizers may need to add additional Components to ST.96 Schema to satisfy business requirements. In these situations it is possible to extend ST.96 Components in a compatible manner.

22. *Extension* means to add to, or associate with, existing Components additional information that may be required for a particular context of use. That is, an extension creates a superset of the original Component. It is recommended that such an extension include the original Component as an association from the Component that extends it. For example, `CitedIPDocumentType` defined in ST.96 is an extension of `CitedDocumentType` in ST.96 because it contains additional information required if the cited document is an IP document. Structurally, `CitedIPDocumentType` has an association to `CitedDocumentType`, making `CitedIPDocumentType` a superset of `CitedDocumentType`.

23. Compatible extensions can be implemented in parts of a schema. This allows validation checks to be built into the compatible schema that cannot be enforced in the extension area of a conformant schema.

The compatible schema does not share the same namespace as ST.96 Schemas. It is allowed for OIS to use default namespace. They should import ST.96 Schemas. To avoid confusion with ST.96 Schemas, it is not recommended to redefine them. If redefined, namespace prefix for OIS must follow the two-letter codes in WIPO Standard ST.3.

[OI-03] Compatible Schemas should use the import construct to refer to ST.96 Schemas.

[OI-04] In Compatible Schemas, redefinition of all Types, elements, attributes defined in WIPO Standard ST.96 using `xsd:redefine` construct should be avoided.

[OI-05] Compatible Schemas may use a default namespace for efficient operations

[OI-06] Compatible Schemas must use the two-letter codes defined in WIPO Standard ST.3 for office namespace prefix.

##### *2.2.2.1 Using qualified names*

24. Besides namespace qualification, ST.96 supports the principle of qualifying the property term of a component specified as its Dictionary Entry Name in IP Data Dictionary, i.e., Annex II to ST.96, to indicate that customized component is based on ST.96 Component. For example, "*Address. Geographic Region*" in ST.96 can be qualified for Canadian office as "Address. CA Geographic Region", indicating that the use of the geographic region values is restricted to the context of Canadian provinces. The element name would be `CAGeographicRegion` which uses `GeographicRegionType` defined in ST.96.

[OI-07] Compatible Schemas may use the two-letter codes defined in WIPO Standard ST.3 as qualifier for customized components based on ST.96 Schema.

#### *2.2.2.2 Re-using aggregate Components*

25. The principle applied is that if a required aggregate component has the same structure as a standard ST.96 component, then it should not be a redefinition but a re-use by association. The qualifying terms used to name the new associated component then describe the role it plays. For example, if an `AddressBook` is required for an Applicant, and this uses the normal `AddressBookType` structure, it could be defined as `ApplicantAddressBook`.

26. If the new aggregate component does not have the same structure as a standard ST.96 Component, then the required component MUST have a new name, not a qualified name. If possible, the new aggregate component may then be associated with the ST.96 Component being extended. For example, if an Address has additional components when the address is in Korean, then a new aggregate component called Korean Address would be created. This is not a qualification, but a new name. Ideally this should contain the original Address structure by association plus the new Korean components.

#### *2.2.2.3 New basic components*

27. A customization may require new basic components. These should be based on an existing ST.96 or W3C build-in data type (or a refinement thereof). Note that where the new basic component is included in an aggregate component, it will result in a new aggregate component being defined as well.

28. When establishing a new basic component, it is necessary to associate it with a data type. This is determined by the Representation Term part of component's Dictionary Entry Name. For example, a Korean Address may have an additional component called `Dong. Text`. This new basic information entity would use the standard Text data type.

29. Changing or specializing a component's definition changes the component. Therefore, a new basic component must be defined. For example, in ST.96, `Person. Other Name. Name` is defined as "Person name other than first name, middle name, last name, first last name, second last name, suffix and prefix". If a component is required to specify the Skype name as a specific other name, then a new component (perhaps called `Person. Skype Name. Name`) should be defined.

30. In cases where the required component's representation does not fit an existing data type, a new data type may be required. New data types can be based on either ST.96 data types or W3C built-in data types. The following operations are allowed on ST.96 simple Types and examples of XML schema codes are available in Appendix A:

- Add a character length restriction
- Add/change an enumeration list
- Add a pattern restriction

#### *2.2.2.4 New associations*

31. Aggregate components are included in a schema model by associating them with a parent aggregate. If the required aggregation has the same structure as an existing aggregate, a new association should be created with the existing aggregate (as in 2.2.2.2). This new association represents a new use of the aggregate, so qualifying terms can be used to describe the new role. For example: in ST.96, `AddressBook` is re-used in contexts such as `ClaimantAddressBook ApplicantAddressBook` and `LicenseeAddressBook`. They all share the same structure as `AddressBookType` with the terms "*Claimant*", "*Applicant*", and "*Licensee*" providing the qualification.

32. By re-using the unqualified aggregate (`AddressBook`), the same XML type (`AddressBookType`) will be used for implementation of all these components.

#### *2.2.2.5 New aggregate components*

33. A new aggregate component should be created if the required aggregation component does not exist in ST.96 or is an extension of an existing aggregate component, making it no longer conformant. When creating new aggregate components, there are some general principles to follow:

(a) A new aggregate component may also include the aggregate component being extended, as a child by extension. For example, ST.96 itself follows these principles. In ST.96, `CitedIPDocumentType` is a new aggregate that has a different structure than `CitedDocumentType`. The `CitedDocumentType` structure is re-used by extension in `CitedIPDocumentType`. In addition, `CitedIPDocumentType` also contains additional components. The name `CitedIPDocumentType` is not a qualification of the name `CitedDocumentType`, but an extension to the ST.96 `CitedDocumentType` to create a new aggregate

(b) New aggregate components should attempt to re-use patterns of ST.96 structures where possible.

(c) When defining new aggregate components, different constructs and mechanisms are not allowed in ST.96 Schemas but are tolerated in Office Implementation Schemas. This is the case for the all compositor and substitution groups.

(d) The following operations can be performed to customize an existing aggregate component and examples of XML schema codes are provided in Appendix A:

- Add and/or remove an element;
- Add and/or remove an attribute;
- Rename an element;
- Rename an attribute; and
- Make an element required.

[OI-08] Compatible Schemas may use the all compositor.

[OI-09] Compatible Schemas may use substitution groups

### *2.2.2.6 New document component schemas*

34. Where existing document component schema s defined in ST.96 do not meet requirements, it is necessary to create a new document model. The key steps in assembling new document type structures are:

(a) Select/create the root aggregate for the document type

(b) Assemble the required ST.96 Components (and/or customized extensions), applying cardinality constraints.

(c) For all required associations from these Components assemble the required components (and/or customized extensions), applying cardinality constraints.

(d) Continue step 3 recursively through all required associations.

## 3. IMPLEMENTATION

### *3.1 Customized schemas*

35. At least two scenarios in particular lend themselves to XSD derivations performed on existing types:

- An existing ST.96 Type fits the requirements for the application with modifications supported by XSD derivation. These modifications can include extension (adding new information to an existing type) and/or refinement (restricting the set of information allowed to a subset of that permitted by the existing type).
- No existing ST.96 type is found that can be used as the basis for the new type. Nevertheless, the base library of Components that underlies ST.96 can be used to build up the new type so as to ensure that interoperability is at least possible on the basic Component level.

### 3.1.1 New document schemas

36. XSD schemas are used in ST.96 to express normative document constraints. It is possible to express the same document constraints in other schema languages such as RELAX NG or even by using imperative or declarative assertion languages such as Schematron. Since ST.96 uses XSD for its standard schemas, however, it is assumed in the following that new schemas based on ST.96 will use XSD simply to save labor. If XSD is chosen, new compatible document types should adhere to the ST.96 Design Rules and Conventions, and if other formalisms are chosen, the ST.96 DRC conventions should be followed where possible.

### 3.1.2 Subset schemas

37. Where the requirements are for a pure subset, it is possible to prune a ST.96 schema to create a new, smaller schema defining only the subset required. Because ST.96 relies on a common library of re-usable types, this approach does not support the restriction of selective types based on context. That is, an Address when used in one part of the subset schema cannot have a different restriction from an Address in another part of the document.

38. One approach for producing subset schemas is to work with the ST.96 schemas as input and use the XML comment construct to elide all of the Components not used by the customization. A human reader of the schema specifications can see all of the ST.96 standardized constructs, easily distinguishing those that are in the customization and those that are not.

39. Another approach for producing subset schemas is to work at an abstract model level and to synthesize the schema fragments from scratch from the subset model. This approach implies to have all required elements defined as global. In the following example , the first approach is applied.

```
<xsd:complexType name="PersonNameType">
<xsd:sequence>
                 <xsd:element ref="com:PersonNamePrefix"  minOccurs="0"/>
                 <xsd:element ref="com:FirstName" />
                 <xsd:element ref="com:MiddleName"  minOccurs="0"/>
                 <xsd:element ref="com:LastName" />
                 <!--<xsd:element ref="com:FirstLastName"  minOccurs="0"/> →
                 <!--<xsd:element ref="com:SecondLastName" minOccurs="0"/>→
                 <xsd:element ref="com:PersonNameSuffix" minOccurs="0"/>
                 <xsd:element ref="com:PreferredName"  minOccurs="0"/>
                 <xsd:element ref="com:PersonOtherName" minOccurs="0"/>
           </xsd:sequence>
</xsd:complexType>
```

40. Figure 2 shows a way to organize schema modifications in creating a subset. The customized schema fragments on the left are overlaid onto a copy of the XMLSchema/Common/1 subdirectory from the ST.96 distribution package, replacing the corresponding document schema, aggregates schema, and basics schema. This creates a customization suite of schema fragments representing instances with only those constructs allowed by the customization and not simply all elements allowed by ST.96. Only those original schema fragments that correspond to the changed fragments are replaced, thus preserving all of the schema fragment linkages for those fragments that remain unchanged.

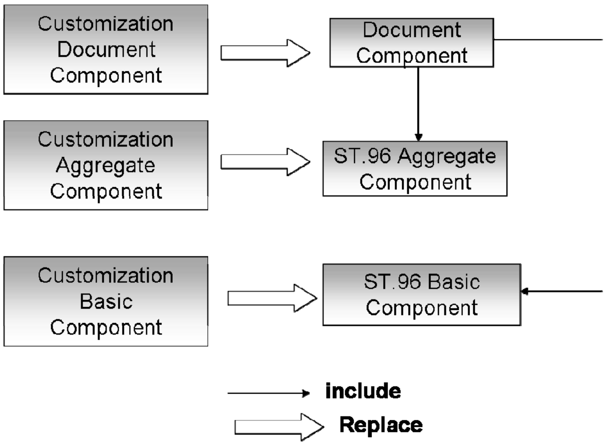
Customization Document Component
Document Component
Customization Aggregate Component
ST.96 Aggregate Component
Customization Basic Component
ST.96 Basic Component
include
Replace

Figure 2. Overlaying customization schema fragments

*3.2 Using the ST.96 components for non-ST.96 document types*

41. Even when a completely new document type must be defined, it can prove advantageous to use as much of the ST.96 Components as possible. Figure 3 shows an approach to specifying the schema fragments defining a non-ST.96L document using both ST.96 and non-ST.96 Components.

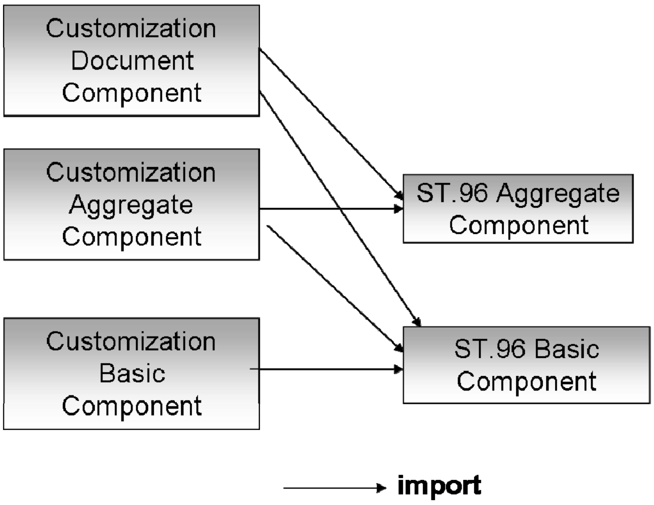
Customization Document Component
Customization Aggregate Component
Customization Basic Component
ST.96 Aggregate Component
ST.96 Basic Component
import

Figure 3. Using the ST.96 Components for non-ST.96 documents

## 4. DEVELOPMENT-STAGE AND PRODUCTION-STAGE SCHEMAS

42. Depending on IPOs, production-stage schema can be different from their development-stage schema. In this case, the design-stage schema, in general, is used to produce the production-stage schema for efficiency purpose. It must be ensured that the XML instances for data exchange must conform to the design-stage schema, which means that the instances should contain the namespace prefixes and schema version.

43. In a design-stage schema, all namespaces should be identified and elements and attributes fully qualified, and all components should have own version information. In the course of designing a schema based on ST.96 Standard, it is important to keep careful track of components using namespaces and version information. For producing production-stage schema, it might be advisable to resolve all external references in a schema in advance and use the fully integrated schema in a production environment to reduce processing time for each instance. Because such a practice is completely dependent on local circumstances, no further recommendation can be given in this document.

## 5. REFERENCES

- UBL 2 Guidelines for Customization, available at: *http://docs.oasis-open.org/ubl/guidelines/UBL2-Customization1.0cs01.pdf*

- Exchange Network shared schema components- usage guide, available at: *http://www.exchangenetwork.net/dev_schema/SharedSchemaComponents-UsageGuide_v2.0.pdf*

- Guidelines for Electronic Text Encoding and Interchange- the TEI Consortium, P5, Using the TEI section, available at: *http://www.tei-c.org/release/doc/tei-p5-doc/en/html/USE.html*

## APPENDIX A

## EXAMPLES OF ST.96 CONFORMANT SCHEMAS

The following examples are listed to show how developers define ST.96 conformant schemas. Designing for conformance applies primarily to restrictions which are:

- Subsets of the Schema model — restricting the number of Components in a Schema; and
- Constraints on content — restricting the possible values a Component can have

### 1. SUBSET OF THE SCHEMA MODEL

### 1.1 Remove an optional element

In order to remove an optional ST.96 element(s), developers SHOULD follow the following steps:

1. Create a new W3C schema file for Type, e.g., XXPersonNameType.xsd
2. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Import ST.96 Components which are needed to define a new Type, e.g., XXPersonNameType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Mi
ddleName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/La
stName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonNameSuffix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pr
eferredName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonOtherName.xsd"/>
```

4. Create the new Type (e.g., XXPersonNameType).which refers to mandatory elements of corresponding Type defined in ST.96 (e.g., FirstName, LastName of PersonNameType), and some optional element(s) of the corresponding Type (e.g., PersonNamePrefix, MiddleName, PersonNameSuffix, PreferredName and PersonOtherName of PersonNameType).

```
<xsd:complexType name="XXPersonNameType">
         <xsd:sequence>
             <xsd:element ref="com:PersonNamePrefix" minOccurs="0"/>
             <xsd:element ref="com:FirstName" />
             <xsd:element ref="com:MiddleName" minOccurs="0"/>
             <xsd:element ref="com:LastName" />
             <xsd:element ref="com:PersonNameSuffix" minOccurs="0"/>
             <xsd:element ref="com:PreferredName" minOccurs="0"/>
             <xsd:element ref="com:PersonOtherName" minOccurs="0"/>
       </xsd:sequence>
</xsd:complexType>
```

5. Create a new W3C schema file for element which has the same file name as the corresponding ST.96 schema file, e.g., PersonName-V1.2.xsd

6. Set the targetNamespace to ST.96 namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.wipo.int/standards/XMLSchema/Common/1"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified"
version="1.2">
```

7. Import the schema file in which the new IPO Type was defined, e.g., XXPersonNameType.xsd

```
<xsd:import namespace="http://www.Organization.com"
schemaLocation="XXPersonNameType.xsd"/>
```

8. Declare the new element based on the corresponding Type above

```
<xsd:element name="PersonName" type="xx:XXPersonNameType"/>
```

## 1.2 Make an element required

In order to make an optional ST.96 element required, developers SHOULD follow the following steps:

1. Create a new W3C schema file, e.g., XXPersonNameType.xsd
2. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Import ST.96 Components which are needed to define a new Type, e.g., XXPersonNameType.

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Mi
ddleName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/La
stName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Se
condLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonNameSuffix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pr
eferredName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonOtherName.xsd"/>
```

4. Create a new Type (e.g., XXPersonNameType).which refers to mandatory elements of corresponding Type defined in ST.96 (e.g., FirstName, LastName of PersonNameType), and some optional element(s) (e.g., FirstLastName and SecondLastName of PersonNameType) which become required.

```
<xsd:complexType name="XXPersonNameType">
         <xsd:sequence>
             <xsd:element ref="com:PersonNamePrefix" minOccurs="0"/>
             <xsd:element ref="com:FirstName" />
             <xsd:element ref="com:MiddleName" minOccurs="0"/>
             <xsd:element ref="com:LastName" />
             <xsd:element ref="com:FirstLastName" />
             <xsd:element ref="com:SecondLastName" />
             <xsd:element ref="com:PersonNameSuffix" minOccurs="0"/>
             <xsd:element ref="com:PreferredName" minOccurs="0"/>
             <xsd:element ref="com:PersonOtherName" minOccurs="0"/>
       </xsd:sequence>
</xsd:complexType>
```

5. Create a new W3C schema file which has the same file name as the corresponding ST.96 schema file, e.g., PersonName-V1.2.xsd

6. Set the targetNamespace to ST.96 namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.wipo.int/standards/XMLSchema/Common/1"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified"
version="1.2">
```

7. Import the schema file in which the new IPO type was defined type, e.g., XXPersonNameType.xsd

```
<xsd:import namespace="http://www.Organization.com"
schemaLocation="XXPersonNameType.xsd"/>
```

8. Declare the new element and the corresponding new type

```
<xsd:element name="PersonName" type="xx:XXPersonNameType"/>
```

### 1.3 Reduce the required maximum number of occurences

1. Create a new W3C schema file, e.g., XXReportCitationBagType .xsd

2. Set the targetNamespace to IPO's namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:pat="http://www.wipo.int/standards/XMLSchema/Patent/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Import ST.96 Components which are needed to define a new Type, e.g., XXReportCitationBagType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Patent/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Patent/1/Aggregat
e/ReportCitationBag.xsd"/>
```

4. Create a new Type (e.g., XXReportCitationBagType) which reduces the maximum number of occurrences (e.g., from "unbounded" to 20) of the ST.96 element defined in the corresponding ST.96 Type.

```
<xsd:complexType name="XXReportCitationBagType">
         <xsd:sequence>
             <xsd:element ref="pat:ReportCitation" maxOccurs="20"/>
       </xsd:sequence>
</xsd:complexType>
```

5. Create a new W3C schema file which has the same file name as the corresponding ST.96 schema file, e.g., ReportCitationBag-V1.2.xsd

6. Set the targetNamespace to ST.96 namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.wipo.int/standards/XMLSchema/Common/1"
xmlns:xx="http://www.Organization.com"
xmlns:pat="http://www.wipo.int/standards/XMLSchema/Patent/1"
elementFormDefault="qualified" attributeFormDefault="qualified"
version="1.2">
```

7. Import the schema file in which the new IPO type was defined type, e.g., XXReportCitationBagType.xsd

```
<xsd:import namespace="http://www.Organization.com"
schemaLocation="XXReportCitationBagType.xsd"/>
```

8. Declare the new element and the corresponding new type

```
<xsd:element name="ReportCitationBag" type="xx:XXReportCitationBagType"/>
```

## 2. ENUMERATION LIST CONSTRAINTS ON CONTENT

In order to add an enumeration list constraint on content, developers SHOULD follow the following steps:

1. Create a new .W3C schema, e.g., XXPersonNamePrefix.xsd
2. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Create an element and its corresponding simple Type, based on the existing ST.96 simple Type or a W3C Built-in Datatype, for the element you wish to create an enumeration for.

```
<xsd:element name="PersonNamePrefix" type="xx:XXPersonNamePrefixType"/>
<xsd:simpleType name="XXPersonNamePrefixType">
<xsd:restriction base="xsd:token">
<xsd:enumeration value="Mrs"/>
<xsd:enumeration value="Mr"/>
<xsd:enumeration value="Ms"/>
<xsd:enumeration value="Dr"/>
</xsd:restriction>
</xsd:simpleType>
```

4. Create a new W3C schema file, e.g., XXPersonNameType.xsd
5. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

6. Include the new schema

```
<xsd:include schemaLocation ="XXPersonNamePrefix.xsd"/>
```

7. Import ST.96 Components which are needed to define a new Type, e.g., XXPersonNameType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/FirstName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/MiddleName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/LastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/FirstLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/SecondLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/PersonNameSuffix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/PreferredName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/PersonOtherName.xsd"/>
```

8. Create a new Type (e.g., XXPersonNameType) which refers to all elements defined in ST.96 corresponding Type (e.g., PersonNameType) and replace the element in which the content is restricted by the new one (e.g., xx:XXPersonNamePrefix).

```
<xsd:complexType name="XXPersonNameType">
         <xsd:sequence>
              <xsd:element ref="xx:XXPersonNamePrefix" minOccurs="0"/>
            <xsd:element ref="com:FirstName"/>
            <xsd:element ref="com:MiddleName" minOccurs="0"/>
            <xsd:element ref="com:LastName" />
            <xsd:element ref="com:FirstLastName" minOccurs="0"/>
            <xsd:element ref="com:SecondLastName" minOccurs="0"/>
            <xsd:element ref="com:PersonNameSuffix" minOccurs="0"/>
            <xsd:element ref="com:PreferredName" minOccurs="0"/>
            <xsd:element ref="com:PersonOtherName" minOccurs="0"/>
       </xsd:sequence>
</xsd:complexType>
```

9. Create a new W3C schema file which has the same file name as the corresponding ST.96 schema file, e.g., PersonName-V1.2.xsd

10. Set the targetNamespace to ST.96 namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.wipo.int/standards/XMLSchema/Common/1"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified"
version="1.2">
```

11. Import the schema file in which the new IPO type was defined  type, e.g., XXPersonNameType.xsd

```
<xsd:import namespace="http://www.Organization.com"
schemaLocation="XXPersonNameType.xsd"/>
```

12. Declare the new element and the corresponding new type

```
<xsd:element name="PersonName" type="xx:XXPersonNameType"/>
```

## 3. OTHER CONTRAINTS ON CONTENT

### 3.1 ADD A PATTERN RESTRICTION

In order to add a pattern restriction on a simple Type, developers SHOULD follow the following steps:

1. Create a new W3C schema file, e.g., XXPhoneNumberContentType.xsd

2. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Create a new simple Type (e.g., XXPhoneNumberContentType), based on the corresponding ST.96 simple Type or a W3C Built-in Datatype, for the element you wish to place a pattern restriction on.

```
<xsd:simpleType name="XXPhoneNumberContentType">
<xsd:restriction base="xsd:string">
<xsd:pattern value="\d{3}-\d{3}-\d{4}"/>
</xsd:restriction>
</xsd:simpleType>
```

4. Create a new W3C schema file, e.g., XXPhoneNumberType.xsd

5. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

6. Include the schema defining the new content (e.g., XXPhoneNumberContentType.xsd)

```
<xsd:include schemaLocation="PhoneNumberContentType.xsd"/>
```

7. Import ST.96 Components which are needed to define a new Type, e.g., XXPhoneNumberType)

```
<xsd: import namespace="www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/ph
oneNumberCategory.xsd"/>
```

8. Add a new Type (e.g., XXPhoneNumberType) which refers to some ST.96 Components (e.g., phoneNumberCategory) and to the new Type (xx:XXPhoneNumberContentType) instead of the type used as a basis in the corresponding ST.96 Type (e.g., xsd:string used in PhoneNumberType).

```
<xsd:complexType name="XXPhoneNumberType">
<xsd:complexContent>
<xsd:extension base="xx:XXPhoneNumberContentType">
<xsd:attribute ref="com:phoneNumberCategory"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
```

9. Create a new W3C schema file which has the same file name as the corresponding ST.96 schema file, e.g., PhoneNumber-V1.2.xsd

10. Set the targetNamespace to ST.96 namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.wipo.int/standards/XMLSchema/Common/1"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified"
version="1.2">
```

11. Import the schema file in which the new IPO type was defined type, e.g., XXPhoneNumberType.xsd

```
<xsd:import namespace="http://www.Organization.com"
schemaLocation="XXPhoneNumberType.xsd"/>
```

12. Declare the new element and the corresponding new type

```
<xsd:element name="PhoneNumber" type="xx:XXPhoneNumberType"/>
```

## APPENDIX B

## EXAMPLES OF ST.96 COMPATIBLE SCHEMAS

The following examples are listed to show how developers can reuse ST.96 Components in their implementation schemas to make them compatible with ST.96 Schemas. Note that targetNamespace for compatible schemas MUST be IPO namespace.

### 1. COMPLEX TYPES SCHEMAS

### 1.1 Make an element required

In order to make an ST.96 element required, developers SHOULD follow the following steps:

1. Create a new W3C schema file, e.g., XXPersonName.xsd
2. Set the targetNamespace to IPO's namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Import ST.96 Components which are needed to define a new Type, e.g., XXPersonNameType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Mi
ddleName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/La
stName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Se
condLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonNameSuffix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pr
eferredName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonOtherName.xsd"/>
```

4. Add a new Type (e.g., XXPersonNameType) which refers to some ST.96 elements using different occurrence indicator(s) than the ones used in the corresponding ST.96 Type (e.g., FirstLastName and SecondLastName of PersonNameType)

```
<xsd:complexType name="XXPersonNameType">
         <xsd:sequence>
             <xsd:element ref="com:PersonNamePrefix" minOccurs="0"/>
             <xsd:element ref="com:FirstName" />
             <xsd:element ref="com:MiddleName" minOccurs="0"/>
             <xsd:element ref="com:LastName" />
             <xsd:element ref="com:FirstLastName" />
             <xsd:element ref="com:SecondLastName" />
             <xsd:element ref="com:PersonNameSuffix" minOccurs="0"/>
             <xsd:element ref="com:PreferredName" minOccurs="0"/>
             <xsd:element ref="com:PersonOtherName" minOccurs="0"/>
       </xsd:sequence>
</xsd:complexType>
```

5. Declare the new element and the corresponding new type

```
<xsd:element name="PersonName" type="xx:XXPersonNameType"/>
```

### 1.2 Change occurrence indicators

In order to change the minimum and maximum occurrences of data elements in a ST.96 Component, developers SHOULD follow the following steps:

1. Create a new W3C schema file, e.g., XXReportCitationBag .xsd
2. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.wipo.int/standards/XMLSchema/Patent/1"
xmlns:pat="http://www.wipo.int/standards/XMLSchema/Patent/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Import ST.96 Components which are needed to define a new Type, e.g., XXReportCitationBagType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Patent/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Patent/1/Aggregat
e/ReportCitationBag.xsd"/>
```

4. Add a new Type (e.g., XXReportCitationBagType) which is a restriction of the corresponding ST.96 complex Type (e.g., pat:ReportCitationBagType) in which the occurrence indicator(s) of the children are modified.

```
<xsd:complexType name="XXReportCitationBagType">
<xsd:complexContent>
<xsd:restriction base="pat:ReportCitationBagType">
         <xsd:sequence>
             <xsd:element ref="pat:ReportCitation" maxOccurs="20"/>
       </xsd:sequence>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
```

5. Declare the new element and the corresponding new type

```
<xsd:element name="XXReportCitationBag"
type="xx:XXReportCitationBagType"/>
```

### 1.3 Add an element

In order to add a new element, developers SHOULD follow the following steps:

1. Create a new W3C schema XXOfficeSpecificElement.xsd

2. Create a new element (e.g., XXOfficeSpecificElement) and a type for your new element (e.g., XXOfficeSpecificType)

```
<xsd:element name="XXOfficeSpecificElement"
type="xx:XXOfficeSpecificType"/>
<xsd:simpleType type="XXOfficeSpecificType">
<xsd:restriction base="xsd:string">
<xsd:maxLength value="2"/>
</xsd:restriction>
</xsd:simpleType>
```

3. Create a new W3C schema file, e.g., XXPersonName.xsd

4. Set the targetNamespace to IPO's namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

5. Import ST.96 Components which are needed to define a new Type, e.g., XXPersonNameType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Aggregat
e/PersonName.xsd"/>
```

6. Include the schema that defines the new element (e.g., XXOfficeSpecificElement.xsd)

```
<xs:include schemaLocation ="XXOfficeSpecificElement.xsd"/>
```

7. Create a new Type (e.g., XXPersonNameType) which is an extension of the corresponding ST.96 complex Type (e.g., com:PersonNameType) in which the new element (e.g., xx:XXOfficeSpecificElement) is added.

```
<xsd:complexType name="XXPersonNameType">
<xsd:complexContent>
<xsd:extension base="com:PersonNameType">
 <xsd:sequence>
         <xsd:element ref="xx:XXOfficeSpecificElement"/>
 </xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
```

8. Declare the new element and the corresponding new type

```
<xsd:element name="XXPersonName" type="xx:XXPersonNameType"/>
```

### 1.4 Remove and add an element

In order to add a new element and remove another one, developers SHOULD follow the following steps:

1. Create a new W3C schema, e.g., XXOfficeSpecificElement.xsd

2. Set the targetNamespace to IPO's namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Create a new element (e.g., XXOfficeSpecificElement) and a type for your new element (e.g., XXOfficeSpecificType)

```
<xsd:element name="XXOfficeSpecificElement"
type="xx:XXOfficeSpecificType"/>
<xsd:simpleType type="XXOfficeSpecificType">
<xsd:restriction base="xsd:string">
<xsd:maxLength value="2"/>
</xsd:restriction>
</xsd:simpleType>
```

4. Create a new W3C schema file, e.g., XXPersonName.xsd

5. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

6. Import ST.96 Components which are needed to define a new Type, e.g., XXPersonNameType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonNamePrefix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Mi
ddleName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/La
stName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonNameSuffix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pr
eferredName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonOtherName.xsd"/>
```

7. Include the schema that defines the new element (e.g., XXOfficeSpecificElement.xsd)

```
<xsd:include schemaLocation ="XXOfficeSpecificElement.xsd"/>
```

8. Add a new Type (XXPersonNameType) which refers to some ST.96 elements as defined in the corresponding ST.96 Type (e.g., PersonNamePrefix, FirstName, MiddleName, LastName, PersonNameSuffix, PreferredName and PersonOtherName of PersonNameType) and add to the new element (e.g., xx:XXOfficeSpecificElement).

```
<xsd:complexType name="XXPersonNameType">
         <xsd:sequence>
             <xsd:element ref="com:PersonNamePrefix" minOccurs="0"/>
             <xsd:element ref="com:FirstName" />
             <xsd:element ref="com:MiddleName" minOccurs="0"/>
             <xsd:element ref="com:LastName" />
             <xsd:element ref="com:PersonNameSuffix" minOccurs="0"/>
             <xsd:element ref="com:PreferredName" minOccurs="0"/>
             <xsd:element ref="com:PersonOtherName" minOccurs="0"/>
              <xsd:element ref="xx:XXOfficeSpecificElement"
minOccurs="0"/>
       </xsd:sequence>
</xsd:complexType>
```

9. Declare the new element and the corresponding new type

```
<xsd:element name="XXPersonName" type="xx:XXPersonNameType"/>
```

### 1.5 Add an attribute

In order to add a new attribute, developers SHOULD follow the following steps:

1. Create a new W3C schema xxofficeSpecificAttribute.xsd
2. Set the targetNamespace to IPO's namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Define a new attribute (e.g., xxofficeSpecificAttribute) and create a type for it (e.g., XXOfficeSpecificType)

```
<xsd:attribute name="xxofficeSpecificAttribute"
type="xx:XXOfficeSpecificType"/>
<simpleType type="XXOfficeSpecificType">
<xsd:restriction base="xsd:string">
<xsd:maxLength value="2"/>
</xsd:restriction>
</xsd:simpleType>
```

4. Create a new W3C schema file, e.g., XXPersonName.xsd
5. Set the targetNamespace to IPO's namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

6. Import ST.96 Components which are needed to define a new Type, e.g., XXPersonNameType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Aggregat
e/PersonName.xsd"/>
```

7. Include the schema that defines the new attribute (e.g., xxofficeSpecificAttribute.xsd)

```
<xsd:include schemaLocation ="xxOfficeSpecificAttribute.xsd"/>
```

8. Add a new Type (e.g., XXPersonNameType) which is an extension of the corresponding ST.96 Complex Type (e.g., com:PersonNameType) in which the new Attribute (e.g., xx:xxOfficeSpecificAttribute) is added.

```
<xsd:complexType name="XXPersonNameType">
<xsd:complexContent>
<xsd:extension base="com:PersonNameType">
         <xsd:attribute ref="xx:xxOfficeSpecificAttribute"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
```

9. Declare the new element and the corresponding new type

```
<xsd:element name="XXPersonName" type="xx:xx:XXPersonNameType"/>
```

### 1.6 Remove and add an attribute

In order to add a new attribute and remove another one, developers SHOULD follow the following steps:

1. Create a new W3C schema xxofficeSpecificAttribute.xsd

2. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Define a new attribute (e.g., xxofficeSpecificAttribute) and create a type for it (e.g., XXOfficeSpecificType)

```
<xsd:attribute name="xxofficeSpecificAttribute"
type="xx:OfficeSpecificType">
<xsd:simpleType name="XXOfficeSpecificType">
<xsd:restriction base="xsd:string">
<xsd:maxLength value="2"/>
</xsd:restriction>
</xsd:simpleType>
```

4. Create a new W3C schema file, e.g., XXPersonName.xsd

5. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

6. Include the schema that defines the new attribute (e.g., xxofficeSpecificAttribute.xsd)

```
<xsd:include schemaLocation ="xxofficeSpecificAttribute.xsd"/>
```

7. Import ST.96 Components which are needed to define a new Type, e.g., XXPersonNameType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonNamePrefix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Mi
ddleName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/La
stName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Se
condLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonNameSuffix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pr
eferredName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonOtherName.xsd"/>
```

8. Add a new Type (e.g., XXPersonNameType) which refers to all elements and attributes defined in the corresponding ST.96 Type except one attribute (e.g., languageCode) which is replaced by the new defined attribute (e.g., xx:xxofficeSpecificAttribute)

```
<xsd:complexType name="XXPersonNameType">
 <xsd:sequence>
           <xsd:element ref="com: PersonNamePrefix" minOccurs="0"/>
           <xsd:element ref="com:FirstName" />
           <xsd:element ref="com:MiddleName" minOccurs="0"/>
           <xsd:element ref="com:LastName" />
           <xsd:element ref="com:FirstLastName" minOccurs="0"/>
           <xsd:element ref="com:SecondLastName" minOccurs="0"/>
           <xsd:element ref="com:PersonNameSuffix"  minOccurs="0"/>
           <xsd:element ref="com:PreferredName" minOccurs="0"/>
           <xsd:element ref="com:PersonOtherName" minOccurs="0"/>
      </xsd:sequence>
<xsd:attribute ref="xx:xxofficeSpecificAttribute"/>
</xsd:complexType>
```

9. Declare the new element and the corresponding new type

```
<xsd:element name="XXPersonName" type="xx:XXPersonNameType"/>
```

## 1.7 Rename an element

In order to rename new element, developers SHOULD follow the following steps:

1. Create a new W3C schema file, e.g., XXSurName.xsd
2. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organiazion.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Define an element with a new name (e.g., XXSurname) and refer to its corresponding type

```
<xsd:element name="XXSurName" type="xsd:string">
```

4. Create a new W3C schema file, e.g., XXPersonName.xsd
5. Set the targetNamespace to your namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

6. Import ST.96 Components which are needed to define a new Type, e.g., XXPersonNameType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonNamePrefix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Mi
ddleName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Se
condLastName.xsd"/>
```

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonNameSuffix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pr
eferredName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonOtherName.xsd"/>
```

7. Include the schema that defines the renamed element (e.g., XXSurName.xsd)

```
<xsd:include schemaLocation ="XXSurName.xsd"/>
```

8. Add a new Type (e.g., XXPersonNameType) which refers to the ST.96 elements defined in the corresponding ST.96 Type (e.g., PersonNameType) except one element (e.g., LastName of PersonNameType) which is replaced by the new defined element (e.g., XXSurName)

```
<xsd:complexType name="XXPersonNameType">
         <xsd:sequence>
             <xsd:element ref="com:PersonNamePrefix" minOccurs="0"/>
             <xsd:element ref="com:FirstName" />
             <xsd:element ref="com:MiddleName" minOccurs="0"/>
          <xsd:element ref="xx:XXSurName"/>
             <xsd:element ref="com:FirstLastName" minOccurs="0"/>
             <xsd:element ref="com:SecondLastName" minOccurs="0"/>
             <xsd:element ref="com:PersonNameSuffix" minOccurs="0"/>
             <xsd:element ref="com:PreferredName" minOccurs="0"/>
             <xsd:element ref="com:PersonOtherName" minOccurs="0"/>
       </xsd:sequence>
</xsd:complexType>
```

9. Declare the new element and the corresponding new type

```
<xsd:element name="XXPersonName" type="xx:XXPersonNameType"/>
```

1.8 Rename an attribute

In order to rename an attribute, developers SHOULD follow the following steps:

1. Create a new W3C schema file for new attribute name, e:g., xxwebAddressKind.xsd
2. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Import ST.96 Components which are needed to declare a new attribute, e.g., xxwebAddressKind

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Aggregat
e/WebAddressCategoryType.xsd"/>
```

4. Declare the new attribute (xxwebAddressKind) and reuse an ST.96 type (com:WebAddressCategoryType)to define its type.

```
<xsd:attribute name="xxwebAddressKind"
type="com:WebAddressCategoryType"/>
```

5. Create a new W3C schema file, e.g., XXWebAddress.xsd

6. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

7. Include the schema that defines the new attribute (e.g., xxwebAddressKind.xsd) into the schema (e.g., XXWebAddress.xsd)

```
<xsd:include schemaLocation="xxwebAddressKind.xsd"/>
```

8. Add a new Type (e.g., XXWebAddressType) which refers to the elements and attributes defined in the ST.96 corresponding Type except one  attribute (e.g., webAddressCategory of WebAddressType) which is replaced by the new defined one (e.g., xx:xxWebAddressKind)

```
<xsd:complexType name="XXWebAddressType">
<xsd:simpleContent>
<xsd:extension base="xsd:string">
        <xsd:attribute ref="xx:xxWebAddressKind"/>
</xsd:extension>
</xsd:simpleContent>
</xsd:complexType>
```

9. Declare the new element and the corresponding new type

```
<xsd:element name="XXWebAddress" type="xx:xxWebAddressType"/>
```

## 2. SIMPLE TYPES SCHEMAS

### 2.1 Add a character length restriction

In order to add a character length restriction on a simple Type, developers SHOULD take the following steps:

1. Create a new .W3C schema, e:g., XXPersonNamePrefix.xsd

2. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Create an element and its corresponding simple type, based on the existing ST.96 simple Type or on a W3C Built-in Datatype, for the element you wish to place a character length restrict on.

```
<xsd:element name="XXPersonNamePrefix" type="xx:XXPersonNamePrefixType"/>
<xsd:simpleType name="XXPersonNamePrefixType">
<xsd:restriction base="xsd:string">
<xsd:maxLength value="3"/>
</xsd:restriction>
</xsd:simpleType>
```

4. Create a new W3C schema file, e.g., XXPersonName.xsd

5. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

6. Include the schema that defines the new content (e.g., XXPersonNamePrefix.xsd)

```
<xsd:include schemaLocation ="XXPersonNamePrefix.xsd"/>
```

7. Import ST.96 Components which are needed to define a new Type, e.g., XXPersonNameType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Mi
ddleName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/La
stName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Se
condLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonNameSuffix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pr
eferredName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonOtherName.xsd"/>
```

8. Add a new Type (e.g., XXPersonNameType) which refers to some ST.96 elements defined in the corresponding ST.96 Type and in which an element (e.g., com:PersonNamePrefix) is replaced by the new one (xx:XXPersonNamePrefix).

```
<xsd:complexType name="XXPersonNameType">
         <xsd:sequence>
              <xsd:element ref="xx:XXPersonNamePrefix" minOccurs="0"/>
             <xsd:element ref="com:FirstName"/>
             <xsd:element ref="com:MiddleName" minOccurs="0"/>
             <xsd:element ref="com:LastName" />
             <xsd:element ref="com:FirstLastName" minOccurs="0"/>
             <xsd:element ref="com:SecondLastName" minOccurs="0"/>
             <xsd:element ref="com:PersonNameSuffix" minOccurs="0"/>
             <xsd:element ref="com:PreferredName" minOccurs="0"/>
             <xsd:element ref="com:PersonOtherName" minOccurs="0"/>
       </xsd:sequence>
</xsd:complexType>
```

9. Declare the new element and the corresponding new type

```
<xsd:element name="XXPersonName" type="xx:XXPersonNameType"/>
```

### 2.2 Add an enumeration list

In order to add an enumeration list to a simple Type, developers SHOULD follow the following steps:

1. Create a new .W3C schema, e:g., XXPersonNamePrefix.xsd
2. Set the targetNamespace to IPO's namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Create an element and its corresponding simple Type, based on the existing ST.96 simple Type or a W3C Built-in Datatype, for the element you wish to create an enumeration for.

```
<xsd:element name="XXPersonNamePrefix" type=”xx:XXPersonNamePrefixType”/>
<xsd:simpleType name="XXPersonNamePrefixType">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="Mrs"/>
<xsd:enumeration value="Mr"/>
<xsd:enumeration value="Ms"/>
<xsd:enumeration value="Dr"/>
</xsd:restriction>
</xsd:simpleType>
```

4. Create a new W3C schema file, e.g., XXPersonName.xsd

5. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

6. Include the schema that defines the new content (e.g., XXPersonNamePrefix.xsd)

```
<xsd:include schemaLocation ="XXPersonNamePrefix.xsd"/>
```

7. Import ST.96 Components which are needed to define a new Type, e.g., XXPersonNameType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Mi
ddleName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/La
stName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Fi
rstLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Se
condLastName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonNameSuffix.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pr
eferredName.xsd"/>
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/Pe
rsonOtherName.xsd"/>
```

8. Add a new Type (e.g., XXPersonNameType) which refers to some ST.96 elements defined in the corresponding ST.96 Type and in which an element (com:PersonNamePrefix).is replaced by a new one (xx:XXPersonNamePrefix )

```
<xsd:complexType name="XXPersonNameType">
         <xsd:sequence>
              <xsd:element ref="xx:XXPersonNamePrefix" minOccurs="0"/>
            <xsd:element ref="com:FirstName"/>
            <xsd:element ref="com:MiddleName" minOccurs="0"/>
            <xsd:element ref="com:LastName" />
            <xsd:element ref="com:FirstLastName" minOccurs="0"/>
            <xsd:element ref="com:SecondLastName" minOccurs="0"/>
            <xsd:element ref="com:PersonNameSuffix" minOccurs="0"/>
            <xsd:element ref="com:PreferredName" minOccurs="0"/>
            <xsd:element ref="com:PersonOtherName" minOccurs="0"/>
       </xsd:sequence>
```

```
</xsd:complexType>
```

9. Declare the new element and the corresponding new type

```
<xsd:element name="XXPersonName" type="xx:XXPersonNameType"/>
```

### 2.3 Add a pattern restriction

In order to add a pattern restriction on a simple Type, developers SHOULD follow the following steps:

1. Create a new W3C schema file, e.g., XXPhoneNumberContentType.xsd
2. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

3. Create a simple Type, based on the existing ST.96 simple Type or a W3C Built-in Datatype, for the element you wish to place a pattern restriction on.

```
<xsd:simpleType name="XXPhoneNumberContentType">
<xsd:restriction base="xsd:string">
<xsd:pattern value="\d{3}-\d{3}-\d{4}"/>
</xsd:restriction>
</xsd:simpleType>
```

4. Create a new W3C schema file, e.g., XXPhoneNumber.xsd
5. Set the targetNamespace to IPO’s namespace

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.Organization.com"
xmlns:xx="http://www.Organization.com"
xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
elementFormDefault="qualified" attributeFormDefault="qualified">
```

6. Include the schema that defines the new content (e.g., XXPhoneNumberContentType.xsd)

```
<xsd:include schemaLocation="XXPhoneNumberContentType.xsd"/>
```

7. Import ST.96 Components which are needed to define a new Type, e.g., XXPhoneNumberType

```
<xsd:import namespace="http://www.wipo.int/standards/XMLSchema/Common/1"
schemaLocation="http://www.wipo.int/standards/XMLSchema/Common/1/Basic/ph
oneNumberCategory.xsd"/>
```

8. Add a new Type (XXPhoneNumberType) which refers to the new content (e.g., xx:XXPhoneNumberContentType) as a basis for the new type (xx:XXPhoneNumberType) instead of the Type used as a basis for the corresponding ST.96 Type (e.g., xsd:string for PhoneNumberType)

```
<xsd:complexType name="XXPhoneNumberType">
<xsd:complexContent>
<xsd:extension base="xx:XXPhoneNumberContentType">
<xsd:attribute ref="com:phoneNumberCategory"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
```

9. Declare the new element and the corresponding new type

```
<xsd:element name="XXPhoneNumber" type="xx:XXPhoneNumberType"/>
```

### 2.4 Change an existing enumeration list

In order to change an existing enumeration list, developers SHOULD follow the following steps:

1. Create a new W3C schema file, e.g., xxwebAddressCategory.xsd
2. Set the targetNamespace to IPO’s namespace

   ```
   <?xml version="1.0"?>
   <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://www.Organization.com"
   xmlns:xx="http://www.Organization.com"
   xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
   elementFormDefault="qualified" attributeFormDefault="qualified">
   ```

3. Define an attribute and create a new simple Type including old and additional values.

   ```
   <xsd:attribute name="xxwebAddressCategory"
   type="xx:XXWebAddressCategoryType"/>
   <xsd:simpleType name="XXWebAddressCategoryType">
   <xsd:restriction base="xsd:string">
   <xsd:enumeration value="Private"/>
   <xsd:enumeration value="Business"/>
   <xsd:enumeration value="Personal"/>
   <xsd:enumeration value="Professional"/>
   </xsd:restriction>
   </xsd:simpleType>
   ```

4. Create a new W3C schema file, e.g., XXWebAddress.xsd
5. Set the targetNamespace to IPO’s namespace

   ```
   <?xml version="1.0"?>
   <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://www.Organization.com"
   xmlns:xx="http://www.Organization.com"
   xmlns:com="http://www.wipo.int/standards/XMLSchema/Common/1"
   elementFormDefault="qualified" attributeFormDefault="qualified">
   ```

6. Include the schema of the new attribute (e.g., xxwebAddressCategory)

   ```
   <xsd:include schemaLocation="xxwebAddressCategory.xsd"/>
   ```

7. Add a new Type (xx:XXWebAddressType) which refers to the same basis used in the corresponding ST.96 Type (e.g., XXWebAddressType) and in which the new attribute (e.g., ipo:xxWebAddressCategory) is used instead of the ST.96 attribute (e.g., webAddressCategory).

   ```
   <xsd:complexType name="XXWebAddressType">
   <xsd:simpleContent>
   <xsd:extension base="xsd:string"/>
   <xsd:attribute ref="xx:xxWebAddressCategory"/>
   </xsd:simpleContent>
   </xsd:complexType>
   ```

8. Declare the new element and the corresponding new type

   ```
   <xsd:element name="XXWebAddress" type="xx:xxWebAddressType"/>
   ```

[End of Annex VI]