Metadata Registry XML Schema

David Milward

1 Introduction

This document forms a definition of the XML Schema which is being used to define the core meta-model used by the Metadata Registry. The core meta-model used is derived from the ISO Standard for Metadata Registries - ISO11179, however there are a number of fundamental differences. The most basic is the fact that ISO11179 doesn't have a notion of a collection of data elements, and here we are using the notion of a Collection and a Model. A *Collection* can be viewed as a dataset used for a particular purpose, it is closely related to the idea of an Ontology, although it is not, it simply a collection of Models and Data Elements.

The core model of this metadata registry around the core ideas defined in ISO11179, it is therefore appropriate to run through some of the core ideas included in that standard. Firstly what is metadata, ISO11179-1 defines a datum, designation and value as follows:

> **Datum:** *Designation whose concept is a value.*
> **Designation:** *Designation whose concept is a value.*
> **Value:** *Designation whose concept is a value.*

Metadata is defined as follows:

> **Metadata:** *Descriptive data about an object*

The idea of an object is not defined, but it can be assumed to be a data object, a class or a data entity model. Metadata about an object is assumed to include its attributes which are separated into groupings as follows :

- characteristics
- property values
- datatypes
- codings
- identifiers

2 Model Catalogue Core Schema

The next sections examine the XML Schema section by section.

2.1 Section 1

The namespace for the Metadata Registry schema is identified using the abbreviation *mc* which stands for Model Catalogue, of which the MDR forms a key component. The namespace is defined by the schema file located on the *www.metadataregistry.org.uk* website. The rest of this part of the schema file defines 3 XML Simple types.

Listing 1: section 1

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns:mc="http://www.metadataregistry.org.uk/assets/schema/1.0/mdr.xsd"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.metadataregistry.org.uk/assets/schema/1.0/mdr.xs
    elementFormDefault="qualified" version="1.0" xml:lang="EN">
    <xs:simpleType name="Name">
        <xs:restriction base="xs:string">
            <xs:minLength value="1"/>
            <xs:maxLength value="255"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="ID">
        <xs:restriction base="xs:anyURI">
            <xs:pattern value="https?://.*"/>
            <xs:minLength value="1"/>
            <xs:maxLength value="255"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="Status">
        <xs:restriction base="xs:string">
            <xs:enumeration value="DRAFT"/>
            <xs:enumeration value="FINALIZED"/>
            <xs:enumeration value="DEPRECATED"/>
        </xs:restriction>
    </xs:simpleType>
```

2.1.1 Name

This element is specified to be an xml string as defined by *xs:string* base, which has a restriction on it to limit it to a set of characters of length between 1 and 255.

2.1.2 ID

This element can be any uri as given by the base *xs:anyURI* and it is constrained to be a set of charaters having a pattern conforming to the regular expression of *https?://.** , and length between 1 and 255 characters.

2.1.3 Status

This element again has a base of *xs:string*, but is constrained to have one of three values : DRAFT, FINALIZED, or DEPRECATED.

2.2 Section 2

2.2.1 PreservedString

This element is simply defining itself as an string which preserves white space, this means that any XML processors will not remove whitespace from any elements which are using this as their base.

2.2.2 Description

The description is based on the previous *white-space preserving* string, and is restricted to a maximum length of 2000 characters. c This element is a *Rule* or constraint which can be applied to a string, it could be a regular expression or set of rules for how an element is formed. The element itself is based on the PreservedString element

Listing 2: section 2

```
<xs:simpleType name="PreservedString">
    <xs:restriction base="xs:string">
        <xs:whiteSpace value="preserve"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="Description">
    <xs:restriction base="mc:PreservedString">
        <xs:maxLength value="2000"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="Rule">
    <xs:restriction base="mc:PreservedString">
        <xs:maxLength value="10000"/>
    </xs:restriction>
</xs:simpleType>
```

2.3 Section 3

These elements relate to the extensions which can be attached to existing element. They are simply designed to hold undefined metadata. In nearly every dataset there is metadata which is unique to that dataset, perhaps due to the nature of the data iteself or due to the way it is stored, so the metadata registry needs a means of capturing this *unplanned* data.

2.3.1 ExtensionKey

This is simply a string between 1 and 255 characters length which can hold a key.

2.3.2 ExtensionValue

This is simply a string which can hold a string value up to 2000 characters in length.

2.3.3 Symbol

This is a string restricted to 100 characters long. ?????NOT SURE HERE

2.3.4 Extension

This is a complex type which contains the previously defined key-value pair

2.3.5 Extensions

This is a complex type which defines an unbounded sequence of Extensions

Listing 3: section 3

```
<xs:simpleType name="ExtensionKey">
    <xs:restriction base="xs:string">
        <xs:minLength value="1"/>
        <xs:maxLength value="255"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="ExtensionValue">
    <xs:restriction base="xs:string">
        <xs:maxLength value="2000"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="Symbol">
    <xs:restriction base="xs:string">
        <xs:minLength value="1"/>
        <xs:maxLength value="100"/>
    </xs:restriction>
</xs:simpleType>
<xs:complexType name="Extension">
    <xs:simpleContent>
        <xs:extension base="mc:ExtensionValue">
            <xs:attribute name="key" type="mc:ExtensionKey" use="required"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>
```

```
<xs:complexType name="Extensions">
    <xs:sequence>
        <xs:element name="extension" type="mc:Extension" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
```

2.4 Section 4

This section describes the Enumerations which are normally represented in datatypes, as well as defining the basic metadata elements : Classification, model, dataElement, valueDomain, dataType and measurementUnit. A known element type will be restricted to having a name that is one of these previously defined values.

2.4.1 Enumeration

This is a single values, the values are based on xs:string elements, and these values can be contained as a set in the following definintion of Enumerations.

2.4.2 Enumerations

This is an unbounded set of Enumeration elements, as defined in the previous paragraph.

2.4.3 KnownCatalogueElementType

This is an xml construction to define a *KnownCatalogueElementType* as having a name which is contrained to being one of the shown character values.

2.4.4 JavaClass

????What is the RE doing???

Listing 4: section4

```
<xs:complexType name="Enumeration">
    <xs:simpleContent>
        <xs:extension base="xs:string">
            <xs:attribute name="value" use="required" type="xs:string"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>
<xs:complexType name="Enumerations">
    <xs:sequence>
        <xs:element name="enumeration" type="mc:Enumeration" maxOccurs="unbounde
    </xs:sequence>
</xs:complexType>
<xs:simpleType name="KnownCatalogueElementType">
```

```
    <xs:restriction base="mc:Name">
        <xs:enumeration value="classification"/>
        <xs:enumeration value="model"/>
        <xs:enumeration value="dataElement"/>
        <xs:enumeration value="valueDomain"/>
        <xs:enumeration value="dataType"/>
        <xs:enumeration value="measurementUnit"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="JavaClass">
    <xs:restriction base="xs:string">
        <xs:pattern value="([\p{L}_$][\p{L}\p{N}_$]*\.)*[\p{L}_$][\p{L}\p{N}_$]*"
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="CatalogueElementType">
    <xs:union memberTypes="mc:JavaClass mc:KnownCatalogueElementType"/>
</xs:simpleType>
```

2.5 Section 5

2.5.1 Reference

A Reference is a means of linking to another metadata element, perhaps in a different classifications (or registry), and ii uses several different mechanisms. Ideally the *ref* will simply be a unique mc:id, but it can also be a combination of name, classification and datatype.

*****

Listing 5: section5

```
<xs:complexType name="Reference">
    <xs:annotation>
        <xs:documentation>
            The Reference can must always contain at least "ref" attribute or "n
            provided the name will be searched only within this classification.
            is used inside element of type "Classification" it is assumed that t
            within the surrounding classification. This does not apply on elemen
            provided. In that case the element is resolved by the ID. The type o
            be provided to distinguish between elements having same name and cla
            (this is quite common for value domains and it's data types)
        </xs:documentation>
    </xs:annotation>
    <xs:attribute name="name" type="mc:Name" />
    <xs:attribute name="classification" type="mc:Name" />
    <xs:attribute name="ref" type="mc:ID" />
    <xs:attribute name="type" type="mc:CatalogueElementType" />
</xs:complexType>
```

2.6 Section 6

2.6.1 KnownRelationshipTypes

This is an enumeration of possible relationship tyupes, at present these consist of the following:

- containment
- base
- hierarchy
- relatedTo
- synonym
- favourite
- classification

Containment indicates that a metadata element is contained by another element. Base indicates that an element is based on another element. Hierarchy indicates that there is parent-child relationship between two elements **** relatedTo is a stop-gap relationship. Synonym means that one element *means* the same as another one.

2.6.2 RelationshipType

A RelationshipType

2.6.3 Relationship

A Relationship is an element which is based on the previously defined mc:Reference, and which must define what kind of Relationship it is, out of the RelationshipTypes previously defined.

2.6.4 Relationships

A Relationships element is designed to constrain a Relationship to be either a *to* or a *from* relationship
****

Listing 6: section6

```
<xs:simpleType name="KnownRelationshipTypes">
    <xs:restriction base="mc:Name">
        <xs:enumeration value="containment"/>
        <xs:enumeration value="base"/>
        <xs:enumeration value="hierarchy"/>
        <xs:enumeration value="relatedTo"/>
```

```
        <xs:enumeration value="synonym"/>
        <xs:enumeration value="favourite"/>
        <xs:enumeration value="classification"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="RelationshipType">
    <xs:union memberTypes="mc:Name mc:KnownRelationshipTypes"/>
</xs:simpleType>
<xs:complexType name="Relationship">
    <xs:complexContent>
        <xs:extension base="mc:Reference">
            <xs:attribute name="relationshipType" type="mc:RelationshipType" use
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<xs:complexType name="Relationships">
    <xs:choice maxOccurs="unbounded">
        <xs:element name="from" type="mc:Relationship"/>
        <xs:element name="to" type="mc:Relationship"/>
    </xs:choice>
</xs:complexType>
```

2.7 Section 7

2.7.1 CatalogueElement

This is the core metadata element in the registry, and this xml element defines the basic attributes that it is expected to have, namely:

- basedOn
- description
- extensions
- relationships

Listing 7: section7

```
<xs:complexType name="CatalogueElement" abstract="true">
    <xs:annotation>
        <xs:documentation>
            CatalogueElement is either a in-lined definition of the element or a
            If "ref" attribute is set, any other attributes or nested elements a
        </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="mc:Reference">
```

```
            <xs:sequence>
                <xs:element name="basedOn" type="mc:Reference" minOccurs="0" max
                <xs:element name="description" type="mc:Description" minOccurs="
                <xs:element name="extensions" type="mc:Extensions" minOccurs="0"
                <xs:element name="relationships" type="mc:Relationships" minOccu
            </xs:sequence>
            <xs:attribute name="id" type="mc:ID"/>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

2.8 Section 8

2.8.1 MeasurementUnit

A measurement unit is just that, where a dataType is related to commonly used measurement unit, this can be recorded by a separate element which is contained in the value domain. The xml element descriped here is a catalogue element, with an attribute of type mc:Symbol. **** Usage?

Listing 8: section8

```
<xs:complexType name="MeasurementUnit">
    <xs:complexContent>
        <xs:extension base="mc:CatalogueElement">
            <xs:attribute name="symbol" type="mc:Symbol"/>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

2.9 Section 9

2.9.1 DataType

This XML element is based on a catalogue element and can contain zero or more enumerations.

Listing 9: section9

```
<xs:complexType name="DataType">
    <xs:complexContent>
        <xs:extension base="mc:CatalogueElement">
            <xs:sequence>
                <xs:element name="enumerations" type="mc:Enumerations" minOccurs
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

2.10 Section 10

2.10.1 ValueDomain

The xml representation of a value domain is based again on the catalogue element, and has a number of attributes. The first is the choice between a rule, of type mc:Rule or a regular expression, of type xs:string. A value domain element can also contain zero or more dataTypes, and zero or more unitsOfMeasure

Listing 10: section10

```
<xs:complexType name="ValueDomain">
    <xs:complexContent>
        <xs:extension base="mc:CatalogueElement">
            <xs:sequence>
                <xs:choice minOccurs="0">
                    <xs:element name="rule" type="mc:Rule" />
                    <xs:element name="regex" type="xs:string" />
                </xs:choice>
                <xs:element name="unitOfMeasure" type="mc:MeasurementUnit" minOc
                <xs:element name="dataType" type="mc:DataType" minOccurs="0" />
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

2.11 Section 11

2.11.1 DataElement

The XML element representing a data element can have zero or more value domain elements.

Listing 11: section11

```
<xs:complexType name="DataElement">
    <xs:complexContent>
        <xs:extension base="mc:CatalogueElement">
            <xs:sequence>
                <xs:element name="valueDomain" type="mc:ValueDomain" minOccurs="
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

2.12 Section 12

2.12.1 Model

The XML element for a model can contain other models and other data elements. *** Can we have a model containing no other models or data elements ? Maybe this should be a sequence?

Listing 12: section12

```
<xs:complexType name="Model">
    <xs:complexContent>
        <xs:extension base="mc:CatalogueElement">
            <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:element name="model" type="mc:Model" />
                <xs:element name="dataElement" type="mc:DataElement" />
            </xs:choice>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

2.13 Section 13

2.13.1 Classification

A Classification needs to contain an unbounded number of catalogue elements, together with a globalSearchFor element, which in turn is union of Java classes and known catalogue elements. **** Need to understand this a bit more.

Listing 13: section13

```
<xs:complexType name="Classification">
    <xs:complexContent>
        <xs:extension base="mc:CatalogueElement">
            <xs:sequence>
                <xs:element name="globalSearchFor" type="mc:CatalogueElementType
                <xs:choice minOccurs="0" maxOccurs="unbounded">
                    <xs:element name="model" type="mc:Model" />
                    <xs:element name="dataElement" type="mc:DataElement" />
                    <xs:element name="valueDomain" type="mc:ValueDomain" />
                    <xs:element name="dataType" type="mc:DataType" />
                    <xs:element name="measurementUnit" type="mc:MeasurementUnit"
                </xs:choice>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

2.14 Section 14

2.14.1 Catalogue

This XML Element encompasses the whole catalogue, it is basically an unbounded sequence of XML Elements which represents the various elements in the metadata registry. Is there a createAutomatically for each element in the metadata registry - or is it one createAutomatically for a set of elements ?

Listing 14: section14

```
<xs:complexType name="Catalogue">
    <xs:sequence>
        <xs:element name="createAutomatically" type="mc:CatalogueElementType" mi
        <xs:choice maxOccurs="unbounded">
            <xs:element name="classification" type="mc:Classification" />
            <xs:element name="model" type="mc:Model" />
            <xs:element name="dataElement" type="mc:DataElement" />
            <xs:element name="valueDomain" type="mc:ValueDomain" />
            <xs:element name="dataType" type="mc:DataType" />
            <xs:element name="measurementUnit" type="mc:MeasurementUnit" />
        </xs:choice>
    </xs:sequence>
</xs:complexType>
<xs:element name="catalogue" type="mc:Catalogue"/>
</xs:schema>
```