INTERNATIONAL STANDARD

ISO/IEC 24800-2

Second edition
2021-04

Information technology — JPSearch —

Part 2: Registration, identification and management of schema and ontology

Technologies de l'information — JPSearch —

Partie 2: Enregistrement, identification et gestion des schémas et de l'ontologie

Reference number
ISO/IEC 24800-2:2021(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

This second edition cancels and replaces the first edition (ISO/IEC 24800-2:2011), which has been technically revised. It also incorporates the amendment ISO/IEC 24800-2:2011/Amd.1:2015.

The main changes compared to the previous edition are as follows:

— editorial changes throughout the text to fully align this document with ISO/IEC Directives;

— changes to the registration procedure for JPOnto in 8.3 and Annex A.

A list of all parts in the ISO/IEC 24800 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

This document provides a standardized set of technologies for metadata representation, querying and management of images. It specifies the JPSearch's Core Metadata Schema as the cornerstone of metadata interoperability in the ISO/IEC 24800 series. It also specifies the structure and rules to which any metadata annotation of images must conform in order to be considered valid within a JPSearch compliant system.

In addition to the definition of JPSearch Core Metadata Schema, the ISO/IEC 24800 series provides a mechanism which allows a JPSearch compliant system taking profit from proprietary or community-specific metadata schemas. A translation rules language is defined, allowing the publication of machine-readable translations between metadata terms belonging to proprietary metadata schemas and metadata terms in the JPSearch Core Metadata Schema. Users can choose which metadata language to use in a JPSearch-based interaction (annotation, querying, etc.) if the proper translations are available.

In order to specify the issues in a detailed manner in this document, this document first provides the fundamental information including scope definition, description of terms and definitions, and conventions that are necessary to understand this document. The definition of JPSearch Core Metadata Schema is described in the context of XML structure. Management of information regarding other metadata schema is also described in respect of registration, maintenance, and translation rules.

Information technology — JPSearch —

Part 2: Registration, identification and management of schema and ontology

1 Scope

This document specifies a series of interfaces to allow disparate systems an interoperable management of image repositories. It also specifies the general rules which govern the usage of metadata in JPSearch and provides a specification which

— provides rules for the representation of image metadata descriptions, consisting of the definition of the JPSearch Core Metadata Schema,

— provides rules for the publication of machine-readable translations between metadata terms belonging to proprietary metadata schemas and metadata terms in the JPSearch Core Metadata Schema, and

— provides rules for the registration and request of metadata schemas and its translation rules or links to them.

JPSearch is an extensible standard. The method of extending the structures and rules beyond the JPSearch Core Metadata Schema is provided in this document.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

W3C Recommendation. *Extensible Markup Language (XML) 1.0 (Fifth Edition).* 26 November 2008, available at http://wwww3.org/TR/xml/

W3C Recommendation. *XML Schema Part 1: Structures Second Edition*. 28 October 2004, available at http://www.w3.org/TR/xmlschema-1/

W3C Recommendation. *XML Schema Part 2: Datatypes Second Edition*. 28 October 2004, available at http://www.w3.org/TR/xmlschema-2/

W3C Recommendation. *XML Path Language (XPath)*. 16 November 1999, available at http://www.w3.org/TR/xpath

W3C Recommendation. *Resource Description Framework (RDF): Concepts and Abstract Syntax*. 10 February 2004, available at http://www.w3.org/TR/2004/REC-rdf-concepts-20040210

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
core schema
metadata basis supporting interoperability during search among multiple image retrieval systems

Note 1 to entry: The core schema is used by clients to formulate in combination with the JPEG Query Format search requests to JPSearch compliant search systems. Note, that only metadata described by the core schema is guaranteed to be processed by JPSearch compliant systems.

3.2
translation rules
machine-readable declaration of the semantic and syntactic mappings between a proprietary metadata schema and the JPSearch's core metadata schema

4 Conventions

4.1 Naming convention

In order to specify the JPSearch Core metadata description scheme, this document uses constructs provided by XML such as "element" and "complexType." The names associated to these constructs are created on the basis of the following conventions:

If the name is composed of multiple words, the first letter of each word is capitalized, with the exception that the capitalization of the first word depends on the type of construct and is described below.

— Element naming: the first letter of the first word is capitalized (e.g. `Identifier` element of `JPSearchCoreType`).

— Attribute naming: the first letter of the first word is not capitalized (e.g. jpsearchID attribute of `ManagementType` type).

— complexType naming: the first letter of the first word is capitalized, and the suffix "Type" is used at the end of the name (e.g. `JPSearchCoreType`).

— simpleType naming: the first letter of the first word is not capitalized, the suffix "Type" may be used at the end of the name (e.g. `xPathType`).

4.2 Document convention

The syntax of each description is specified using the constructs provided by XML as defined in XML, XML Schema Part 1, and XML Schema Part 2, and is presented in this document using a specific font and background as shown in the following example:

```
<complexType name="ExampleType">
   <sequence>
      <element name="Element1" type="string"/>
   </sequence>
   <attribute name="attribute1" type="string" default="attrvalue1"/>
</complexType>
```

The semantics of each description tool is specified in text using a table format, where each row contains the name and a definition of a type, element or attribute as shown in the following example:

Name	Definition
ExampleType	Specifies an …
element1	Describes the …
attribute1	Describes the …

4.3 Wrapper of the schema

The description examples and syntax of description tools specified in this document assume that a schema wrapper is provided which identifies the XML Schema namespace (XML Schema) and JPSearch namespace:

```
<schema xmlns="http://www.w3.org/2001/XMLSchema"
        xmlns:JPCore="JPSearch:schema:coremetadata"
        targetNamespace="JPSearch:schema:coremetadata"
        elementFormDefault="qualified"
        attributeFormDefault="unqualified">
```

The following tag is used to close the schema:

```
</schema>
```

5 JPSearch Core Metadata Schema

5.1 General

JPSearch Core Metadata Schema contains four types: `PersonNameType`, `SourceType`, `PublisherType` and `JPSearchCoreType`. Moreover, in order to support `JPSearchCoreType`, several types are defined: `RightsDescriptionType`, `PlaceType`, `PersonType`, `OrganizationType`, `EventType`, `ObjectType`, `RegionOfInterestType`, `RegionLocatorType`, `ExternalDescriptionType`, `ControlledRatingTermType`, `ImageIdentifierType` and `GPSPositionType`.

NOTE See Annex A for details of the registration process for this document.

5.2 JPSearchCoreType

5.2.1 General

The `JPSearchCoreType` type is devised in order to describe the information about an image in metadata layer. At the same time, as JPSearch core metadata is utilized for image search among the set of images that are described by using heterogeneous metadata schemes, `JPSearchCoreType` contains the most important fields in JPSearch core metadata, as shown in Figure 1.

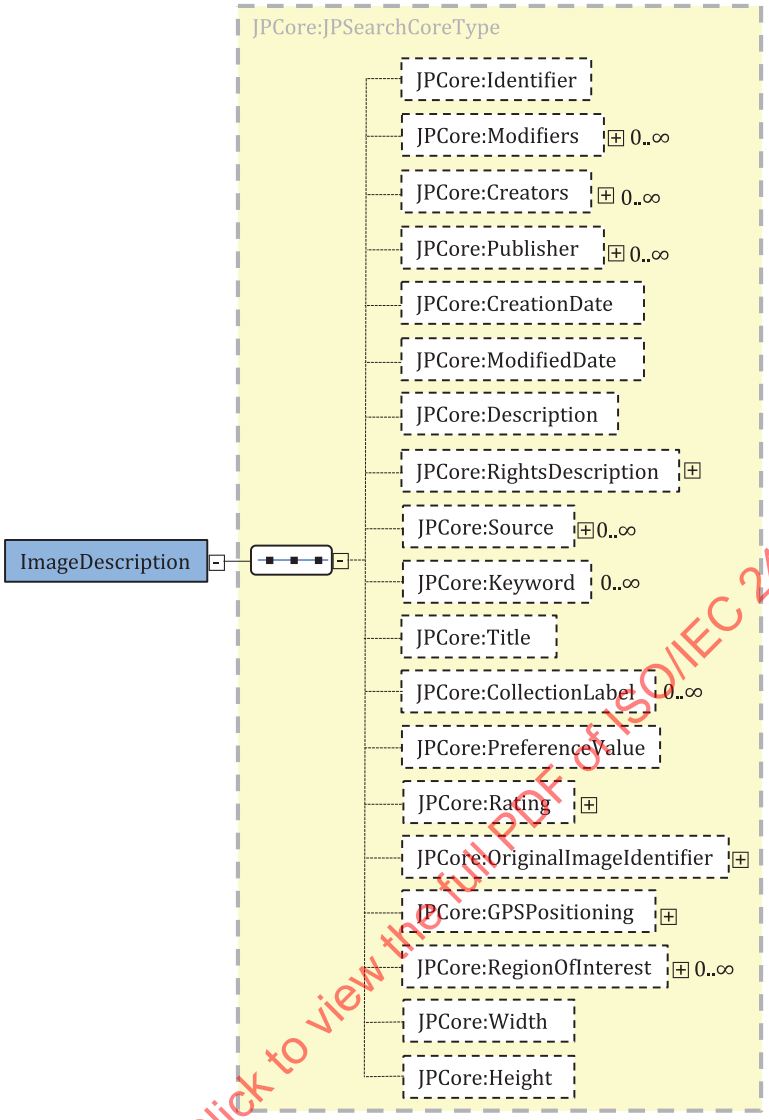

Figure 1 — Diagram representing the JPSearchCoreType

5.2.2 Syntax

```
<element name="ImageDescription" type="JPCore:JPSearchCoreType"/>
<complexType name="JPSearchCoreType">
   <sequence>
     <element name="Identifier" type="anyURI"
              minOccurs="0"/>
     <element name="Modifiers" type="JPCore:PersonNameType"
              minOccurs="0" maxOccurs="unbounded"/>
     <element name="Creators" type="JPCore:PersonNameType"
              minOccurs="0" maxOccurs="unbounded"/>
     <element name="Publisher" type="JPCore:PublisherType"
              minOccurs="0" maxOccurs="unbounded"/>
     <element name="CreationDate" type="dateTime"
              minOccurs="0"/>
     <element name="ModifiedDate" type="dateTime"
              minOccurs="0"/>
     <element name="Description" type="string"
              minOccurs="0"/>
     <element name="RightsDescription" type="JPCore:RightsDescriptionType"
              minOccurs="0"/>
     <element name="Source" type="JPCore:SourceType"
              minOccurs="0" maxOccurs="unbounded"/>
     <element name="Keyword" type="string" minOccurs="0"
              maxOccurs="unbounded"/>
     <element name="Title" type="string" minOccurs="0"/>
     <element name="CollectionLabel" type="string" minOccurs="0"
              maxOccurs="unbounded"/>
     <element name="PreferenceValue" type="integer"
              minOccurs="0"/>
     <element name="Rating"
              type="JPCore:ControlledRatingTermType"
              minOccurs="0"/>
     <element name="OriginalImageIdentifier" type="JPCore:ImageIdentifierType"
minOccurs="0"/>
     <element name="GPSPositioning"
              type="JPCore:GPSPositioningType" minOccurs="0"/>
     <element name="RegionOfInterest"
              type="JPCore:RegionOfInterestType" minOccurs="0"
              maxOccurs="unbounded"/>
     <element name="Width" type="int"/>
     <element name="Height" type="int"/>
   </sequence>
 </complexType>
```

5.2.3 Semantic

Semantics of the `JPSearchCoreType` type:

Name	Definition
`JPSearchCoreType`	Specifies information for an image.
`Identifier`	Describes an identifier of the image in the form of a URI. The identifier must be unique.
`Modifiers`	Describes a modifier's name or a list of names who changed the original image resulting in the creation of the image (optional).
`Creators`	Describes a person's name or a list of the names who created the image or made contributions in the creation of the image (optional).
`Publisher`	Describes information about the publishing people or organization of the image
`CreationDate`	Describes the date when the image is created.
`ModifiedDate`	Describes the date when the image is modified.
`Description`	Specifies the content of the image in the form of text.
`RightsDescription`	Describes the right related information by providing information about existing rights description standard, explanation concerning the standard in free text, and rights description in the form of external information or string value.
`Source`	Describes a source of the image. It can be another image or an object in the form of such as painting, book and so on.
`Keyword`	Describes a list of keywords that characterize the image (optional).
`Title`	Describes the title of the image (optional).
`CollectionLabel`	Describes user provided labels that can be used for the purpose of collection and categorization of images (optional).
`PreferenceValue`	Describes the value of the preference of the image in the form of integer value.
`Rating`	Describes the rating results that should be one of the corresponding controlled terms. The definition of the terms is provided by JPSearch.
`OriginalImageIdentifier`	Describes the identifier of the original image from which the image is created. Moreover, it can be used for the identifiers that can be created by a particular organization or method.
`GPSPositioning`	Describes the location of the place shown in the image
`RegionOfInterest`	Describes the information (e.g., content description, keywords, etc.) of a certain region within the image. Note, the whole image itself can also be described by one RegionOfInterest.

5.2.4 Example

This example shows a complete description of an imaginary image. It has a unique ID, some modifier, creator and publisher information. In addition, the creation and modification date have been annotated.

Please note, that subparts of this example are reused at the corresponding type definitions and explained in detail there.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3
    </Identifier>
  <Modifiers>
    <GivenName>Jonh</GivenName>
    <FamilyName>Smith</FamilyName>
  </Modifiers>
  <Creators>
    <GivenName>Jonh</GivenName>
    <FamilyName>Smith</FamilyName>
  </Creators>
  <Publisher>
    <PersonName>
      <GivenName>Jonh</GivenName>
      <FamilyName>Smith</FamilyName>
    </PersonName>
    <OrganizationInformation>
      <Name>University of Passau</Name>
      <Address>
        <Name>Innstrasse 43, 94034 Passau, Germany
              </Name>
        <Description>
                  Small city at the border to Austria surrounded
                  by three rivers
              </Description>
      </Address>
    </OrganizationInformation>
  </Publisher>
  <CreationDate>2001-12-17T09:30:47.0Z</CreationDate>
  <ModifiedDate>2001-12-17T09:30:47.0Z</ModifiedDate>
  <Description>This is a sample description and this part can contain
                    an arbitrary unstructured text according to the image
    </Description>
  <RightsDescription>
    <RightsDescriptionInformation>
          http://www.rdstandard.org
      </RightsDescriptionInformation>
    <Description>
          Provides a standard for rights description.
</Description>
    <ActualRightsDescriptionReference>
http://www.rdstandard.org/particulaValueRD
```

```
</ActualRightsDescriptionReference>
  </RightsDescription>
  <Source>
    <SourceElementType>Oil Painting</SourceElementType>
    <SourceElement>
      <SourceElementTitle>Mona Lisa</SourceElementTitle>
      <SourceElementDescription>Leonardo da Vinci, Louvre, France
</SourceElementDescription>
    </SourceElement>
    <CreationMethod>Photographing</CreationMethod>
    <CreationDescription>CN 450D</CreationDescription>
  </Source>
  <Keyword>Sardinia</Keyword>
  <Keyword>Italy</Keyword>
  <Keyword>50th JPEG meeting</Keyword>
  <Title>Example Instance document of the JPSearch core schema</Title>
  <CollectionLabel>JPEG Meetings</CollectionLabel>
  <PreferenceValue>1</PreferenceValue>
  <Rating>
    <LabelDefinition>urn:ratingtable:quality</LabelDefinition>
    <LabelValue>urn:ratingtable:perfect</LabelValue>
  </Rating>
  <GPSPositioning latitude="34" longitude="34" altitude="10"/>
  <RegionOfInterest>
    <RegionLocator>
      <Region dim="2"> 0 0 100 100</Region>
    </RegionLocator>
    <Description>A short description about the selected region
       </Description>
    <Keyword>plenary meeting</Keyword>
    <Title>plenary meeting</Title>
    <ContentDescription>
      <Person>
        <Name>
          <GivenName>Jonh</GivenName>
          <FamilyName>Smith</FamilyName>
        </Name>
      </Person>
      <Object>
        <Name>Laptop</Name>
        <Description>running laptop of UoP
               </Description>
        <Label>urn:laptop:id:1:2:3</Label>
      </Object>
```

```
      <Place>
        <Name>meeting room xyz</Name>
        <Description>meeting room at the sardinia JPEG meeting
                </Description>
      </Place>
      <Event>
        <Label>urn:writing:event:1:2</Label>
        <Description>writing some document
                </Description>
      </Event>
    </ContentDescription>
    <ExternalDescription>
      <TagName fromNamespace="urn:mpeg:mpeg7:schema:2004" fromNamespacePrefix="mpeg7">mpeg7:Color
Layout</TagName>
      <StructuredValue fromNamespace="urn:mpeg:mpeg7:schema:2004">
        <mpeg7:Mpeg7 xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004" xsi:schemaLocation="urn:mpeg:mpeg7:
schema:2004 M7v2schema.xsd">
          <mpeg7:DescriptionUnit xsi:type="mpeg7:ColorLayoutType">
            <mpeg7:YDCCoeff>1</mpeg7:YDCCoeff>
            <mpeg7:CbDCCoeff>2</mpeg7:CbDCCoeff>
            <mpeg7:CrDCCoeff>3</mpeg7:CrDCCoeff>
            <mpeg7:YACCoeff2>1 2</mpeg7:YACCoeff2>
            <mpeg7:CbACCoeff2>1 2</mpeg7:CbACCoeff2>
            <mpeg7:CrACCoeff2>1 2</mpeg7:CrACCoeff2>
          </mpeg7:DescriptionUnit>
        </mpeg7:Mpeg7>
      </StructuredValue>
    </ExternalDescription>
    <ExternalDescription>
      <TagName/>
      <LiteralValue/>
    </ExternalDescription>
  </RegionOfInterest>
  <Width>640</Width>
  <Height>480</Height>
</ImageDescription>
```


5.3 PersonNameType

5.3.1 General

The `PersonNameType` type support the description of human beings within images or image regions. The information that can be provided is simplified to `GivenName` and `FamilyName`.

5.3.2 Syntax

```
<complexType name="PersonNameType">
        <sequence>
             <choice maxOccurs="unbounded">
                  <element name="GivenName" type="string" />
                  <element name="FamilyName" type="string" minOccurs="0" />
            </choice>
      </sequence>
        <attribute ref="xml:lang" use="optional" />
</complexType>
```

5.3.3 Semantic

Semantics of the `PersonNameType` type:

Name	Definition
`PersonNameType`	Specifies a list of names that consist of given names or family names. The constituent names for a person's name should be described by using the same language.
`GivenName`	Describes a given name. A number of given names can be defined, if necessary.
`FamilyName`	Describes a family name(optional). A number of family names can be defined, if necessary.
`xml:lang`	Describes the language used for defined names(optional).

5.3.4 Example

Instances of the `PersonNameType` are used on several places within the core schema. The following example shows its use by the `Modifiers` tag. In general, the `PersonNameType` allows the description of family and given names of human beings.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3
    </Identifier>
  <Modifiers>
    <GivenName>John</GivenName>
    <FamilyName>Smith</FamilyName>
  </Modifiers>
</ImageDescription>
```

5.4 SourceType

5.4.1 General

The `SourceType` type specifies the source of the image when the image is created from a pre-existing artificial work or product such as painting, book, video and so on.

5.4.2 Syntax

```
<complexType name="SourceType">
    <sequence>
      <element name="SourceElementType" type="string"/>
      <element name="SourceElement" maxOccurs="unbounded">
        <complexType>
          <sequence>
            <element name="SourceElementTitle" type="string"/>
            <element name="SourceElementDescription" type="string"/>
            <element name="SourceElementIdentifier" type="anyURI" minOccurs="0"/>
            <!-- Editor's Note: the type of SourceElementIdentifier needs to be
eliminated if the image identifier type use another type rather than anyURI -->
          </sequence>
        </complexType>
      </element>
      <element name="CreationMethod" type="string"/>
      <element name="CreationDescription" type="string"/>
    </sequence>
  </complexType>
  <complexType name="SourceElementType">
</complexType>
```

5.4.3 Semantic

Semantics of the `SourceType` type:

Name	Definition
`SourceType`	Specifies the `SourceType` type that is used for describing the source that is used for the image.
`SourceElementType`	Specifies a type of image source that can be an original image or an object in the form such as painting, book, image and so on.
`SourceElement~~Object~~`	Describes the source element(s) that is used for creation of the image. In the case that the image is created by using multiple source elements, each source element is described in a SourceElement field.
`SourceElementTitle`	Specifies a title for the source element being used.
`SourceElementDescription`	Describes information for the source element that is used for creation of the image such as location, contact point, and so on.
`SourceElementIdentifier`	Specifies the identifier for the source element if exists.
`CreationMethod`	Specifies the method that is used for creation of the image. Recommended values for this field are Photographing, Film Scanning, Reflection Print Scanning, Video Still Capturing, and Computer Graphic Editing.
`CreationDescription`	Describes the information about the creation of the image. Or it can describe the technical information that is necessary for the production work of the image such as workflow parameters.

5.4.4 Example

This example shows the image created from an oil painting 'Mona Lisa' by photographing. The painting is created by 'Leonardo da Vinci' and is kept in 'Louvre, France'. The camera used for the production is 'CN 450D'.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3
    </Identifier>
  <Modifiers>
    <GivenName>Jonh</GivenName>
    <FamilyName>Smith</FamilyName>
  </Modifiers>
</ImageDescription>
```

5.5 PublisherType

5.5.1 General

The `PublisherType` type supports the description of information about the publisher which can be any public or private organization or person.

5.5.2 Syntax

```
<complexType name="PublisherType">
  <sequence>
    <choice minOccurs="0" maxOccurs="unbounded">
      <element name="PersonName" type="JPCore:PersonNameType"/>
      <element name="OrganizationInformation" type="JPCore:OrganizationType"/>
    </choice>
  </sequence>
</complexType>
```

5.5.3 Semantic

Semantics of the `PublisherType` type:

Name	Definition
`PublisherType`	Specifies information of the publisher for the image. No information can be defined. The publisher information can be either in the form of person's name or organization's description.
`PersonName`	Describes a publisher in the form of a person's name.
`OrganizationInformation`	Describes a publishing organization.

5.5.4 Example

This example describes the information given for the publisher of an image. It supports the annotation of an organization and/or a human being.

```
<ImageDescription>
    <?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3
    </Identifier>
  <Publisher>
    <PersonName>
      <GivenName>Jonh</GivenName>
      <FamilyName>Smith</FamilyName>
    </PersonName>
    <OrganizationInformation>
      <Name>University of Passau</Name>
      <Address>
        <Name>Innstrasse 43, 94034 Passau, Germany
                </Name>
        <Description>
                    Small city at the border to Austria surrounded
                    by three rivers
                </Description>
      </Address>
    </OrganizationInformation>
  </Publisher>
</ImageDescription>
```


5.6 RightsDescriptionType

5.6.1 General

The RightsDescriptionType type is used to provide information about rights description standard and corresponding rights description. Among the information, a URN value is necessary for the existing rights description standard. In addition, free text to describe the standard might be provided. Actual rights description based on the specified standard would be provided using either reference external file or embedding the description itself as a string. Existence of the rights description itself might be optional.

5.6.2 Syntax

```
<complexType name="RightsDescriptionType">
  <sequence>
    <element name="RightsDescriptionInformation" type="anyURI" minOccurs="1"
maxOccurs="1"/>
    <element name="Description" type="string" minOccurs="0"/>
    <element name="ActualRightsDescriptionReference" type="anyURI" minOccurs="1"
maxOccurs="1"/>
    <element name="ActualRightsDescription" type="string" minOccurs="0"/>
  </sequence>
</complexType>
```

5.6.3 Semantics

Semantics of the `RightsDescriptionType` type:

5.6.4 Example

Name	*Definition*
`RightsDescriptionInformation`	Specifies the location where the rights description standard is provided.
`Description`	Describes the rights description standard in the form of free text.
`ActualRightsDescriptionReference`	Specifies an external file that includes the actual rights description.
`ActualRightsDescription`	Specifies the actual rights description.

This example shows the description of a rights description providing the location where the actual rights description exists.

```
<ImageDescription>
    <Identifier>urn:unique:identifier:1:2:3
    </Identifier>
    <RightsDescriptionType>
       <RightsDescriptionInformation>
       http://www.rdstandard.org
      </RightsDescriptionInformation>
      <Description>
       Provides a standard for rights description.
      </Description>
      <ActualRightsDescriptionReference>
       http://www.rdstandard.org/particulaValueRD
      </ActualRightsDescriptionReference>
    </RightsDescriptionType>
</ImageDescription>
```

5.7 PlaceType

5.7.1 General

The `PlaceType` type supports the description of locations that are shown in the image or certain regions thereof.

5.7.2 Syntax

```
<complexType name="PlaceType">
  <sequence>
    <element name="Name" type="string" minOccurs="0"
          maxOccurs="unbounded"/>
    <element name="Description" type="string" minOccurs="0"
          maxOccurs="unbounded"/>
  </sequence>
</complexType>
```

5.7.3 Semantics

Semantics of the `PlaceType` type:

Name	*Definition*
`PlaceType`	Specifies the definition of a place description.
`Name`	Specifies the name of the place.
`Description`	Specifies a description about the place.

5.7.4 Example

This example shows the description of a certain location shown in the image. The `PlaceType` type is embedded in the `ContentDescription` tag which supports the annotation of region specific information.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <RegionOfInterest>
    <RegionLocator>
      <Region dim="2"> 0 0 100 100</Region>
    </RegionLocator>
    <Description>A short description about the selected region
        </Description>
    <Keyword>plenary meeting</Keyword>
    <Title>plenary meeting</Title>
    <ContentDescription>
      <Place>
        <Name>meeting room xyz</Name>
        <Description>meeting room at the sardinia JPEG meeting
                </Description>
      </Place>
    </ContentDescription>
  </RegionOfInterest>
</ImageDescription>
```

5.8 PersonType

5.8.1 General

The `PersonType` type describes information about a certain human being shown within an image or a certain region.

5.8.2 Syntax

```
<complexType name="PersonType">
 <sequence>
  <element name="Name" type="JPCore:PersonNameType"/>
  <element name="Affiliation" type="JPCore:OrganizationType"
           minOccurs="0" maxOccurs="unbounded"/>
  <element name="Address" type="JPCore:PlaceType" minOccurs="0"/>
  <element name="Description" type="string" minOccurs="0"/>
  <element name="Nationality" type="string" minOccurs="0"/>
 </sequence>
</complexType>
```

5.8.3 Semantics

Semantics of the `PersonType` type:

Name	*Definition*
`PersonType`	Specifies the definition of a person description.
`Name`	Specifies the name of the person.
`Affiliation`	Specifies a description about the affiliation of the person.
`Address`	Specifies the address of the person.
`Description`	Specifies a description of the person.
`Nationality`	Specifies the nationality of the person.

5.8.4 Example

The following example demonstrates the description of a human being. Information about human beings can be given within annotated regions by using the `ContentDescription` tag.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3
    </Identifier>
  <RegionOfInterest>
    <RegionLocator>
      <Region dim="2"> 0 0 100 100</Region>
    </RegionLocator>
    <ContentDescription>
      <Person>
        <Name>
          <GivenName>Jonh</GivenName>
          <FamilyName>Smith</FamilyName>
        </Name>
        <Affiliation>
          <Name/>
        </Affiliation>
        <Address>
          <Name>Innstrasse 43, 94034 Passau, Germany
                    </Name>
          <Description>Small city at the border to Austria
                                        surrounded by three rivers
                    </Description>
        </Address>
        <Description>this is a description of the person
                </Description>
        <Nationality>AUT</Nationality>
      </Person>
    </ContentDescription>
  </RegionOfInterest>
</ImageDescription>
```


5.9 OrganizationType

5.9.1 General

The `OrganizationType` type describes information about a certain organization (e.g., political, cultural, industry, etc.).

5.9.2 Syntax

```
<complexType name="OrganizationType">
   <sequence>
     <element name="Name" type="string" maxOccurs="unbounded"/>
     <element name="Address" type="JPCore:PlaceType"
              minOccurs="0"/>
   </sequence>
 </complexType>
```

5.9.3 Semantics

Semantics of the `OrganizationType` type:

Name	*Definition*
`OrganizationType`	Specifies the definition of an organization description.
`Name`	Specifies the name of the organization.
`Address`	Specifies the address of the organization.

5.9.4 Example

The `OrganizationType` type supports the description of information about public, industrial, private, ... organizations. The given example demonstrates its use within the `Publisher` tag.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3
    </Identifier>
  <Publisher>
    <OrganizationInformation>
      <Name>University of Passau</Name>
      <Address>
        <Name>Innstrasse 43, 94034 Passau, Germany
              </Name>
        <Description>Small city at the border to Austria
                                  surrounded by three rivers
              </Description>
      </Address>
    </OrganizationInformation>
  </Publisher>
</ImageDescription>
```

5.10 EventType

5.10.1 General

The `EventType` type describes information about a certain event (e.g., handshake, etc.) within a specific region of the image.

5.10.2 Syntax

```
<complexType name="EventType">
 <sequence>
  <element name="Label" type="anyURI" maxOccurs="unbounded"/>
  <element name="Description" type="string" minOccurs="0"
           maxOccurs="unbounded"/>
 </sequence>
</complexType>
```

5.10.3 Semantics

Semantics of the `EventType` type:

Name	*Definition*
`EventType`	Specifies the definition of an event description.
`Label`	Specifies the label of the event. A label is a concrete event instantiation (e.g., handshake) based on a specific taxonomy.
`Description`	Specifies the description of a label.

5.10.4 Example

The `EventType` type supports the annotation of events (e.g., handshaking, playing cards, etc.) that are shown in an image or a specific region thereof. An event is described by a `Label` and a `Description` tag.

The `Label` tag refers to a URI specifying the event by a taxonomy. In series, the `Description` tag can provide additional information.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3</Identifier>
  <RegionOfInterest>
    <RegionLocator>
      <Region dim="2"> 0 0 100 100</Region>
    </RegionLocator>
    <Description>A short description about the selected region
        </Description>
    <Keyword>plenary meeting</Keyword>
    <Title>plenary meeting</Title>
    <ContentDescription>
      <Event>
        <Label>urn:writing:event:1:2</Label>
        <Description>writing some document
                </Description>
      </Event>
    </ContentDescription>
  </RegionOfInterest>
</ImageDescription>
```

5.11 ObjectType

5.11.1 General

The `ObjectType` type describes information about a certain object (e.g., car, house, etc.) visible in an image or a certain region thereof.

5.11.2 Syntax

```
<complexType Name="ObjectType" >
 <sequence>
  <element name="Name" type="string" maxOccurs="unbounded"/>
  <element name="Description" type="string" minOccurs="0" />
  <element name="Label" type="anyURI" minOccurs="0"/>
 </sequence>
</complexType>
```

5.11.3 Semantics

Semantics of the `ObjectType` type:

Name	*Definition*
`ObjectType`	Specifies the definition of an object (e.g., car, house) visible in the image.
`Name`	Specifies the name of the object.
`Description`	Specifies the description of the object.
`Label`	Specifies the label information of the object based on a defined taxonomy.

5.11.4 Example

The `ObjectType` type allows the annotation of arbitrary objects (car, house, table, etc.) shown at a certain region within an image. Similar to the `EventType` type, an object is identified by a `Label` tag pointing to a URI of a certain taxonomy. In addition, the name and a description of the object can be given as demonstrated in the example below.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3</Identifier>
  <RegionOfInterest>
    <RegionLocator>
      <Region dim="2"> 0 0 100 100</Region>
    </RegionLocator>
    <Description>A short description about the selected region
        </Description>
    <Keyword>plenary meeting</Keyword>
    <Title>plenary meeting</Title>
    <ContentDescription>
      <Object>
        <Name>Laptop</Name>
        <Description>running laptop of UoP
                </Description>
        <Label>urn:laptop:id:1:2:3</Label>
      </Object>
    </ContentDescription>
  </RegionOfInterest>
</ImageDescription>
```


5.12 RegionOfInterestType

5.12.1 General

The `RegionOfInterestType` type describes information about a certain region in an image.

5.12.2 Syntax

```
<complexType name="RegionOfInterestType">
  <sequence>
    <element name="RegionLocator" type="JPCore:RegionLocatorType"/>
    <element name="Description" type="string" minOccurs="0"/>
    <element name="Keyword" type="string" minOccurs="0"
             maxOccurs="unbounded"/>
    <element name="Title" type="string" minOccurs="0"/>
    <element name="ContentDescription" minOccurs="0">
      <complexType>
        <sequence>
          <element name="Person" type="JPCore:PersonType"
                   minOccurs="0" maxOccurs="unbounded"/>
          <element name="Object" type="JPCore:ObjectType"
                   minOccurs="0" maxOccurs="unbounded"/>
          <element name="Place" type="JPCore:PlaceType"
                   minOccurs="0" maxOccurs="unbounded"/>
          <element name="Event" type="JPCore:EventType"
                   minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
      </complexType>
    </element>
    <element name="ExternalDescription"
             type="JPCore:ExternalDescriptonType" minOccurs="0"
             maxOccurs="unbounded"/>
  </sequence>
 </complexType>
```


5.12.3 Semantics

Semantics of the `RegionOfInterestType` type:

Name	*Definition*
`RegionOfInterestType`	Specifies the definition of a certain region in the image.
`RegionLocator`	Specifies the spatial location of the region within the image.
`Description`	Specifies the description of the region.
`Keyword`	Specifies keywords for the region.
`Title`	Specifies a title for the region.
`ContentDescription`	Specifies the semantic description of the content shown within the region.
`Person`	Specifies information about visible persons in the region.
`Object`	Specifies information about visible objects in the region
`Place`	Specifies information about the location shown in the region
`Event`	Specifies information about events.
`ExternalDescription`	Specifies a formal way in order to describe information which is not covered by the core schema and can be integrated from other standards (e.g., MPEG-7). Note, by using this tag it is not guaranteed that the referenced description can be evaluated by all JPSearch compliant systems.

5.12.4 Example

The `RegionOfInterestType` type supports the description of regions within images. For such a region, a description, keywords and a title can be specified, Furthermore, the content of the region such as occurring persons, events, or shown places can be annotated by the `ContentDescription` tag. Descriptive information that is not covered by the core schema can be integrated by using either the `UserDefinedTag` tag or the `ExternalDescription` tag.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3</Identifier>
  <RegionOfInterest>
    <RegionLocator>
      <Region dim="2"> 0 0 100 100</Region>
    </RegionLocator>
    <Description>A short description about the selected region
        </Description>
    <Keyword>plenary meeting</Keyword>
    <Title>plenary meeting</Title>
    <ContentDescription>
      <Place>
        <Name>meeting room xyz</Name>
        <Description>meeting room at the sardinia JPEG meeting
                </Description>
      </Place>
    </ContentDescription>
    <ExternalDescription>
      <TagName fromNamespace="urn:mpeg:mpeg7:schema:2004" fromNamespacePrefix="mpeg7">mpe
g7:ColorLayout</TagName>
      <StructuredValue fromNamespace="urn:mpeg:mpeg7:schema:2004">
        <mpeg7:Mpeg7 xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004" xsi:schemaLocation="urn:mpe
g:mpeg7:schema:2004 M7v2schema.xsd">
          <mpeg7:DescriptionUnit xsi:type="mpeg7:ColorLayoutType">
            <mpeg7:YDCCoeff>1</mpeg7:YDCCoeff>
            <mpeg7:CbDCCoeff>2</mpeg7:CbDCCoeff>
            <mpeg7:CrDCCoeff>3</mpeg7:CrDCCoeff>
            <mpeg7:YACCoeff2>1 2</mpeg7:YACCoeff2>
            <mpeg7:CbACCoeff2>1 2</mpeg7:CbACCoeff2>
            <mpeg7:CrACCoeff2>1 2</mpeg7:CrACCoeff2>
          </mpeg7:DescriptionUnit>
        </mpeg7:Mpeg7>
      </StructuredValue>
    </ExternalDescription>
  </RegionOfInterest>
</ImageDescription>
```

5.13 RegionLocatorType

5.13.1 General

The `RegionLocatorType` type describes information about the location of a certain region in an image.

5.13.2 Syntax

```
<complexType name="RegionLocatorType">
  <sequence>
   <element name="Region" type="JPCore:IntegerMatrixType"
            maxOccurs="unbounded"/>
  </sequence>
 </complexType>
 <complexType name="IntegerMatrixType">
  <simpleContent>
   <extension base="JPCore:listOfInteger">
     <attribute name="dim" type="positiveInteger" use="required"/>
   </extension>
  </simpleContent>
 </complexType>
 <simpleType name="listOfInteger">
    <list itemType="int"/>
 </simpleType>
```

5.13.3 Semantics

Semantics of the `RegionLocatorType` type:

Name	*Definition*
`RegionLocatorType`	Specifies the definition of the location of a certain region in the image.
`Region`	Specifies the spatial location and dimension of a region within the image.
`IntegerMatrixType`	Specifies a spatial region. A spatial region is defined by the `IntegerMatrixType` type which allows the specification of a list of positive integer values describing individual points. The amount of necessary integer values per point is defined by the `dim` attribute of `IntegerMatrixType` type. The individual points define the region where two points mean a rectangular, three points a triangle and so on.
`dim`	Attribute that defines the dimension of the individual points (e.g., 2 means that 2 integer values are required to define one point)
`listOfInteger`	Simple type that specifies a list of positive integer values which describes individual points. The amount of necessary integer values per point is defined by the `dim` attribute of the `IntegerMatrixType` type.

5.13.4 Example

The RegionLocatorType type allows the specification of regions within images. Every region can be of arbitrary shape depending on the used dimension and amount of points. For instance the given example

describes a two dimensional (the dim attribute is set to 2) region (a rectangular) by the following coordinates (0, 0), (100, 100).

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3</Identifier>
  <RegionOfInterest>
    <RegionLocator>
      <Region dim="2"> 0 0 100 100</Region>
    </RegionLocator>
  </RegionOfInterest>
</ImageDescription>
```

5.14 ExternalDescriptonType

5.14.1 General

The `ExternalDescriptonType` type allows embedding metadata fields not defined by JPEG but used in external user communities or defined by external metadata formats (e.g., MPEG-7 descriptors or descriptor schemes).

5.14.2 Syntax

```
<complexType name="ExternalDescriptonType">
   <sequence>
     <element name="TagName">
       <complexType mixed="true">
         <attribute name="fromNamespace" type="anyURI"
                    use="optional"/>
         <attribute name="fromNamespacePrefix" type="string"
                    use="optional"/>
       </complexType>
     </element>
     <choice>
       <element name="LiteralValue">
         <complexType mixed="true">
           <attribute name="fromControlledVocabularyOfLiterals"
                      type="anyURI" use="optional"/>
           <attribute name="fromGrammarOfLiterals" type="anyURI"
                      use="optional"/>
         </complexType>
       </element>
       <element name="ResourceValue">
         <complexType mixed="true">
           <attribute name="fromNamespace" type="anyURI"
                      use="optional"/>
         </complexType>
       </element>
      <element name="StructuredValue">
        <complexType mixed="true">
          <sequence>
            <any namespace="##any"/>
          </sequence>
          <attribute name="fromNamespace" type="anyURI" use="required"/>
        </complexType>
      </element>
     </choice>
   </sequence>
 </complexType>
```

5.14.3 Semantics

Semantics of the `ExternalDescriptonType` type:

Name	*Definition*
`ExternalDescriptonType`	Allows the integration of components of other metadata formats (e.g., MPEG-7 descriptors or descriptor schemes). It can be 1) a flat literal value, 2) a resource value (identified by a URI) or 3) a structured XML value from an external namespace.
`TagName`	Specifies the name of the tag.
`fromNamespace`	Optional attribute which specifies a namespace to which the tag name belongs.
`fromNamespacePrefix`	Optional attribute which specifies the prefix of the namespace to which the tag belongs.
`LiteralValue`	Specifies a literal value for the tag.
`fromControlledVocabularyOfLiterals`	Optional attribute which specifies, if there exists one, the URI of a controlled vocabulary to which the literal value belongs (e.g. MIME types)
`fromGrammarOfLiterals`	Optional attribute which specifies, if there exists one, the URI of a grammar of literals to which the literal value belongs (e.g. a datetime format)
`ResourceValue`	Specifies the URI pointing to a resource.
`fromNamespace`	Optional attribute which specifies, if there exists one, the URI of a namespace to which the resource value belongs.
`StructuredValue`	Specifies a structured XML value from an external namespace.
`fromNamespace`	Optional attribute which specifies, if there exists one, the URI of a namespace to which the description belongs.

5.14.4 Example

The following example shows a UserDefinedTag being used to wrap an Mpeg7 description.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3</Identifier>
  <RegionOfInterest>
    <RegionLocator>
      <Region dim="2"> 0 0 100 100</Region>
    </RegionLocator>
    <ExternalDescription>
      <TagName fromNamespace="urn:mpeg:mpeg7:schema:2004" fromNamespacePrefix="mpeg7">mpe
g7:ColorLayout</TagName>
      <StructuredValue fromNamespace="urn:mpeg:mpeg7:schema:2004">
        <mpeg7:Mpeg7 xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004" xsi:schemaLocation="urn:mpe
g:mpeg7:schema:2004 M7v2schema.xsd">
          <mpeg7:DescriptionUnit xsi:type="mpeg7:ColorLayoutType">
            <mpeg7:YDCCoeff>1</mpeg7:YDCCoeff>
            <mpeg7:CbDCCoeff>2</mpeg7:CbDCCoeff>
            <mpeg7:CrDCCoeff>3</mpeg7:CrDCCoeff>
            <mpeg7:YACCoeff2>1 2</mpeg7:YACCoeff2>
            <mpeg7:CbACCoeff2>1 2</mpeg7:CbACCoeff2>
            <mpeg7:CrACCoeff2>1 2</mpeg7:CrACCoeff2>
          </mpeg7:DescriptionUnit>
        </mpeg7:Mpeg7>
      </StructuredValue>
    </ExternalDescription>
    <ExternalDescription>
      <TagName fromNamespace="urn:medical" fromNamespacePrefix="mpeg7">PatientName</
TagName>
      <LiteralValue>John Smith</LiteralValue>
    </ExternalDescription>
    <ExternalDescription>
      <TagName fromNamespace="urn:medical" fromNamespacePrefix="mpeg7">PatientName</
TagName>
      <ResourceValue>http://www.patients.org/jsmith</ResourceValue>
    </ExternalDescription>
  </RegionOfInterest>
</ImageDescription>
```


5.15 ControlledRatingTermType

5.15.1 General

The `ControlledRatingTermType` type specifies rating value and the definition of terms used for image rating. The terms for rating should be one of controlled terms that are provided by JPSearch.

5.15.2 Syntax

```
<complexType name="ControlledRatingTermType">
  <sequence>
    <element name="LabelDefinition" type="anyURI"/>
    <element name="LabelValue" type="string"/>
  </sequence>
</complexType>
```

5.15.3 Semantics

Semantics of the `ControlledRatingTermType` type:

Name	*Definition*
`ControlledRatingTermType`	Specifies the `ControlledRatingTermType` type that is used for rating and corresponding term definition.
`LabelDescription`	Specifies the location of the term definition.
`LabelValue`	Specifies the rating value for the image. The rating value should be defined in the location given in LavelDefinition.

5.15.4 Example

This example shows the image has rating label 'General' that is defined in 'urn:unique:ratingdefinition:1'.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3
  </Identifier>
  <Rating>
    <LabelDefinition>urn:ratingdefinition:1
     </LabelDefinition>
    <LabelValue>General</LabelValue>
  </Rating>
</ImageDescription>
```

5.16 ImageIdentifierType

5.16.1 General

The `ImageIdentifierType` type provides the necessary information so that the origination of a particular image can be described. Moreover, identifiers that are created by a particular organization or method can be provided.

5.16.2 Syntax

```
<complexType name="ImageIdentifierType">
    <sequence>
      <element name="OriginationOfID" type="anyURI"/>
      <element name="Identifier" type="string"/>
    </sequence>
  </complexType>
```

5.16.3 Semantics

Semantics of the `ImageIdentifierType` type:

Name	Definition
`OriginationOfID`	Specifies either the organization or person who issued the identifier, or the location where the method used to create the identifier can be found.
`Identifier`	Specifies the identifier issued by the particular organization or the method specified by the `OriginationOfID`.

5.16.4 Example

This example shows the usage of OriginalImageIdentifier in order to describe identifier that is created by a particular user.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>
urn:unique:identifier:1:2:3
    </Identifier>
  <OriginalImageIdentifier>
    <OriginationOfID>
http://www.newidentifier.com/ID3_Algorithm
</OriginationOfID>
    <Identifier>
        Id3::ABCED1234ABCDFFCD
</Identifier>
  </OriginalImageIdentifier>
</ImageDescription>
```


5.17 GPSPositioningType

5.17.1 General

The `GPSPositioningType` type provides information in order to describe the location of a certain place.

5.17.2 Syntax

```
<complexType name="GPSPositioningType">
  <attribute name="longitude" use="required">
    <simpleType>
      <restriction base="double">
        <minInclusive value="-180.0"/>
        <maxInclusive value="180.0"/>
      </restriction>
    </simpleType>
  </attribute>
  <attribute name="latitude" use="required">
    <simpleType>
      <restriction base="double">
        <minInclusive value="-90.0"/>
        <maxInclusive value="90.0"/>
      </restriction>
    </simpleType>
  </attribute>
  <attribute name="altitude" type="double" use="optional"/>
</complexType>
```

5.17.3 Semantics

Semantics of the `GPSPositioningType` type:

Name	*Definition*
`GPSPositioningType`	Specifies the geographic position of a place shown within the image.
`longitude`	Specifies the longitude of the place.
`latitude`	Specifies the latitude of the place
`altitude`	Specifies the altitude of the place.

5.17.4 Example

This example shows the use of GPS coordinates in order to describe location information.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImageDescription xmlns="JPSearch:schema:coremetadata" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:coremetadata jpcore.xsd">
  <Identifier>urn:unique:identifier:1:2:3</Identifier>
  <GPSPositioning latitude="34" longitude="34" altitude="10"/>
</ImageDescription>
```

6 Management of core schema and translation rules

6.1 General

NOTE See Annex A for details of the registration process for this document.

6.2 Wrapper of the schema

The description examples and syntax of description tools specified in this document assume that a schema wrapper is provided which identifies the XML Schema namespace (XML Schema) and JPSearch namespace for the management part:

```
<schema xmlns="http://www.w3.org/2001/XMLSchema"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:jps_mngt="JPSearch:schema:management"
           xmlns:jpqf="urn:jpeg:jpqf:schema:2008"
           targetNamespace="JPSearch:schema:management"
           elementFormDefault="qualified">
   <import namespace="urn:jpeg:jpqf:schema:2008"
           schemaLocation="JPQF_SFCD.xsd"/>
```

The following tag is used to close the schema:

```
</schema>
```

6.3 Root element

6.3.1 General

The `SchemaManagement` element serves as the root element of the JPSearch Management Process. The root element shall be used as the topmost element in all messages transmitted. This applies on all operations in the corresponding message direction (input or output).

6.3.2 Syntax

```
<element name="SchemaManagement" type="jps_mngt:ManagementType"/>

<complexType name="ManagementType">
  <choice>
 <element name="Register">
   <complexType>
    <choice>
      <element name="Input" type="jps_mngt:RegisterInputType"/>
      <element name="Output" type="jps_mngt:RegisterOutputType"/>
    </choice>
   </complexType>
 </element>
 <element name="Request">
   <complexType>
    <choice>
      <element name="Input" type="jps_mngt:RequestInputType"/>
      <element name="Output" type="jps_mngt:RequestOutputType"/>
    </choice>
   </complexType>
 </element>
 <element name="Replace">
   <complexType>
    <choice>
      <element name="Input" type="jps_mngt:ReplaceInputType" />
      <element name="Output" type="jps_mngt:ReplaceOutputType" />
    </choice>
   </complexType>
 </element>
  </choice>
  <attribute name="jpsearchID" type="anyURI" use="required"/>
</complexType>
```


6.3.3 Semantics

Semantics of the `ManagementType` type:

Name	*Definition*
`SchemaManagement`	Serves as the root element of the JPSearch Management Process. The `SchemaManagement` element shall be used as the topmost element in any JPsearch management message.
`ManagementType`	Specifies the syntax of the root element. Within this element, either one of the following operations can be chosen: Register, Request and Replace.
`Register`	Allows the registration of a schema, its translation rules and additional information.
`Input`	Wraps the user register request specified using the Register Input Format, which is to be sent from the service provider to the JPSearch schema authority.

Name	*Definition*
Output	Wraps the result (e.g., error message, ack, etc.) of the register request specified using the Register Output Format.
Request	Allows the request of registered schema information and translation rules.
Input	Specifies an input request for a schema or translation rules
Output	Specifies the output of a request for a schema or translation rules
Replace	Specifies the replacement functionality of schema related information such as provider information and translation rules.
Input	Specifies the input of a replacement request.
Output	Specifies the output of a replacement request.
jpsearchID	Specifies a unique identifier which is assigned to every message sent between a requester and a responder.

6.3.4 Example

This example shows a number of cases according to different schema management operations. The first example shows a scenario in which a new schema 'DC-Based-Schema' which is designed by 'Schema Creator' is being registered with two translation rules.

EXAMPLE 1 In order to simplify the code, the namespace is assumed as 'urn:JPSearch jps_mngt'.

```
<SchemaManagement jpsearchID= "JPS-MSG-01"
                  xmlns="JPSearch:schema:management">
   <Register>
      <Input>
         <ProviderInformation>
           <Name>Schema Creator</Name>
           <Description>Image schema Developer based on MPEG</Description>
           <Contact>
              <EMail>creator-email@schema.com</EMail>
              <PostalAddress> 2 1st St., Los Angeles, CA 876554, USA</PostalAddress>
              <Phone>212-1234-4567</Phone>
              <lang>eng</lang>
           </Contact>
              <ProviderCapability xsi:type="QueryCapabilityType">
                <ServiceCapability>
                  <jpqf:SupportedQueryTypes href="urn:X:Y:Z"/>
                </ServiceCapability>
              </ProviderCapability>
              <ProviderCapability xsi:type="BenchmarkCapabilityType"
                                  name="Example Benchmark"
                                  reference="urn:unique_identifier">
                <QualityRating queryType="urn:X:Y:Z" rating="0.3"/>
              </ProviderCapability>
```

```
        </ProviderInformation>
        <Schema>
           <SchemaInformation>
              <Description>new version 2009</Description>
              <SpecificationName>DC-Based-Schema</SpecificationName>
              <Version>1.0</Version>
              <Identifier>urn:DC-Based-Schema:1</Identifier>
              <Location>urn:DC-Based-Schema:registry:1</Location>
           </SchemaInformation>
           <TranslationRules>
              <TranslationRule>
                  <!-- example of one-to-one translation -->
                  <FromField>DCB-Field 1</FromField>
                  <ToField>Core-Field K</ToField>
              </TranslationRule>
              <TranslationRule>
                  <!-- example of one-to-one translation -->
                  <FromField>DCB-Field 2</FromField>
                  <ToField>Core-Field K+1</ToField>
              </TranslationRule>
              <fromFormat>urn:DC-Based-Schema:1</fromFormat>
              <toFormat>urn:JPSearchCore:1</toFormat>
           </TranslationRules>
        </Schema>
      </Input>
   </Register>
</SchemaManagement>
```

EXAMPLE 2 Result when a new schema 'DC-Based-Schema' is registered with the appropriate translation rules.

```
<SchemaManagement jpsearchID="JPS-MSG-02" xmlns="JPSearch:schema:management">
   <Register>
      <Output>
         <Identifier>urn:DC-Based-Schema:1</Identifier>
         <Status>
            <Code>status-code-x</Code>
            <Description>registered and being evaluated</Description>
         </Status>
      </Output>
   </Register>
</SchemaManagement>
```

EXAMPLE 3 Scenario when schema information is requested.

```
<SchemaManagement jpsearchID="JPS-MSG-03"
                  xmlns="JPSearch:schema:management">
   <Request>
      <Input>
         <SchemaID>urn:DC-Based-Schema:1</SchemaID>
      </Input>
   </Request>
</SchemaManagement>
```

EXAMPLE 4 Output message for a request input message.

```
<SchemaManagement jpsearchID="JPS-MSG-04"
                  xmlns="JPSearch:schema:management">
   <Request>
      <Output>
         <ResultItem>
            <Identifier>urn:DC-Based-Schema:1</Identifier>
            <Location>urn:DC-Based-Schema:registry:1</Location>
            <ProviderInformation>
              <Name>Schema Creator</Name>
              <Description>Image schema Developer based on MPEG
              </Description>
              <Contact>
                 <EMail>creator-email@schema.com</EMail>
                 <PostalAddress>2 1st St., Los Angeles, CA 876554, USA
                 </PostalAddress>
                 <Phone>212-1234-4567</Phone>
                 <lang>eng</lang>
              </Contact>
              <ProviderCapability xsi:type="QueryCapabilityType">
                <ServiceCapability>
                 <jpqf:SupportedQueryTypes href="urn:X:Y:Z"/>
                </ServiceCapability>
              </ProviderCapability>
              <ProviderCapability xsi:type="BenchmarkCapabilityType"
                                  name="Example Benchmark"
                                  reference="urn:unique_identifier">
                 <QualityRating queryType="urn:X:Y:Z" rating="0.3"/>
              </ProviderCapability>
            </ProviderInformation>
         </ResultItem>
      </Output>
   </Request>
</SchemaManagement>
```

EXAMPLE 5 Input scenario for replace operation that changes a translation rule.

```
<SchemaManagement jpsearchID="JPS-MSG-05"
                  xmlns="JPSearch:schema:management">
   <Replace>
      <Input>
         <SchemaID>urn:DC-Based-Schema:1</SchemaID>
         <TranslationRules>
              <TranslationRule>
                   <!-- example of one-to-one translation -->
                   <FromField>DCB-Field 1</FromField>
                   <ToField>Core-Field K+3</ToField>
              </TranslationRule>
              <fromFormat>urn:DC-Based-Schema:1</fromFormat>
              <toFormat>urn:JPSearchCore:1</toFormat>
         </TranslationRules>
      </Input>
   </Replace>
</SchemaManagement>
```

EXAMPLE 6 Output scenario for the above input message.

```
<SchemaManagement jpsearchID="JPS-MSG-06"
                  xmlns="JPSearch:schema:management">
   <Replace>
      <Output>
         <Identifier>urn:DC-Based-Schema:1</Identifier>
         <Status>
            <Code>status-code-y</Code>
            <Description>new trans. Rule registered and being evaluated
            </Description>
         </Status>
      </output>
   </Replace>
</SchemaManagement>
```


6.4 RegisterInputType

6.4.1 General

The `RegisterInputType` type provides information that is required during the registration process of schema and translation rules.

6.4.2 Syntax

```
<complexType name="RegisterInputType">
    <sequence>
        <element name="ProviderInformation"
        type="jps_mngt:ProviderInformationType"/>
         <element name="Schema" type="jps_mngt:SchemaType" minOccurs="0"/>
    </sequence>
</complexType>
```

6.4.3 Semantics

Semantics of the `RegisterInputType` type:

Name	*Definition*
`RegisterInputType`	Specifies the RegisterInputType type which is used for describing all information that is necessary during the registration of schemas and translation rules.
`ProviderInformation`	Specifies the information that identifies the provider of the schema.
`Schema`	Specifies details about the registered schema and its translation rules

6.4.4 Example

The following example shows a scenario in which a new schema 'DC-Based-Schema' which is designed by 'Schema Creator' is being registered with a translation rule.

```
<SchemaManagement jpsearchID="JPS-MSG-01"
                xmlns="JPSearch:schema:management">
  <Register>
    <Input>
      <ProviderInformation>
        <Name>Schema Creator</Name>
        <Description>Image schema Developer based on MPEG</Description>
        <Contact>
          <EMail>creator-email@schema.com</EMail>
          <PostalAddress> 2 1st St., Los Angeles, CA 876554, USA
          </PostalAddress>
          <Phone>212-1234-4567</Phone>
          <lang>eng</lang>
        </Contact>
          <ProviderCapability xsi:type="QueryCapabilityType">
            <ServiceCapability>
              <jpqf:SupportedQueryTypes href="urn:X:Y:Z"/>
            </ServiceCapability>
          </ProviderCapability>
          <ProviderCapability xsi:type="BenchmarkCapabilityType"
                          name="Example Benchmark"
                          reference="urn:unique identifier">
            <QualityRating queryType="urn:X:Y:Z" rating="0.3"/>
          </ProviderCapability>
      </ProviderInformation>
      <Schema>
        <SchemaInformation>
          <Description>new version 2009</Description>
          <SpecificationName>DC-Based-Schema</SpecificationName>
          <Version>1.0</Version>
          <Identifier>urn:DC-Based-Schema:1</Identifier>
          <Location>urn:DC-Based-Schema:registry:1</Location>
        </SchemaInformation>
        <TranslationRules>
          <TranslationRule>
            <!-- example of one-to-one translation -->
            <FromField>DCB-Field 1</FromField>
            <ToField>Core-Field K</ToField>
          </TranslationRule>
          <fromFormat>urn:DC-Based-Schema:1</fromFormat>
          <toFormat>urn:JPSearchCore:1</toFormat>
        </TranslationRules>
      </Schema>
    </Input>
  </Register>
</SchemaManagement>
```

6.5 RequestInputType

6.5.1 General

The RequestInputType type allows the formulation of requests of registered schema and translation rules.

6.5.2 Syntax

```
<complexType name="RequestInputType" >
  <choice>
    <element name="SchemaID" type="anyURI" />
    <element name="TranslationRuleFrom" type="anyURI" />
    <element name="TranslationRuleTo" type="anyURI" />
  </choice>
</complexType>
```

6.5.3 Semantics

Semantics of the `RequestInputType` type:

Name	*Definition*
`RequestInputType`	Specifies the `RequestInputType` type which is used for describing all information that is necessary during the request of schemas and/or translation rules.
`SchemaID`	Requests the schema information that is registered by the specified URI.
`TranslationRuleFrom`	Requests the information for the translation rules that are registered for the translation between a proprietary schema to the core schema. The proprietary schema is identified by its schema ID.
`TranslationRuleTo`	Requests the information for the translation rules that are registered for the translation between the core schema to a proprietary schema. The proprietary schema is identified by its schema ID.

6.5.4 Example

The following example shows a message that requests information when the source schema for translation is given.

```
<SchemaManagement jpsearchID="JPS-MSG-07"
                  xmlns="JPSearch:schema:management">
   <Request>
      <Input>
         <TranslationRuleFrom>urn:DC-Based-Schema:1
         </TranslationRuleFrom>
      </Input>
   </Request>
</SchemaManagement>
```

6.6 RequestOutputType

6.6.1 General

The `RequestOutputType` type specifies the result format for a schema or translation rule request.

6.6.2 Syntax

```
<complexType name="RequestOutputType" >
  <sequence>
   <element name="ResultItem" maxOccurs="unbounded">
    <complexType>
      <sequence>
      <element name="Identifier" type="anyURI" />
      <element name="Location" type="anyURI" maxOccurs="unbounded" />
      <element name="ProviderInformation"
                          type="jps_mngt:ProviderInformationType" />
      </sequence>
    </complexType>
   </element>
  </sequence>
</complexType>
```

6.6.3 Semantics

Semantics of the `RequestOutputType` type:

Name	*Definition*
`RequestOutputType`	Specifies the `RequestOutputType` type which describes the information that is returned for an input request.
`ResultItem`	Specifies the container for one piece of information that fits the incoming request.
`Identifier`	Specifies the ID of either the schema or the translation rules that have been requested.
`Location`	Specifies the location where the schema or the translation rules can be obtained.
`ProviderInformation`	Specifies the provider information which corresponds to the requested schema or translation rules.

6.6.4 Example

The following example shows a resulting output message for a request input message.

```
<SchemaManagement jpsearchID="JPS-MSG-04"
                  xmlns="JPSearch:schema:management">
   <Request>
      <Output>
         <ResultItem>
            <Identifier>urn:DC-Based-Schema:1</Identifier>
            <Location>urn:DC-Based-Schema:registry:1</Location>
            <ProviderInformation>
              <Name>Schema Creator</Name>
              <Description>Image schema Developer based on MPEG
              </Description>
              <Contact>
                 <EMail>creator-email@schema.com</EMail>
                 <PostalAddress> 2 1st St., Los Angeles, CA 876554, USA
                 </PostalAddress>
                 <Phone>212-1234-4567</Phone>
                 <lang>eng</lang>
              </Contact>
              <ProviderCapability xsi:type="QueryCapabilityType">
                <ServiceCapability>
                  <jpqf:SupportedQueryTypes href="urn:X:Y:Z"/>
                </ServiceCapability>
              </ProviderCapability>
              <ProviderCapability xsi:type="BenchmarkCapabilityType"
                                  name="Example Benchmark"
                                  reference="urn:unique_identifier">
                  <QualityRating queryType="urn:X:Y:Z" rating="0.3"/>
              </ProviderCapability>
            </ProviderInformation>
         </ResultItem>
      </Output>
   </Request>
</SchemaManagement>
```


6.7 ProviderInformationType

6.7.1 General

The `ProviderInformationType` type provides information in order to identify a certain provider.

6.7.2 Syntax

```
<complexType name="ProviderInformationType">
    <sequence>
        <element name="Name" type="string"/>
        <element name="Description" type="string" minOccurs="0"/>
        <element name="Contact" type="jps_mngt:ContactType"/>
        <element name="ProviderCapability"
             type="jps_mngt:ProviderCapabilityType" minOccurs="2"
             maxOccurs="unbounded"/>
    </sequence>
</complexType>

<complexType name="ProviderCapabilityType" abstract="true">
</complexType>
```

6.7.3 Syntax

Name	*Definition*
`ProviderInformationType`	Specifies a `ProviderInformationType` type which is used for describing all information about a certain provider.
`Name`	Specifies the name of a provider.
`Description`	Specifies details about a provider.
`Contact`	Specifies contact information of a provider.
`ProviderCapability`	Specifies a placeholder for capability description of the provider. The description contains query capabilities and benchmark capabilities. In addition, the standard is open to integrate domain specific capabilities. All introduced capability types shall inherit from the abstract `ProviderCapabilityType` type. As the `ProviderCapabilityType` type is abstract and extended by several subtypes, the `QueryCapabililityType` type and the `BenchmarkCapabilityType` type shall be present in every instance document.

6.7.4 Example

In the following example, the usage of `ProviderInformationType` is demonstrated within the element ProviderInformation.

```
<SchemaManagement jpsearchID="JPS-MSG-04"
                  xmlns="JPSearch:schema:management">
   <Request>
      <Output>
         <ResultItem>
            <Identifier>urn:DC-Based-Schema:1</Identifier>
            <Location>urn:DC-Based-Schema:registry:1</Location>
            <ProviderInformation>
              <Name>Schema Creator</Name>
              <Description>Image schema Developer based on MPEG
              </Description>
              <Contact>
                 <EMail>creator-email@schema.com</EMail>
                 <PostalAddress> 2 1st St., Los Angeles, CA 876554, USA
                 </PostalAddress>
                 <Phone>212-1234-4567</Phone>
                 <lang>eng</lang>
              </Contact>
              <ProviderCapability xsi:type="QueryCapabilityType">
                <ServiceCapability>
                  <jpqf:SupportedQueryTypes href="urn:X:Y:Z"/>
                </ServiceCapability>
              </ProviderCapability>
              <ProviderCapability xsi:type="BenchmarkCapabilityType"
                                  name="Example Benchmark"
                                  reference="urn:unique_identifier">
                 <QualityRating queryType="urn:X:Y:Z" rating="0.3"/>
              </ProviderCapability>
            </ProviderInformation>
         </ResultItem>
      </Output>
   </Request>
</SchemaManagement>
```


6.8 ContactType

6.8.1 General

The `ContactType` type provides information in order to contact a certain provider.

6.8.2 Syntax

```
<complexType name="ContactType">
    <sequence>
        <element name="EMail" type="string"/>
        <element name="PostalAddress" type="string" minOccurs="0"/>
        <element name="Phone" type="string" minOccurs="0"/>
    </sequence>
    <attribute ref="xml:lang" use="optional"/>
</complexType>
```

6.8.3 Semantics

Semantics of the `ContactType` type:

Name	*Definition*
`ContactType`	Specifies the `ContactType` type which is used for describing all necessary contact information about a certain provider.
`EMail`	Specifies the email address of a provider.
`PostalAddress`	Specifies the postal address about a provider.
`Phone`	Specifies the phone information of a provider.

6.8.4 Example

In the following example, the usage of `ContactType` is demonstrated within the element Contact.

```
<SchemaManagement jpsearchID="JPS-MSG-08"
                  xmlns="JPSearch:schema:management">
   <Register>
      <Input>
         <ProviderInformation>
           <Name>Schema Creator</Name>
           <Description>Image schema Developer based on MPEG</Description>
           <Contact>
              <EMail>creator-email@schema.com</EMail>
              <PostalAddress> 2 1st St., Los Angeles, CA 876554, USA
              </PostalAddress>
              <Phone>212-1234-4567</Phone>
              <lang>eng</lang>
           </Contact>
              <ProviderCapability xsi:type="QueryCapabilityType">
                <ServiceCapability>
                  <jpqf:SupportedQueryTypes href="urn:X:Y:Z"/>
                </ServiceCapability>
              </ProviderCapability>
              <ProviderCapability xsi:type="BenchmarkCapabilityType"
                                  name="Example Benchmark"
                                  reference="urn:unique_identifier">
                 <QualityRating queryType="urn:X:Y:Z" rating="0.3"/>
              </ProviderCapability>
        </ProviderInformation>
      </Input>
   </Register>
</SchemaManagement>
```

6.9 QueryCapabilityType

6.9.1 General

The `QueryCapabilityType` type extends the `ProviderCapabilityType` and describes the service capabilities of the provider. The service capability type allows the description of supported query types, supported metadata formats, etc. See the service capability description of the JPEG Query Format for detailed information.

6.9.2 Syntax

```
<complexType name="QueryCapabilityType">
  <complexContent>
      <extension base="jps_mngt:ProviderCapabilityType">
        <sequence>
          <element name="ServiceCapability"
                type="jpqf:CapabilityType"/>
          </sequence>
       </extension>
  </complexContent>
</complexType>
```

6.9.3 Semantics

Semantics of the `QueryCapabilityType` type:

Name	*Definition*
`QueryCapabilityType`	Specifies the supported retrieval capabilities of the provider.
`ServiceCapability`	Specifies the retrieval capabilities of a provider. The service capability type allows the description of supported query types, supported metadata formats, etc. See the service capability description of the JPEG Query Fromat for detailed information.

6.9.4 Example

In the following example, the usage of `QueryCapabilityType` is demonstrated within the element ProviderInformation.

```
<SchemaManagement jpsearchID="JPS-MSG-04"
                  xmlns="JPSearch:schema:management">
   <Request>
      <Output>
         <ResultItem>
            <Identifier>urn:DC-Based-Schema:1</Identifier>
            <Location>urn:DC-Based-Schema:registry:1</Location>
            <ProviderInformation>
              <Name>Schema Creator</Name>
              <Description>Image schema Developer based on MPEG
              </Description>
              <Contact>
                 <EMail>creator-email@schema.com</EMail>
                 <PostalAddress> 2 1st St., Los Angeles, CA 876554, USA
                 </PostalAddress>
                 <Phone>212-1234-4567</Phone>
                 <lang>eng</lang>
              </Contact>
              <ProviderCapability xsi:type="QueryCapabilityType">
                <ServiceCapability>
                  <jpqf:SupportedQueryTypes href="urn:X:Y:Z"/>
                </ServiceCapability>
              </ProviderCapability>
              <ProviderCapability xsi:type="BenchmarkCapabilityType"
                                  name="Example Benchmark"
                                  reference="urn:unique_identifier">
                 <QualityRating queryType="urn:X:Y:Z" rating="0.3"/>
              </ProviderCapability>
            </ProviderInformation>
         </ResultItem>
      </Output>
   </Request>
</SchemaManagement>
```


6.10 BenchmarkCapabilityType

6.10.1 General

The `BenchmarkCapabilityType` type extends the `ProviderCapabilityType` and describes the evaluation results of a provider for its supported filter conditions. The particular type of benchmark being used is provided together with the rating achieved. For instance, the retrieval quality of the provided search conditions such as QueryByMedia, TemporalQuery, etc. shall be defined. These measurements are being used to classify the service provided by the database accordingly. It allows for purposefully allocation of databases addressed corresponding user requirements.

6.10.2 Syntax

```
<complexType name="BenchmarkCapabilityType">
  <complexContent>
      <extension base="jps_mngt:ProviderCapabilityType">
        <sequence>
          <element name="QualityRating"
                type="jps_mngt:QualityRatingType" minOccurs="1"
                maxOccurs="unbounded"/>
        </sequence>
        <attribute name="name" type="string"/>
        <attribute name="reference" type="anyURI"/>
      </extension>
  </complexContent>
</complexType>

<complexType name="QualityRatingType">
  <attribute name="queryType" type="anyURI"/>
  <attribute name="rating" type="float"/>
</complexType>
```

6.10.3 Semantics

Semantics of the `BenchmarkCapabilityType` type:

Name	*Definition*
`BenchmarkCapabilityType`	Specifies the benchmark description of a provider.
`QualityRating`	Specifies the quality rating for one filter condition of the service provider. This filter condition can be a QueryByMedia or TemporalQuery, etc. query type. See below for detailed information of the `QualityRatingType` type.
`name`	Specifies name of the benchmark
`reference`	Specifies the location or unique identifier of the benchmark
`QualityRatingType`	Specifies the quality rating for one filter condition.
`queryType`	Specifies the URN of the filter condition (e.g., urn:mpeg:mpqf:cs:ServiceCapabilityCS:2008:100.3.6.1 for QueryByMedia)
`rating`	Specifies the rating for the filter condition. The rating is calculated as the percentage of successful executions of queries that belong to the given filter condition. Note an exact formula depends on the used benchmark system and should be specified there.

6.10.4 Example

In the following example, the usage of `BenchmarkCapabilityType` is demonstrated within the element ProviderInformation.

```
<SchemaManagement jpsearchID="JPS-MSG-04"
                  xmlns="JPSearch:schema:management">
   <Request>
      <Output>
         <ResultItem>
            <Identifier>urn:DC-Based-Schema:1</Identifier>
            <Location>urn:DC-Based-Schema:registry:1</Location>
            <ProviderInformation>
              <Name>Schema Creator</Name>
              <Description>Image schema Developer based on MPEG
              </Description>
              <Contact>
                 <EMail>creator-email@schema.com</EMail>
                 <PostalAddress> 2 1st St., Los Angeles, CA 876554, USA
                 </PostalAddress>
                 <Phone>212-1234-4567</Phone>
                 <lang>eng</lang>
              </Contact>
              <ProviderCapability xsi:type="QueryCapabilityType">
                <ServiceCapability>
                  <jpqf:SupportedQueryTypes href="urn:X:Y:Z"/>
                </ServiceCapability>
              </ProviderCapability>
              <ProviderCapability xsi:type="BenchmarkCapabilityType"
                                  name="Example Benchmark"
                                  reference="urn:unique_identifier">
                 <QualityRating queryType="urn:X:Y:Z" rating="0.3"/>
              </ProviderCapability>
            </ProviderInformation>
         </ResultItem>
      </Output>
   </Request>
</SchemaManagement>
```


6.11 ExtensionCapabilityType

6.11.1 General

The `ExtensionCapabilityType` type extends the `ProviderCapabilityType` and provides a container for any description based on a specific schema specified by the namespace declaration within the description.

6.11.2 Syntax

```
<complexType name="ExtensionCapabilityType">
  <complexContent>
    <extension base="jps_mngt:ProviderCapabilityType">
      <sequence>
       <any namespace="##any" processContents="lax" minOccurs="0"
            maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

6.11.3 Semantics

Semantics of the `ExtensionCapabilityType` type:

Name	*Definition*
`ExtensionCapabilityType`	Specifies the extensibility pattern for domain or provider specific technical information. The type is a container for any description based on a specific schema specified by the namespace declaration within the description.

6.11.4 Example

In the following example, the usage of `ExtensionCapabilityType` is demonstrated within the element ProviderInformation. In this example, the provider integrated capability descriptions about its network connections.

```
<SchemaManagement jpsearchID="JPS-MSG-04"
                  xmlns="JPSearch:schema:management">
   <Request>
      <Output>
         <ResultItem>
            <Identifier>urn:DC-Based-Schema:1</Identifier>
            <Location>urn:DC-Based-Schema:registry:1</Location>
            <ProviderInformation>
              <Name>Schema Creator</Name>
              <Description>Image schema Developer based on MPEG
              </Description>
              <Contact>
                 <EMail>creator-email@schema.com</EMail>
                 <PostalAddress> 2 1st St., Los Angeles, CA 876554, USA
                 </PostalAddress>
                 <Phone>212-1234-4567</Phone>
                 <lang>eng</lang>
              </Contact>
              <ProviderCapability xsi:type="QueryCapabilityType">
                <ServiceCapability>
                  <jpqf:SupportedQueryTypes href="urn:X:Y:Z"/>
                </ServiceCapability>
              </ProviderCapability>
              <ProviderCapability xsi:type="BenchmarkCapabilityType"
                                  name="Example Benchmark"
                                  reference="urn:unique_identifier">
                  <QualityRating queryType="urn:X:Y:Z" rating="0.3"/>
              </ProviderCapability>
              <ProviderCapability xsi:type="ExtensionCapabilityType">
                 <NetworkCapability>
                    <Latency>1223</Latency>
                    <Bandwidth>1234</Bandwidth>
                 </NetworkCapability>
              </ProviderCapability>
            </ProviderInformation>
         </ResultItem>
      </Output>
   </Request>
</SchemaManagement>
```

6.12 SchemaType

6.12.1 General

The `SchemaType` type provides information in order to identify a certain schema.

6.12.2 Syntax

```
<complexType name="SchemaType">
    <sequence>
        <element name="SchemaInformation"
              type="jps_mngt:SchemaInformationType"/>
        <element name="TranslationRules"
              type="jps_mngt:TranslationRulesType"/>
    </sequence>
</complexType>
```

6.12.3 Semantics

Semantics of the `SchemaType` type:

Name	*Definition*
`SchemaType`	Specifies a `SchemaType` type which is used for describing all information in order to register a schema and its translation rules.
`SchemaInformation`	Specifies the description of a certain schema.
`TranslationRules`	Specifies the set of translation rules which are necessary for reformulating a query from the reference metadata model to the registered target metadata model.

6.12.4 Example

In the following example, the usage of `SchemaType` is demonstrated within the element Schema

```
<SchemaManagement jpsearchID="JPS-MSG-01"
                xmlns="JPSearch:schema:management">
   <Register>
      <Input>
         <ProviderInformation>
           <Name>Schema Creator</Name>
           <Description>Image schema Developer based on MPEG</Description>
           <Contact>
              <EMail>creator-email@schema.com</EMail>
              <PostalAddress> 2 1st St., Los Angeles, CA 876554, USA
              </PostalAddress>
              <Phone>212-1234-4567</Phone>
              <lang>eng</lang>
           </Contact>
              <ProviderCapability xsi:type="QueryCapabilityType">
                <ServiceCapability>
                  <jpqf:SupportedQueryTypes href="urn:X:Y:Z"/>
                </ServiceCapability>
              </ProviderCapability>
              <ProviderCapability xsi:type="BenchmarkCapabilityType"
                                  name="Example Benchmark"
                                  reference="urn:unique_identifier">
                 <QualityRating queryType="urn:X:Y:Z" rating="0.3"/>
              </ProviderCapability>
        </ProviderInformation>
        <Schema>
           <SchemaInformation>
              <Description>new version 2009</Description>
              <SpecificationName>DC-Based-Schema</SpecificationName>
              <Version>1.0</Version>
              <Identifier>urn:DC-Based-Schema:1</Identifier>
              <Location>urn:DC-Based-Schema:registry:1</Location>
           </SchemaInformation>
           <TranslationRules>
              <TranslationRule>
                 <!-- example of one-to-one translation -->
                 <FromField>DCB-Field 1</FromField>
                 <ToField>Core-Field K</ToField>
              </TranslationRule>
              <fromFormat>urn:DC-Based-Schema:1</fromFormat>
              <toFormat>urn:JPSearchCore:1</toFormat>
           </TranslationRules>
        </Schema>
      </Input>
   </Register>
</SchemaManagement>
```

6.13 ReplaceInputType

6.13.1 General

The ReplaceInputType type describes the data for an input request during a replacement operation. The requester has the opportunity to replace the stored provider information as well as the set of translation rules of a certain schema which is identified by its schema ID.

6.13.2 Syntax

```
<complexType name="ReplaceInputType">
  <sequence>
    <element name="SchemaID" type="anyURI"/>
    <choice>
      <element name="ProviderInformation"
        type="jps_mngt:ProviderInformationType"/>
      <element name="TranslationRules"
        type="jps_mngt:TranslationRulesType"/>
    </choice>
  </sequence>
</complexType>
```

6.13.3 Semantics

Semantics of the `ReplaceInputType` type:

Name	*Definition*
`ReplaceInputType`	Specifies the `ReplaceInputType` type which describes the data considered for replacement.
`SchemaID`	Specifies the schema ID (URI) which is the target for the replacement process.
`ProviderInformation`	Specifies the provider information of a schema which should be replaced by a new version.
`TranslationRules`	Specifies the translation rules for a certain schema which should be replaced by a new version.

6.13.4 Example

The following example shows an input scenario for replace operation that changes a translation rule.

```
<SchemaManagement jpsearchID="JPS-MSG-05"
            xmlns="JPSearch:schema:management">
   <Replace>
      <Input>
         <SchemaID>urn:DC-Based-Schema:1</SchemaID>
         <TranslationRules>
              <TranslationRule>
                  <!-- example of one-to-one translation -->
                  <FromField>DCB-Field 1</FromField>
                  <ToField>Core-Field K+3</ToField>
              </TranslationRule>
              <fromFormat>urn:DC-Based-Schema:1</fromFormat>
              <toFormat>urn:JPSearchCore:1</toFormat>
           </TranslationRules>
      </Input>
   </Replace>
</SchemaManagement>
```

6.14 ReplaceOutputType

6.14.1 General

The ReplaceOutputType type describes the data which is transmitted in case of a response to a replacement input request.

6.14.2 Syntax

```
<complexType name="ReplaceOutputType">
  <sequence>
   <element name="Identifier" type="anyURI" minOccurs="0"/>
   <choice>
     <element name="Status" type="jps_mngt:InformationType"
        maxOccurs="unbounded"/>
     <element name="Warning" type="jps_mngt:InformationType"
        maxOccurs="unbounded"/>
     <element name="Exception" type="jps_mngt:InformationType"
        maxOccurs="unbounded"/>
   </choice>
  </sequence>
</complexType>
```

6.14.3 Semantics

Semantics of the `ReplaceOutputType` type:

Name	*Definition*
`ReplaceOutputType`	Specifies the `ReplaceOutputType` type which describes a set of messages related to the responder.
`Identifier`	Specifies the schema ID (URI) which is the target for the replacement process.
`Status`	Describes the status of the responder.
`Warning`	Describes the warning messages from the responder.
`Exception`	Describes the exceptions the responder encountered during the process.

6.14.4 Example

The following message shows an example for `ReplaceOutputType` in the element Output.

```
<SchemaManagement jpsearchID="JPS-MSG-06"
                  xmlns="JPSearch:schema:management">
   <Replace>
      <Output>
         <Identifier>urn:DC-Based-Schema:1</Identifier>
         <Status>
            <Code>status-code-y</Code>
            <Description>new trans. Rule registered and being evaluated
            </Description>
         </Status>
      </output>
   </Replace>
</SchemaManagement>
```

6.15 SchemaInformationType

6.15.1 General

The `SchemaInformationType` type provides information to identify the schema and its different versions.

6.15.2 Syntax

```
<complexType name="SchemaInformationType">
  <sequence>
    <element name="Description" type="string" minOccurs="0" />
    <element name="SpecificationName" type="string" minOccurs="0" />
    <element name="Version" type="string" minOccurs="0" />
    <element name="Identifier" type="anyURI" minOccurs="0" />
    <element name="Location" type="anyURI" minOccurs="0" />
  </sequence>
</complexType>
```

6.15.3 Semantics

Semantics of the `SchemaInformationType` type:

Name	*Definition*
`SchemaInformationType`	Specifies a `SchemaInformationType` type which is used for identifying and describing a certain schema.
`Description`	Specifies the description of the schema.
`Version`	Specifies the version of the schema.
`Identifier`	Specifies a unique identifier of the schema.
`Location`	Specifies the location where the schema can be retrieved.
`SpecificationName`	Specifies the name of a standardization body and number/identifier

6.15.4 Example

The following message shows an example for `SchemaInformationType` in the element SchemaInformation.

```
<SchemaManagement jpsearchID="JPS-MSG-01"
                xmlns="JPSearch:schema:management">
  <Register>
    <Input>
      <ProviderInformation>
        <Name>Schema Creator</Name>
        <Description>Image schema Developer based on MPEG</Description>
        <Contact>
          <EMail>creator-email@schema.com</EMail>
          <PostalAddress> 2 1st St., Los Angeles, CA 876554, USA
          </PostalAddress>
          <Phone>212-1234-4567</Phone>
          <lang>eng</lang>
        </Contact>
          <ProviderCapability xsi:type="QueryCapabilityType">
            <ServiceCapability>
              <jpqf:SupportedQueryTypes href="urn:X:Y:Z"/>
            </ServiceCapability>
          </ProviderCapability>
          <ProviderCapability xsi:type="BenchmarkCapabilityType"
                          name="Example Benchmark"
                          reference="urn:unique_identifier">
            <QualityRating queryType="urn:X:Y:Z" rating="0.3"/>
          </ProviderCapability>
      </ProviderInformation>
      <Schema>
        <SchemaInformation>
          <Description>new version 2009</Description>
          <SpecificationName>DC-Based-Schema</SpecificationName>
          <Version>1.0</Version>
          <Identifier>urn:DC-Based-Schema:1</Identifier>
          <Location>urn:DC-Based-Schema:registry:1</Location>
        </SchemaInformation>
        <TranslationRules>
          <TranslationRule>
            <!-- example of one-to-one translation -->
            <FromField>DCB-Field 1</FromField>
            <ToField>Core-Field K</ToField>
          </TranslationRule>
          <fromFormat>urn:DC-Based-Schema:1</fromFormat>
          <toFormat>urn:JPSearchCore:1</toFormat>
        </TranslationRules>
      </Schema>
    </Input>
  </Register>
</SchemaManagement>
```

6.16 RegisterOutputType

6.16.1 General

The `RegisterOutputType` type provides information in order to respond status and/or error information of a register process.

6.16.2 Syntax

```
<complexType name="RegisterOutputType">
  <sequence>
     <element name="Identifier" type="anyURI" minOccurs="0" />
    <choice>
        <element name="Status"
           type="jps_mngt:InformationType" maxOccurs="unbounded"/>
        <element name="Warning"
           type="jps_mngt:InformationType" maxOccurs="unbounded"/>
        <element name="Exception"
           type="jps_mngt:InformationType" maxOccurs="unbounded"/>
    </choice>
  </sequence>
</complexType>

<complexType name="InformationType">
    <sequence>
        <element name="Code" type="positiveInteger"/>
        <element name="Description" type="string"/>
    </sequence>
</complexType>
```

6.16.3 Semantics

Semantics of the `RegisterOutputType` type:

Name	*Definition*
`RegisterOutputType`	Specifies the `RegisterOutputType` type. Describes a set of messages related to the responder.
Status	Describes the status of the responder.
Warning	Describes the warning from the responder.
Exception	Describes the exception the responder encountered during the process.
Identifier	Specifies the identifier (schema or translation rules) where the response belongs to.

Semantics of the `InformationType`:

Name	*Definition*
`Code`	Describes the number assigned for the information.
`Description`	Describes the details of the information.

6.16.4 Example

The following message shows an example for `RegisterOutputType` in the element Output.

```
<SchemaManagement jpsearchID="JPS-MSG-02"
                  xmlns="JPSearch:schema:management">
   <Register>
      <Output>
         <Identifier>urn:DC-Based-Schema:1</Identifier>
         <Status>
            <Code>status-code-x</Code>
            <Description>registered and being evaluated</Description>
         </Status>
      </Output>
   </Register>
</SchemaManagement>
```

7 JPSearch Translation Rules Declaration Language (JPTRDL)

7.1 Wrapper of the schema

The description examples and syntax of description tools specified in this subclause assume that a schema wrapper is provided which identifies the XML Schema namespace (XML Schema) and JPTRDL namespace:

```
<schema xmlns="http://www.w3.org/2001/XMLSchema"
          xmlns:jptrdl="JPSearch:schema:translation"
          targetNamespace="JPSearch:schema:translation"
          elementFormDefault="qualified" attributeFormDefault="unqualified">
```

The following tag is used to close the schema:

```
</schema>
```

7.2 TranslationRulesType

7.2.1 General

The `TranslationRulesType` type provides information in order to define a set of translation rules that provide the translation strategies between the core schema and a registered target metadata schema, which shall be given in the format of XML schema compliant to XML, XML Schema Part 1 and XML Schema Part 2.

7.2.2 Syntax

```
<element name="TranslationRules" type="jptrdl:TranslationRulesType" />
<complexType name="TranslationRulesType">
  <sequence>
 <element name="TranslationRule"
            type="jptrdl:TranslationRuleType" maxOccurs="unbounded"/>
  </sequence>
  <attribute name="fromFormat" type="anyURI"/>
  <attribute name="toFormat" type="anyURI"/>
</complexType>
```

7.2.3 Semantics

Semantics of the `TranformationRulesType` type:

Name	*Definition*
`TranslationRulesType`	Specifies the `TranslationRulesType` type that allows the definition of a set of translation rules providing means for transforming JPQF queries from the core schema to a registered target schema.
`TranslationRule`	Describes one translation rule.
`fromFormat`	Identifies (by the registered URI) the source schema.
`toFormat`	Identifies (by the registered URI) the target schema.

7.2.4 Example

The following example shows a simple translation rule which maps the Creators/FamilyName metadata path from the core schema into the //CreationInformation/Creation/Creator/Agent/Name/FamilyName metadata path in MPEG-7.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<TranslationRules xmlns="JPSearch:schema:translation" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation=" JPSearch:schema:translation jptrdl.xsd"
fromFormat="JPSearch:schema:coremetadata" toFormat="urn:mpeg:mpeg7:schema:2004">
  <TranslationRule xsi:type="OneToOneFieldTranslationType">
    <FromField xsi:type="SourceFieldType">
      <XPathExpression>Creators/FamilyName</XPathExpression>
    </FromField>
    <ToField xsi:type="TargetFieldType">

<XPathExpression>//CreationInformation/Creation/Creator/Agent/Name/FamilyName</
XPathExpression>
    </ToField>
  </TranslationRule>
</TranslationRules>
```

7.3 Abstract Types

7.3.1 General

The following abstract types support the composition of a modular architecture in order to combine individual source and target types during the configuration of certain translation rules.

7.3.2 Syntax

```
<complexType name="TranslationRuleType" abstract="true"/>
<complexType name="SourceType"      abstract="true"/>
<complexType name="TargetType"      abstract="true"/>
```

7.3.3 Semantics

Semantics of the abstract types:

Name	*Definition*
`TranslationRuleType`	Specifies the abstract `TranslationRuleType` type, which serves as root type for all translation rules.
`SourceType`	Specifies the abstract `SourceType` type which, serves as root type for all source definitions for a translation rule.
`TargetType`	Specifies the abstract `TargetType` type which, serves as root type for all target definitions for a translation rule.

7.4 OneToOneFieldTranslationType

7.4.1 General

This type realizes a one-to-one pattern of a translation rule.

7.4.2 Syntax

```
<complexType name="OneToOneFieldTranslationType">
  <complexContent>
 <extension base="jptrdl:TranslationRuleType">
   <sequence>
    <element name="FromField" type="jptrdl:SourceType"/>
    <element name="ToField" type="jptrdl:TargetType"/>
   </sequence>
    </extension>
  </complexContent>
</complexType>
```

7.4.3 Semantics

Semantics of the `OneToOneField`Translation`Type` type:

Name	*Definition*
`OneToOneFieldTranslationType`	Specifies the `OneToOneFieldTranslationType` type which realizes the one-to-one pattern for translation rules. In this case one field element of the source schema will be transformed into one element of the target schema.
`FromField`	This element references to one of the subtypes of the `SourceType` type and points to a specific element of the source schema.
`ToField`	This element references to one of the subtypes of the `TargetType` type and points to a specific element of the target schema.

7.4.4 Example

The example in 7.2.4 shows a one-to-one translation rule.

7.5 ManyToOneFieldTranslationType

7.5.1 General

This type realizes the many-to-one pattern of a translation rule.

7.5.2 Syntax

```
<complexType name="ManyToOneFieldTranslationType">
  <complexContent>
 <extension base="jptrdl:TranslationRuleType">
   <sequence>
    <element name="FromField" type="jptrdl:SourceType"
                  minOccurs="2" maxOccurs="unbounded"/>
    <element name="ToField" type="jptrdl:TargetType"/>
   </sequence>
 </extension>
  </complexContent>
</complexType>
```

7.5.3 Semantics

Semantics of the `ManyToOneFieldTranslationType` type:

Name	*Definition*
`ManyToOneFieldTranslationType`	Specifies the `ManyToOneFieldTranslationType` type which realizes the many-to-one pattern for translation rules. In this case at least 2 field elements of the source schema will be transformed into one element of the schema.
`FromField`	This element references to one of the subtypes of the `SourceType` type and points to a specific element of the source schema. At least two elements shall occur.
`ToField`	This element references to one of the subtypes of the `TargetType` type and points to a specific element of the target schema. Only one element is allowed.

7.5.4 Example

The following example shows a many-to-one translation rule which maps the jpsearch's creators/givenName and jpsearch's creators/familyName elements into a compact Dublin Core creator field.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<TranslationRules xmlns="JPSearch:schema:translation" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:translation jptrdl.xsd" fromForma
t="JPSearch:schema:coremetadata" toFormat="http://dublincore.org/documents/dcmi-terms">
  <TranslationRule xsi:type="ManyToOneFieldTranslationType">
    <FromField xsi:type="SourceFieldType">
      <XPathExpression>Creators/GivenName</XPathExpression>
    </FromField>
    <FromField xsi:type="SourceFieldType">
      <XPathExpression>Creators/FamilyName</XPathExpression>
    </FromField>
    <ToField xsi:type="FormattedTargetFieldType">
      <XPathExpression>creator</XPathExpression>
      <ReplaceWithRegExpr>$1 $2</ReplaceWithRegExpr>
    </ToField>
  </TranslationRule>
</TranslationRules>
```

7.6 OneToManyFieldTranslationType

7.6.1 General

This type realizes the one-to-many pattern of a translation rule.

7.6.2 Syntax

```
<complexType name="OneToManyFieldTranslationType">
  <complexContent>
 <extension base="jptrdl:TranslationRuleType">
   <sequence>
    <element name="FromField" type="jptrdl:SourceType"/>
    <element name="ToField" type="jptrdl:TargetType"
                      minOccurs="2" maxOccurs="unbounded"/>
   </sequence>
 </extension>
  </complexContent>
</complexType>
```

7.6.3 Semantics

Semantics of the `OneToManyFieldTranslationType` type:

Name	*Definition*
`OneToManyFieldMappingType`	Specifies the `OneToManyFieldTranslationType` type which realizes the one-to-many pattern for transfromation rules. In this case one element of the source schema will be transformed into at least 2 elements of the target schema.
`FromField`	This element references to one of the subtypes of the `SourceType` type and points to a specific element of the source schema. Only one element is allowed.
`ToField`	This element references to one of the subtypes of the `TargetType` type and points to a specific element of the target schema. At least two elements shall occur.

7.6.4 Example

The following example shows a one-to-many translation rule which maps the jpsearch's date element into three fields of an example target format.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<TranslationRules xmlns="JPSearch:schema:translation" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="JPSearch:schema:translation jptrdl.xsd" fromForma
t="JPSearch:schema:coremetadata" toFormat="urn:exampleformat">
  <!-- example : Date="05/10/2008" to Day="05" AND Month="10" AND Year="2008" -->
  <TranslationRule xsi:type="OneToManyFieldTranslationType">
    <FromField xsi:type="FilteredSourceFieldType">
      <XPathExpression>date</XPathExpression>
      <FilterWithRegExpr>(\d\d)/(\d\d)/(\d\d\d\d)</FilterWithRegExpr>
    </FromField>
    <ToField xsi:type="FormattedTargetFieldType">
      <XPathExpression>day</XPathExpression>
      <ReplaceWithRegExpr>$1</ReplaceWithRegExpr>
    </ToField>
    <ToField xsi:type="FormattedTargetFieldType">
        <XPathExpression>month</XPathExpression>
        <ReplaceWithRegExpr>$2</ReplaceWithRegExpr>
    </ToField>
    <ToField xsi:type="FormattedTargetFieldType">
        <XPathExpression>year</XPathExpression>
        <ReplaceWithRegExpr>$3</ReplaceWithRegExpr>
    </ToField>
  </TranslationRule>
</TranslationRules>
```

7.7 SourceFieldType

7.7.1 General

This type defines all information for pointing to an element of the source schema.

7.7.2 Syntax

```
<complexType name="SourceFieldType">
  <complexContent>
 <extension base="jptrdl:SourceType">
   <sequence>
    <element name="XPathExpression" type="jptrdl:xPathType"/>
   </sequence>
 </extension>
  </complexContent>
</complexType>

<simpleType name="xPathType">
  <restriction base="token"/>
</simpleType>
```

7.7.3 Semantics

Semantics of the `SourceFieldType` type:

Name	*Definition*
`SourceFieldType`	Specifies the `SourceFieldType` type which allows the selection of a certain element of the source metadata format. The selection of a certain element shall be done as specified in the XPath specification using the `XPathExpression` element. The type is an extension of the abstract `SourceType` type.
`XpathExpression`	This element allows specifying a single element in the source metadata format. The specification of a single element shall be done as defined in the XPath specification.
`XPathType`	This element specifies the xPathType to hold XPath expression as specified in XPath.

7.7.4 Example

The example in 7.2.4 shows the usage of the `SourceFieldType`.

7.8 FilteredSourceFieldType

7.8.1 General

This type defines all information for pointing to an element of the source metadata format. In addition, some filter conditions can be applied which are performed during the translation process.

7.8.2 Syntax

```
<complexType name="FilteredSourceFieldType">
  <complexContent>
    <extension base="jptrdl:SourceFieldType">
      <sequence>
    <element name="FilterWithRegExpr" type="string" minOccurs="0"/>
    <element name="VariableBinding" maxOccurs="unbounded">
      <complexType>
       <choice>
         <element name="ExplicitBinding" type="string"/>
         <sequence>
          <element name="ExplicitPrefixBinding" type="string"
                   minOccurs="0" maxOccurs="unbounded"/>
          <element name="ListBinding" type="string" minOccurs="0"/>
          <element name="ExplicitPostfixBinding" type="string"
                     minOccurs="0" maxOccurs="unbounded"/>
         </sequence>
       </choice>
      </complexType>
    </element>
   </sequence>
    </extension>
  </complexContent>
</complexType>
```

7.8.3 Semantics

Semantics of the `FilteredSourceFieldType` type:

Name	*Definition*
`FilteredSourceFieldType`	Specifies the `FilteredSourceFieldType` type which allows the selection of a certain element of the source metadata format. The type is an extension of the `SourceFieldType` type.
`FilterWithRegExpr`	This element defines a filter condition by using regular expressions. By evaluating this regular expressions the content of the addressed element in the source schema is divided into tokens, which can be bound to variables.
`VariableBinding`	This element is the main section to bind the extracted tokens to particular variables.
`ExplicitBinding`	This element may be used without a regular expression. It simply binds the addressed metadata element / graph pattern to a variable.
`ExplicitPrefixBinding`	This element is used in order to bind the first token defined by a regular expression to a variable.
`ListBinding`	This element binds all tokens defined by a regular expression to a list of variables. Note that only those tokens are considered that are not target of any other binding.
`ExcplicitPostfixBinding`	This element is used in order to bind the last token defined by a regular expression to a variable.

7.8.4 Example

The example in 7.6.4 shows the usage of the `FilteredSourceFieldType`.

7.9 TargetFieldType

7.9.1 General

This type defines all information for pointing to an element of the target metadata format.

7.9.2 Syntax

```
<complexType name="TargetFieldType">
  <complexContent>
 <extension base="jptrdl:TargetType">
   <sequence>
    <element name="XPathExpression" type="jptrdl:xPathType"/>
   </sequence>
 </extension>
  </complexContent>
</complexType>
```

7.9.3 Semantics

Semantics of the `TargetFieldType` type:

Name	*Definition*
`TargetFieldType`	Specifies the `TargetFieldType` type which allows the selection of a certain element of the target metadata format. The selection of a certain element shall be done as specified in the XPath specification using the XPathExpression element. The type is an extension of the abstract `TargetType` type.
`XPathExpression`	This element allows specifying a single element the target metadata format. The specification of a single element shall be done as defined in the XPath specification.

7.9.4 Example

The example in 7.2.4 shows the usage of the `TargetFieldType`.

7.10 FormattedTargetFieldType

7.10.1 General

This type defines all information for pointing to an element of the target metadata format. In addition, some filter conditions can be applied which are performed during the translation process in order to format the content correctly.

7.10.2 Syntax

```
<complexType name="FormattedTargetFieldType">
  <complexContent>
 <extension base="jptrdl:TargetFieldType">
   <sequence>
    <element name="ReplaceWithRegExpr" type="string"
                          minOccurs="0"/>
   </sequence>
  </extension>
  </complexContent>
</complexType>
```

7.10.3 Semantics

Semantics of the `FormattedTargetFieldType` type:

Name	*Definition*
`FormattedTargetFieldType`	Specifies the `FormattedTargetFieldType` type which allows the selection of a certain element of the target metadata format. The type is an extension of the `TargetFieldType` type.
`ReplaceWithRegExpr`	This element defines a format condition by using regular expressions. By evaluating this regular expression the content of the source schema is modified and transformed respectively in order to fit to the addressed element in the target metadata format.

7.10.4 Example

The example in 7.6.4 shows the usage of the `FormattedTargetFieldType`.

8 JPEG Ontology for Image Description (JPOnto)

8.1 General

This clause specifies the set of classes, properties, and restrictions that constitute the JPSearch Metadata Ontology (JPOnto). This ontology specification provides the foundation to implement applications in different domains that can represent, exchange, and integrate digital image information generated in different systems and under different contexts.

JPOnto users may only need to use parts of the entire ontology, depending on their needs and according to how much detail they want to include in their image information. For this, the JPOnto terms (classes and properties) are grouped into two sub-vocabularies to provide an incremental introduction to the ontology.

— **JPOnto-core**: A core vocabulary that acts as a central component to interconnect all the other sub-vocabularies.

— **JPOnto-visual**: A rich set of constructs for semantic visual content description, including, but not limited to, a uniform description of identities, features, aspect, relationships, actions and emotional information of people appearing in the images, as well as description of events, locations and objects.

8.2 JPOnto-core

8.2.1 Outline

A core vocabulary named JPOnto-core is the basic set of entities and properties which serves as a central component to interconnect all the other sub-vocabularies of the JPEG Ontology for Image Description (JPOnto). The following classes provide a basis for the rest of JPOnto-core.

— **jponto:Image**: Represents a digital image.

— **jponto:RegionOfInterest**: A certain region within a digital image.

— **jponto:Agent**: An agent (e.g. person, group, software or physical artifact).

The primary classes relate to one another and to themselves using the properties shown in Figure 2.

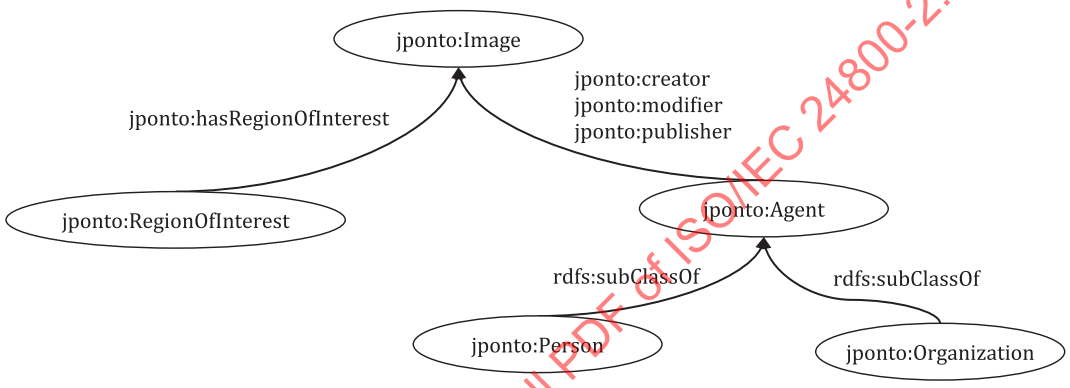

Figure 2 — Outline of JPOnto-core

8.2.2 Example

This example shows a complete description of an imaginary image. It has a unique ID, a title and a creator.

```
@base <http://example.org/> .
@prefix jponto: <http://www.jpeg.org/ns/jponto#> .
<#image1> a jponto:Image;
  jponto:title "An example image";
  jponto:keyword "London";
  jponto:keyword "JPEG";
  jponto:creator [
     jponto:givenName "John";
     jponto:familyName "Smith";
     jponto:mbox <mailto:jsmith@example.com> ];
.
```

8.2.3 Semantics

8.2.3.1 JPOnto-core class definitions (in alphabetical order)

Class	jponto:Agent

Superclasses	—
IRI	http://www.jpeg.org/ns/jponto#Agent
Description	An agent (eg. person, group, software or physical artifact).
Described with properties:	
In range of	jponto:creator, jponto:modifier, jponto:publisher
Has subclasses	jponto:Organisation, jponto:Person
Syntax	:Agent a owl:Class; rdfs:comment "An agent (eg. person, group, software or physical artifact)."; .
Example	*(see examples for subclasses)*

Class	**jponto:Image**
Superclasses	—
IRI	http://www.jpeg.org/ns/jponto#Image
Description	Represents a digital image
Described with properties:	jponto:creationDate, jponto:creator, jponto:depicts, jponto:description, jponto:GPSaltitude, jponto:GPSlatitude, jponto:GPSlongitude, jponto:hasRegionOfInterest, jponto:height, jponto:modifiedDate, jponto:modifier, jponto:publisher, jponto:title, jponto:width
In range of	—
Has subclasses	—
Syntax	:Image a owl:Class; rdfs:comment " Represents a digital image "; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:title "An example image"; .

Class	**jponto:Organisation**
Superclasses	jponto:Agent
IRI	http://www.jpeg.org/ns/jponto#Organisation
Description	Organisation or moral agent.
Described with properties:	jponto:name
In range of	jponto:creator, jponto:modifier, jponto:publisher
Has subclasses	—

Syntax	:Organisation a owl:Class; rdfs:comment "Organisation or moral agent"; rdfs:subClassOf :Agent ; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:hasCreatorPublisher [a jponto:Organisation; jponto:name "Example Company";]; .

Class	**jponto:Person**
Superclasses	
IRI	http://www.jpeg.org/ns/jponto#Person
Description	A physical person.
Described with properties:	jponto:familyName, jponto:givenName, jponto:mbox
In range of	jponto:creator, jponto:modifier, jponto:publisher, jponto:depicts
Has subclasses	-
Syntax	:Person a owl:Class; rdfs:comment "A physical person."; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:hasCreator [a jponto:Person; foaf:givenName "John Smith"; foaf:mbox <mailto:jsmith@example.com>]; .

Class	**jponto:RegionOfInterest**
Superclasses	jponto:Image
IRI	http://www.jpeg.org/ns/jponto#RegionOfInterest
Description	Represents a certain region within a digital image.
Described with properties:	jponto:boundingPolygon, jponto:depicts

In range of	jponto:hasRegionOfInterest
Has subclasses	—
Syntax	:RegionOfInterest a owl:Class; rdfs:comment "Represents a certain region within a digital image."; rdfs:subClassOf :Image; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:hasRegionOfInterest [jponto:boundingPolygon “0 0 100 100”.]; .

8.2.3.2 JPOnto-core property definitions (in alphabetical order)

Property	**jponto:boundingPolygon**
IRI	http://www.jpeg.org/ns/jponto#boundingPolygon
Description	Specifies the coordinates of the edges of the polygon eclosing a region in the form "x1,y1, x2,y2, x3,y3, ...".
Has super-properties:	—
Has domain	jponto:RegionOfInterest
Has range	xsd:string
Syntax	:boundingPolygon a owl:DatatypeProperty; rdfs:comment "Specifies the coordinates of the edges of the polygon eclosing a region in the form "x1,y1, x2,y2, x3,y3, ..."; rdfs:domain :Image ; rdfs:range xsd:string ; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . @prefix xsd: <http://www.w3.org/2001/XMLSchema#> . <#image1> a jponto:Image; jponto:creationDate "2001-10-26T21:32:52"^^xsd:date; jponto:hasRegionOfInterest [jponto:boundingPolygon "0 0 100 100"; .

Property	jponto:creationDate
IRI	http://www.jpeg.org/ns/jponto#creationDate
Description	Describes the date when the image is created.
Has super-properties:	—
Has domain	jponto:Image
Has range	xsd:dateTime
Syntax	:creationDate a owl:DatatypeProperty; rdfs:comment "Describes the date when the image is created."; rdfs:domain :Image ; rdfs:range xsd:dateTime; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . @prefix xsd: <http://www.w3.org/2001/XMLSchema#> . <#image1> a jponto:Image; jponto:creationDate "2001-10-26T21:32:52"^^xsd:date; .

Property	jponto:creator
IRI	http://www.jpeg.org/ns/jponto#creator
Description	Describes the creator of the image.
Has super-properties:	—
Has domain	jponto:Image
Has range	jponto:Agent
Syntax	:creator a owl:ObjectProperty; rdfs:comment "Describes the creator of the image."; rdfs:domain :Image; rdfs:range :Agent; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:creator [jponto:givenName "John Smith"; jponto:mbox <mailto:jsmith@example.com>]; .

Property	jponto:description
IRI	http://www.jpeg.org/ns/jponto#description
Description	the content of the image in the form of text
Has super-properties:	—
Has domain	jponto:Image
Has range	xsd:string
Syntax	:description a owl:DatatypeProperty; rdfs:comment "Specifies the content of the image in the form of text."; rdfs:domain :Image ; rdfs:range xsd:string ; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:description "An example image taken during the MPEG meeting"; .

Property	jponto:familyName
IRI	http://www.jpeg.org/ns/jponto#familyName
Description	A name shared in common to identify the members of a family, as distinguished from each member's given name.
Has super-properties:	—
Has domain	jponto:Person
Has range	xsd:string
Syntax	:familyName a owl:DatatypeProperty; rdfs:comment " Specifies the family name of a person."; rdfs:domain :Person; rdfs:range xsd:string; .

Example	@base <http://example.org/> .
	@prefix jponto: <http://www.jpeg.org/ns/jponto#> .
	<#image1> a jponto:Image;
	  jponto:creator [
	  jponto:givenName "John";
	  jponto:familyName "Smith";
	  jponto:mbox <mailto:jsmith@example.com>];
	.

Property	**jponto:givenName**
IRI	http://www.jpeg.org/ns/jponto#givenName
Description	Specifies the given name of a person.
Has super-properties:	—
Has domain	jponto:Person
Has range	xsd:string

Syntax	:givenName a owl:DatatypeProperty;
	  rdfs:comment " Specifies the given name of a person.";
	  rdfs:domain :Person;
	  rdfs:range xsd:string;
	.

Example	@base <http://example.org/> .
	@prefix jponto: <http://www.jpeg.org/ns/jponto#> .
	<#image1> a jponto:Image;
	  jponto:creator [
	  jponto:givenName "John";
	  jponto:familyName "Smith";
	  jponto:mbox <mailto:jsmith@example.com>];
	.

Property	**jponto:GPSAltitude**
IRI	http://www.jpeg.org/ns/jponto#GPSAltitude
Description	Specifies the GPS altitude of the image, in decimal meters.
Has super-properties:	—
Has domain	jponto:Image
Has range	xsd:double

Syntax	:GPSAltitude a owl:DatatypeProperty; rdfs:comment "Describes the GPS altitude of the image "; rdfs:domain :Image; rdfs:range xsd:double; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:GPSaltitude 1000.0; .

Property	**jponto:GPSLatitude**
IRI	http://www.jpeg.org/ns/jponto#GPSLatitude
Description	Specifies the GPS latitude of the image in decimal degrees.
Has super-properties:	—
Has domain	jponto:Image
Has range	xsd:double
Syntax	:GPSLatitude a owl:DatatypeProperty; rdfs:comment "Describes the GPS latitude of the image "; rdfs:domain :Image; rdfs:range xsd:double; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:GPSlatitude 34.0; .

Property	**jponto:GPSLongitude**
IRI	http://www.jpeg.org/ns/jponto#GPSLongitude
Description	Specifies the GPS longitude of the image in decimal degrees.
Has super-properties:	—
Has domain	jponto:Image
Has range	xsd:double

Syntax	:GPSLongitude a owl:DatatypeProperty; rdfs:comment "Describes the GPS longitude of the image "; rdfs:domain :Image; rdfs:range xsd:double; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:GPSlongitude 34.0; .

Property	**jponto:hasRegionOfInterest**
IRI	http://www.jpeg.org/ns/jponto#hasRegionOfInterest
Description	Describes a region of interest of the image.
Has super-properties:	—
Has domain	jponto:Image
Has range	jponto:RegionOfInterest
Syntax	:hasRegionOfInterest a owl:ObjectProperty; rdfs:comment "Describes a region of interest of the image."; rdfs:domain :Image; rdfs:range :RegionOfInterest; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . @prefix xsd: <http://www.w3.org/2001/XMLSchema#> . <#image1> a jponto:Image; jponto:hasRegionOfInterest [jponto:boundingPolygon "0 0 100 100"; .

Property	**jponto:height**
IRI	http://www.jpeg.org/ns/jponto#title
Description	Specifies the number of pixel rows of the image.
Has super-properties:	—
Has domain	jponto:Image
Has range	xsd:integer

Syntax	:height a owl:DatatypeProperty; rdfs:comment "Specifies the number of pixel rows of the image."; rdfs:domain :Image; rdfs:range xsd:integer; .
Example	s@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:height 1024; .

Property	**jponto:keyword**
IRI	http://www.jpeg.org/ns/jponto#keyword
Description	Describes a keyword that characterize the image
Has super-properties:	—
Has domain	jponto:Image
Has range	xsd:string
Syntax	:keyword a owl:DatatypeProperty; rdfs:comment "Describes a keyword that characterize the image "; rdfs:domain :Image ; rdfs:range xsd:string ; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:keyword "Seoul"; jponto:keyword "MPEG"; jponto:keyword "social event"; .

Property	**jponto:mbox**
IRI	http://www.jpeg.org/ns/jponto#mbox
Description	An email address as defined in RFC 5321
Has super-properties:	—
Has domain	jponto:Person
Has range	xsd:string

Syntax	:mbox a owl:DatatypeProperty; rdfs:comment " Specifies the mbox of a person."; rdfs:domain :Person; rdfs:range xsd:string; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:creator [jponto:givenName "John"; jponto:familyName "Smith"; jponto:mbox <mailto:jsmith@example.com>]; .

Property	jponto:modifiedDate
IRI	http://www.jpeg.org/ns/jponto#modifiedDate
Description	Describes the date when the image is modified.
Has super-properties:	—
Has domain	jponto:Image
Has range	xsd:dateTime
Syntax	:modifiedDate a owl:DatatypeProperty; rdfs:comment "Describes the date when the image is modified."; rdfs:domain :Image ; rdfs:range xsd:dateTime ; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . @prefix xsd: <http://www.w3.org/2001/XMLSchema#> . <#image1> a jponto:Image; jponto:modifiedDate "2001-10-26T21:32:52"^^xsd:date; .

Property	jponto:modifier
IRI	http://www.jpeg.org/ns/jponto#modifier
Description	Describes an agent who changed the image.
Has super-properties:	—
Has domain	jponto:Image
Has range	jponto:Agent
Syntax	:creator a owl:ObjectProperty; rdfs:comment "Describes an agent who changed the image."; rdfs:domain :Image; rdfs:range :Agent; .

Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:modifier [jponto:givenName "John Smith"; jponto:mbox <mailto:jsmith@example.com>]; .

Property	**jponto:publisher**
IRI	http://www.jpeg.org/ns/jponto#publisher
Description	Describes information about the publishing people or organization of the image.
Has super-properties:	—
Has domain	jponto:Image
Has range	jponto:Agent
Syntax	:publisher a owl:ObjectProperty; rdfs:comment " Describes the publisher of the image."; rdfs:domain :Image; rdfs:range :Agent; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:publisher [jponto:givenName "John Smith"; jponto:mbox <mailto:jsmith@example.com>]; .

Property	**jponto:title**
IRI	http://www.jpeg.org/ns/jponto#title
Description	Specifies the title or name given to the image.
Has super-properties:	—
Has domain	jponto:Image
Has range	xsd:string

Syntax	:title a owl:DatatypeProperty; rdfs:comment "Describes the title of the image "; rdfs:domain :Image; rdfs:range xsd:string; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:title "An example image"; .

Property	**jponto:width**
IRI	http://www.jpeg.org/ns/jponto#title
Description	Specifies the number of pixel columns of the image.
Has super-properties:	—
Has domain	jponto:Image
Has range	xsd:integer
Syntax	:width a owl:DatatypeProperty; rdfs:comment "Specifies the number of pixel columns of the image ."; rdfs:domain :Image; rdfs:range xsd:integer; .
Example	@base <http://example.org/> . @prefix jponto: <http://www.jpeg.org/ns/jponto#> . <#image1> a jponto:Image; jponto:width 1024; .

8.2.4 Turtle representation of JPOnto-core

The following is the Turtle (Terse RDF Triple Language) representation of the JPOnto-core sub-vocabulary.

```
#Target namespace
@prefix : <http://www.jpeg.org/ns/jponto#> .

#Basic constructors by W3C (RDF, RDFS, OWL)
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix jponto: <http://www.jpeg.org/ns/jponto#> .

#Datatype properties
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:Agent a owl:Class;
   rdfs:comment "An agent (eg. person, group, software or physical artifact).";
.
:Image a owl:Class;
   rdfs:comment " Represents a digital image ";
.

:Organisation a owl:Class;
   rdfs:comment "Organisation or moral agent";
   rdfs:subClassOf :Agent ;
.

:Person a owl:Class;
   rdfs:comment "A physical person.";

.
:RegionOfInterest a owl:Class;
   rdfs:comment "Represents a certain region within a digital image.";
   rdfs:subClassOf :Image;
.

:boundingPolygon a owl:DatatypeProperty;
   rdfs:comment "Specifies the coordinates of the edges of the polygon eclosing a region in the form "x1,y1, x2,y2, x3,y3, ...";
   rdfs:domain :Image ;
   rdfs:range xsd:string ;
.
```

```
:creationDate a owl:DatatypeProperty;
   rdfs:comment "Describes the date when the image is created.";
   rdfs:domain :Image ;
  rdfs:range xsd:dateTime;
 .
:creator a owl:ObjectProperty;
   rdfs:comment " Describes the author of the image.";
   rdfs:domain :Image;
   rdfs:range  :Agent;
 .

:description a owl:DatatypeProperty;
   rdfs:comment "Specifies the content of the image in the form of text.";
   rdfs:domain :Image ;
   rdfs:range xsd:string ;
.

:familyName a owl:DatatypeProperty;
   rdfs:comment " Specifies the family name of a person.";
   rdfs:domain :Person;
   rdfs:range xsd:string;
.
:givenName a owl:DatatypeProperty;
   rdfs:comment " Specifies the given name of a person.";
   rdfs:domain :Person;
   rdfs:range xsd:string;
.
:GPSAltitude a owl:DatatypeProperty;
   rdfs:comment "Describes the GPS altitude of the image ";
   rdfs:domain  :Image;
   rdfs:range xsd:double;
.
:GPSLatitude a owl:DatatypeProperty;
   rdfs:comment "Describes the GPS latitude of the image ";
   rdfs:domain :Image;
   rdfs:range xsd:double;
.
```

```
:GPSLongitude a owl:DatatypeProperty;
   rdfs:comment "Describes the GPS longitude of the image ";
   rdfs:domain  :Image;
   rdfs:range xsd:double;
.

:hasRegionOfInterest a owl:ObjectProperty;
   rdfs:comment "Describes a region of interest of the image.";
   rdfs:domain :Image;
   rdfs:range  :RegionOfInterest;
 .

:height a owl:DatatypeProperty;
   rdfs:comment "Specifies the height of the image.";
   rdfs:domain  :Image;
   rdfs:range xsd:integer;
.
:keyword a owl:DatatypeProperty;
   rdfs:comment "Describes a keyword that characterize the image ";
   rdfs:domain :Image ;
   rdfs:range xsd:string ;
.
:mbox a owl:DatatypeProperty;
   rdfs:comment " Specifies the mbox of a person.";
   rdfs:domain  :Person;
   rdfs:range xsd:string;
.
:modifiedDate a owl:DatatypeProperty;
   rdfs:comment "Describes the date when the image is modified.";
   rdfs:domain :Image ;
   rdfs:range xsd:dateTime ;
.
:creator a owl:ObjectProperty;
   rdfs:comment "Describes an agent who changed the image.";
   rdfs:domain  :Image;
   rdfs:range  :Agent;
 .
:publisher a owl:ObjectProperty;
   rdfs:comment " Describes the publisher of the image.";
   rdfs:domain  :Image;
   rdfs:range  :Agent;
 .
```

```
:publisher a owl:ObjectProperty;
   rdfs:comment " Describes the publisher of the image.";
   rdfs:domain  :Image;
   rdfs:range  :Agent;
 .
:title a owl:DatatypeProperty;
   rdfs:comment "Describes the title of the image ";
   rdfs:domain  :Image;
   rdfs:range xsd:string;
.
:width a owl:DatatypeProperty;
   rdfs:comment "Describes the title of the image ";
   rdfs:domain  :Image;
   rdfs:range xsd:integer;
.
```

8.3 JPOnto-visual

8.3.1 Outline

The JPOnto-Visual sub-vocabulary provides a core set of classes and properties facilitating the semantic description of the contents of the image. The ***jponto:depicts*** property is the core component of JPOnto-visual. The *jponto:depicts* property allows binding an image with the real-world objects and events that it represents. Defining the *RegionOfInterest* as a subclass of *Image*, allows to also use the *depicts* predicate to bind ROIs with real-world objects and events. These relationships are shown in Figure 3.

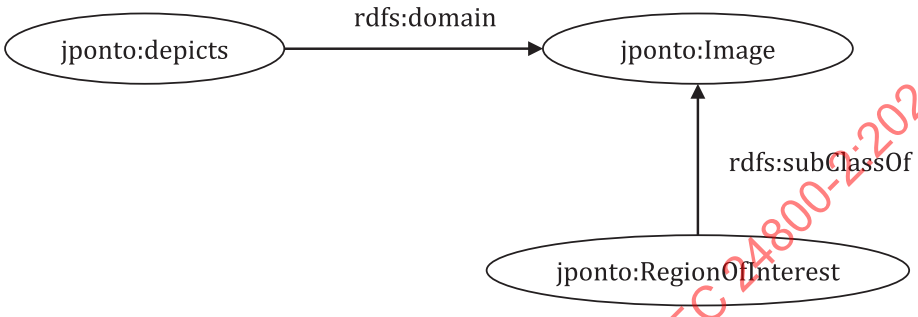

Figure 3 — The *jponto:depicts* property allows binding an image with the real-world objects and events that it represents

8.3.2 Examples

8.3.2.1 General

This Clause provides three examples of semantic description of images.

8.3.2.2 Example 1

This example shows the semantic description of the contents of an imaginary image.

```
@base <http://example.org/> .
@prefix jponto: <http://www.jpeg.org/ns/jponto#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<#image1> a jponto:Image;
  jponto:depicts <#person1>;
  jponto:depicts <#building1>;
.
<#person1> a jponto:Person;
  jponto:givenName "John smith";
  jponto:mbox <mailto:jsmith@example.com>
  jponto:hasAge jponto:Youngness;
  jponto:hasFeeling jponto:Happiness;
  jponto:hasGender jponto:Male;
.
<#building1> a jponto:Building;
  jponto:givenName "Namdaemun gate";
  jponto:wikipedia https://en.wikipedia.org/wiki/Sungnyemun;
.
```

8.3.2.3 Example 2

This example shows the semantic description of the content of an image showing a boy and his grandfather.

```
@base <http://example.org/> .
@prefix jponto: <http://www.jpeg.org/ns/jponto#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<#image1> a jponto:Image;
  jponto:depicts <#person1>;
  jponto:depicts <#person2>;
.
<#person1> a jponto:Person;
  jponto:hasAge jponto:Youngness;
  jponto:hasFeeling jponto:Happiness;
  jponto:hasGender jponto:Male;
.
<#person2> a jponto:Person;
  jponto:hasAge jponto:Oldness;
  jponto:hasGender jponto:Male;
.
<#relationship1> a jponto:Parentage;
  jponto:involves <#person1>;
  jponto:involves <#person1>;
.
```

8.3.2.4 Example 3

This example shows the semantic description of the content of an image showing a man carrying a hat in his hand and talking with a woman.

```
@base <http://example.org/> .
@prefix jponto: <http://www.jpeg.org/ns/jponto#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<#image1> a jponto:Image;
 jponto:depicts <#person1>;
 jponto:depicts <#person2>;
 jponto:depicts <#action1>;
 jponto:depicts <#action2>;
.
<#person1> a jponto:Person;
 jponto:hasGender :Male;
.
<#person2> a jponto:Person;
 jponto:hasGender jponto:Female;
.
<#action1> a jponto:Carrying;
 jponto:subject <#person1>;
 jponto:object [
    rdf:type jponto:Hat;
 ];
.
<#action2> a jponto:Communication;
 jponto:subject  <#person1>;
 jponto:subject <#person1>;
.
```


8.3.3 Taxonomy of classes of JPOnto-visual

8.3.3.1 General

The root classes of JPOnto-Visual (connected by rdfs:subClassOf edges) are shown in Figure 4.

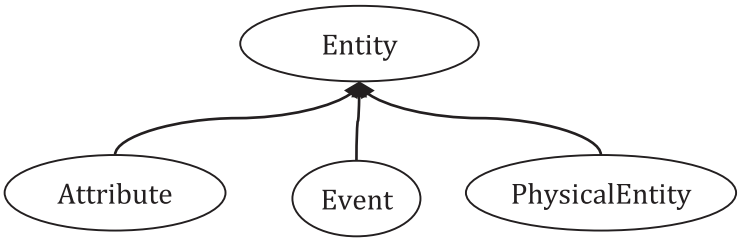

Figure 4 — Root classes of JPOnto-visual (arrows represent *rdfs:subClassOf* relationships)

(informative) Design principle: Selection of classes and properties from WordNet

JPOnto-Visual does not attempt to model all possible entities depicted by the images, it just provides a taxonomy of root classes and properties, a small set of specific classes and properties and an extensibility mechanism. The classes and properties included in JPOnto-Visual, along with their definitions and rdfs: subClassOf/ rdfs:subPropertyOf relationships, have been selected from the WordNet lexical database for the English language.

8.3.3.2 Subclasses of *jponto:PhysicalEntity*

Subclasses of *jponto:PhysicalEntity* (connected by rdfs:subClassOf edges) are shown in Figure 5. A set of specific sublclasses are also defined (*jponto:Rain, jponto:Combustion, jponto:Face, jponto:Monuntain, jponto:Water, jponto:Car*, etc.) but not shown in Figure 5.

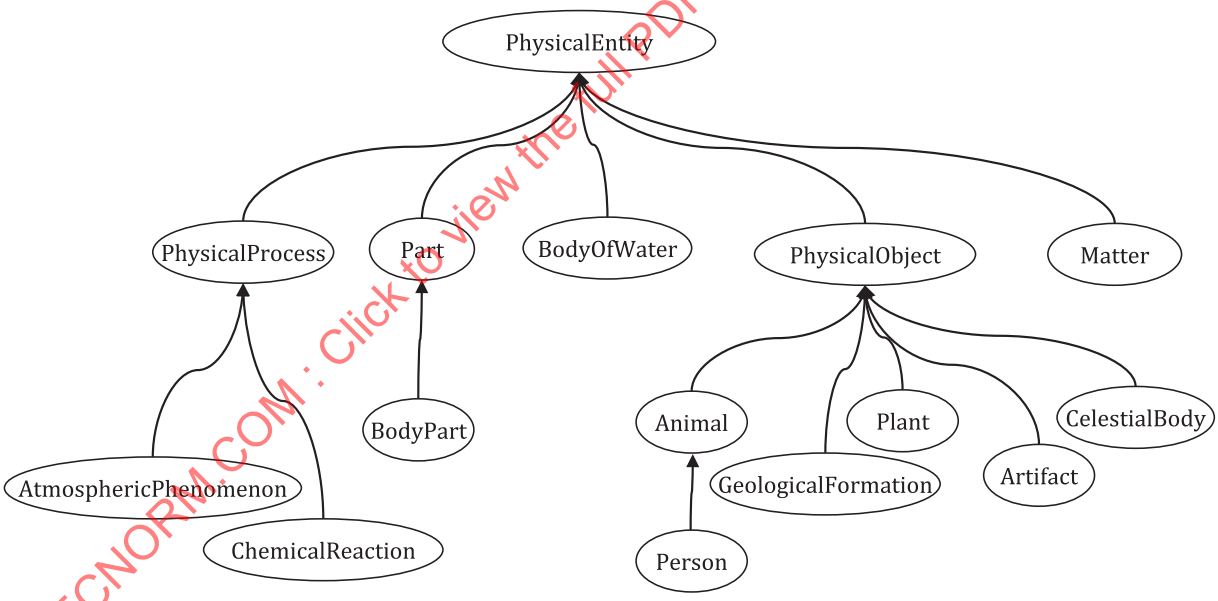

Figure 5 — Subclasses of *jponto:PhysicalEntity*

Table 1 shows all specific subclasses of *jponto:PhysicalEntity* defined within this document.

Table 1 — Subclasses of *jponto:PhysicalEntity*

Subclasses of AtmosphericPhenomenon	Aurora, Cloud, Storm, Meteor, Sunset, Wind, Snow, Rain
Subclasses of ChemicalReaction	Fire
Subclasses of BodyPart	Head, Hand, Face, Lip, Shoulder, Thorax, Venter, Toe, Finger, Back
Subclasses of BodyOfWater	River, Lake, Sea, Ocean, Channel, Bay, Waterfall, Stream, Gulf
Subclasses of Animal	Person, Dog, Cat, Mouse
Subclasses of GeographicalFormation	Mountain, Hill, Ridge, Delta, Cave, Beach, Crater

Table 1 *(continued)*

Subclasses of Artifact	Car, Bicycle, House, Gun, Hat, Shoe, Ball
Subclasses of Plant	Oak, Pine, Grass, Flower
Subclasses of CelestialBody	Sun, Moon
Subclasses of Matter	Water, Plastic, Food, Crystal, Coal, Wood, Quartz, Dust, Foam, Paper, Earth

8.3.3.3 Subclasses of *jponto:Event*

Subclasses of *jponto:Event* (connected by rdfs:subClassOf edges) are shown in Figure 6.

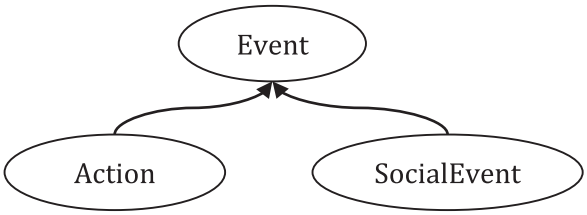

Figure 6 — Subclasses of *jponto:Event*

Table 2 shows all specific subclasses of *jponto:Event* defined within this document.

Table 2 — Subclasses of *jponto:Event*

Subclasses of Action	Playing, Wearing, Aggresion, Aggresion, Touching, Carrying, Communication
Subclasses of SocialEvent	Wedding, BirthdayParty, Circus, Demonstration, Carnival, Play, Funeral

(informative) Design principle: Minimization of description choices

When multiple WordNet terms may be used to describe the same facts, only the ones with more descriptive power have been included. For instance, activities such as Work, Game, etc. have been discarded as they can be described with more detail with subclasses of jponto:Action.

8.3.3.4 Subclasses of *jponto:Attribute*

Subclasses of *jponto:Attribute* (connected by rdfs:subClassOf edges) are shown in Figure 7.

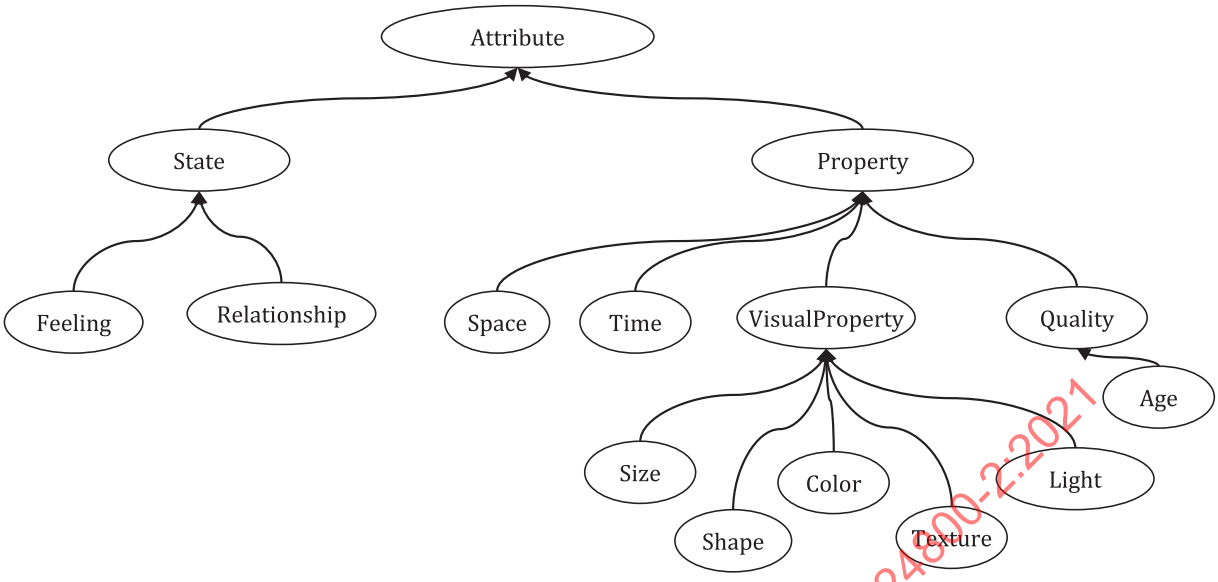

Figure 7 — Subclasses of *jponto:Attribute*

Table 3 shows all specific subclasses of *jponto:Attribute* defined within this document.

Table 3 — Subclasses of *jponto:Attribute*

Subclasses of Feeling	Pain, Pleasure, Astonishment, Fear, Desire, Anger, Anger Sadness, Happiness, Dislike, Love
Subclasses of Relationship	Parentage, MaritalRelationship
Subclasses of Size	Big, Small, Huge
Subclasses of Shape	Round, Circle, Line, Square
Subclasses of Color	Red, Green, Blue, Pink, Olive, Black, White, Yellow, Grey, Orange, Purple, White
Subclasses of Texture	Smoothness, Roughness
Subclasses of Light	Night, Sunlight, Moonlight, Shadow, Darkness, Twilight, Incandescence
Subclasses of Age	Youngness, Oldness

8.3.4 Properties of JPOnto-visual

Figure 8 shows the generic properties of JPOnto-Visual (connected by rdfs:subPOf edges).

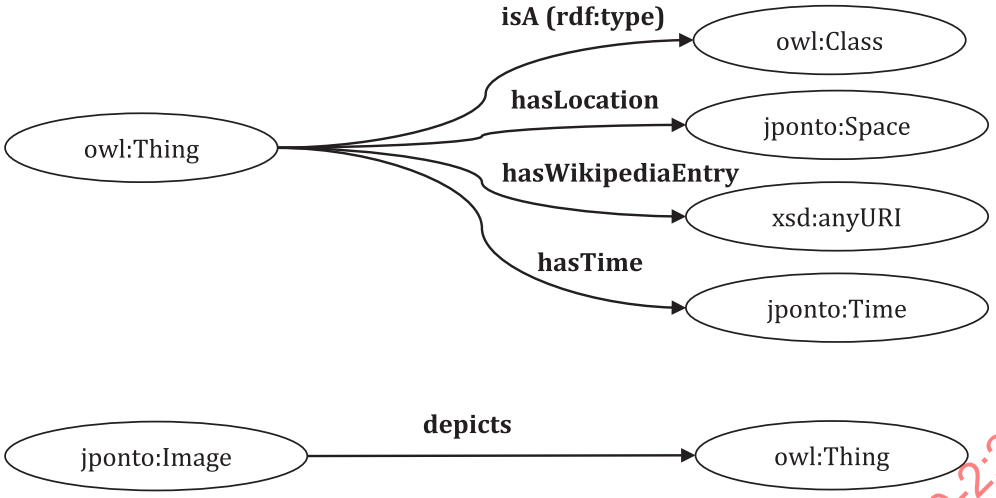

Figure 8 — Generic properties of JPOnto-visual (arrows show the domain and range of the property)

Specific properties of JPOnto-Visual (connected by rdfs:subPOf edges) are shown in Figure 9.

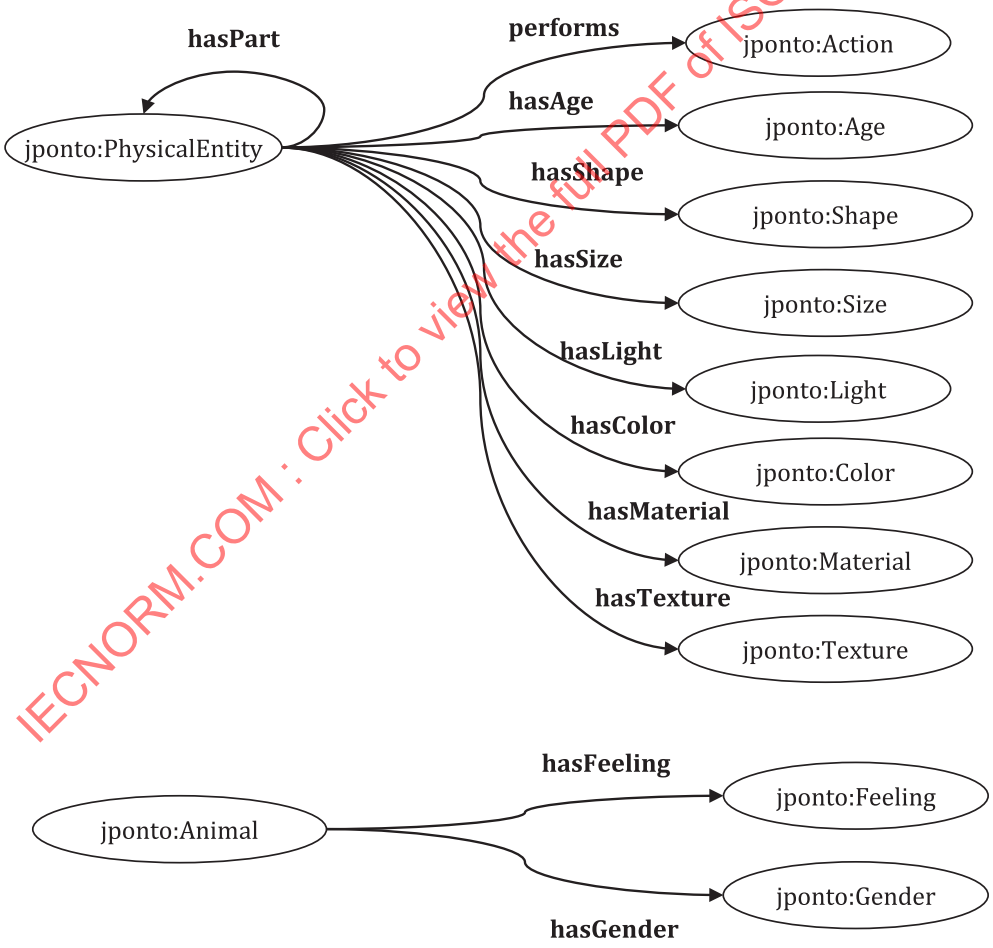

Figure 9 — Specific properties of JPOnto-visual (arrows show the domain and range of the property)