

ISO/IEC JTC 1/SC 32 N 1108

Date: 2004-05-09

REPLACES: --

<p style="text-align: center;">ISO/IEC JTC 1/SC 32</p> <p style="text-align: center;">Data Management and Interchange</p> <p style="text-align: center;">Secretariat: United States of America (ANSI)</p> <p style="text-align: center;">Administered by Pacific Northwest National Laboratory on behalf of ANSI</p>
--

DOCUMENT TYPE	Working Draft Text (for information or comment)
TITLE	ISO/IEC WD 19763-3 Information technology - Framework for Metamodel interoperability Part 3: Metamodel for ontologies
SOURCE	Project Editor – Wuhan University , P.R.China
PROJECT NUMBER	1.32.22.01.03.00
STATUS	For Review and Comment
REFERENCES	
ACTION ID.	FYI
REQUESTED ACTION	
DUE DATE	
Number of Pages	25
LANGUAGE USED	English
DISTRIBUTION	P & L Members SC Chair WG Conveners and Secretaries

Douglas Mann, Secretariat, ISO/IEC JTC 1/SC 32

Pacific Northwest National Laboratory *, 13600 Angelica Court, Chantilly, VA, 20151-3360,
United States of America

Telephone: +1 202-566-2126; Facsimile: +1 202-566-1639; E-mail: MannD@battelle.org

available from the JTC 1/SC 32 WebSite <http://www.jtc1sc32.org/>

*Pacific Northwest National Laboratory (PNL) administers the ISO/IEC JTC 1/SC 32 Secretariat on behalf of ANSI

**Information Technology – Framework for Metamodel Interoperability-- Part-3 :
Metamodel Framework for Ontology**

Composed By

SKLSE, Wuhan University , P.R.China

2004-03

Copyright notice

This ISO document is a Draft International Standard and is copyright-protected by ISO. Except as permitted under the applicable laws of the user's country, neither this ISO draft nor any extract from it may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, photocopying, recording or otherwise, without prior written permission being secured.

Requests for permission to reproduce should be addressed to either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.ch
Web www.iso.ch

Reproduction may be subject to royalty payments or a licensing agreement.

Violators may be prosecuted.

Contents

Introduction	6
1 Scope	7
1.1 Scope- Structure of MetaModel Framework For Ontology	7
1.2 Problems to be resolved	7
1.3 Scope-Exclusions	8
1.4 Overview of MMF Core Model	8
2 Normative references	8
3 Definition	9
3.1 Definitions of Metamodel Constructs	9
3.2 Terms used in this part of ISO/IEC WD19763-3	10
4 Structure of a MetaModel Framework for Ontology	11
4.1 Overview of MMF for Ontology.....	11
4.2 MMF for Ontology.....	12
4.2.1 AllDifferent	12
4.2.2 AnnotationProperty	12
4.2.3 DataRange	13
4.2.4 DatatypeProperty.....	13
4.2.5 DeprecatedClass.....	14
4.2.6 DeprecatedDatatypeProperty.....	14
4.2.7 DeprecatedObjectProperty	14
4.2.8 List.....	15
4.2.9 Literal	15
4.2.10 Nothing.....	15
4.2.11 ObjectProperty	16
4.2.12 Ontology.....	16
4.2.13 OntologyClass	16
4.2.14 OntologyConstruct	18
4.2.15 OntologyProperty	19
4.2.16 Property	19
4.2.17 Restriction	20
4.2.18 ReservedValue.....	21
4.2.19 Thing	21
5 Abstract Syntax (Normative)	22
5.1 Ontologies	22
5.2 Facts	23
5.3 Axiom.....	23

Table of Figures

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. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this part of ISO/IECWD 19763 may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC xxxxx was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information Technology*, Subcommittee SC 32, *Data Management and Interchange*.

ISO/IEC xxxxx consists of the following parts, under the general title *Information technology — Framework for Metamodel Interoperability*:

- Part 1: Reference Model
- Part 2: Core Model
- Part 3: Metamodel framework for Ontology
- Part 4: Metamodel framework for Mapping

Introduction

Due to the proliferation of E-Business or E-Commerce through the internet, the effective exchange and the interchange of business transactions or other related information across countries and cultures became the first concerns for peoples in both IT industry and other non-IT industries.

To follow the current trends of EB or EC, a lot of industrial consortia have been in charge of standardization of domain specific business objects including business process models and software components using common modeling facilities and exchanging facilities such as UML and XML. They are endeavor to standardize domain specific business process models which represent the best practices of businesses, and standard modeling constructs such as data elements, entity profiles and value domains, at each business domains.

Metamodels define the semantic of modeling elements and constructs that will be used to model the universe of discourse . One of the things to be mentioned today is that most of those standard efforts tend to be focused on the contents of metamodel to represent and exchange the semantics of businesses, using the UML stereotype mechanism and the XML.

The development of metamodels and UML profiles has been progressed through standardization activities such as UN/CEFACT and OASIS for UMM, ebXML, and OMG for MOF, XMI, CWM, EDOC, EAI, etc. However, every standard group has to specify their metamodel scheme by their own manners. There is no explicit specification about the modeling elements and constructs we use to build metamodels. Due to lack of standards that specify common bases for consistent development and registration of metamodels, fudge duplications and inconsistencies have to be brought. A unified framework for classifying and registering normative model elements could be required to establish harmonization of the metamodels, which are developed independently and to reuse them widely across organizations. And an ontology framework for metamodels is needed to ensure the interoperability between metamodels.

A useful de facto standard or draft standard developed by a standardization organization may be taken up and established as an IS of ISO/IEC/JTC1. Also it is meaningful to build a registry for metamodels based on IS or de facto standard in order to share the information about those model elements. When defining a business object model according to a metamodel and UML profile, stereotype, pattern, component, framework etc. are basic modelling construct elements to be referred as normative. The business model and information system model within an enterprise or among enterprises should be developed consistently based on those normative elements.

Ontology is a word borrowed from philosophy, in which it refers to the existence of entities. In knowledge sharing, the meaning of ontology is more general. It is used to describe the specification of formal vocabulary, specifically the universe of discourse and the describable relationships among the objects .

An ontology is the first step in knowledge sharing because it is the basis of various knowledge representations. And interoperation can be fulfilled between different representations as long as the ontology governing them is the same .To ensure the knowledge sharing between metamodels, an ontology framework for metamodeling should be specified. The ontology framework will depict what aspects of model elements and constructs we will meet in metamodelling.

Information Technology–Framework for Metamodel interoperability –Part 2:Core Model

1 Scope

The primary purpose of ISO/IEC WD 17963-3 is to specify the Meta Model Framework For Ontology. ISO/IEC WD 17963-3 also specifies the metamodel, which is required to describe metamodel items around ontology issue and inherits the core model.

1.1 Scope- Structure of MetaModel Framework For Ontology

Metamodel Framework for Ontology is a part of metamodel framework family of standards. Its target is ontology domain and it is developed by inheriting Core Model. See Fig. 1

All concepts in Metamodel Framework for Ontology could and should be tracked back in Core model. That's to say Core model defines the meta concept of items in MMF for Ontology, which is a specification responsible for guiding constructing ontology Schema and ontology classification.

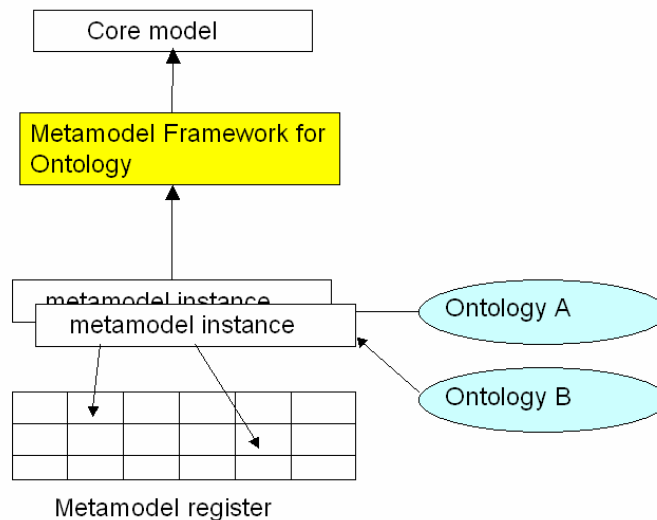


Fig. 1 Idea for Metamodel Framework for Ontology

1.2 Problems to be resolved

The information and data mount up greatly and rapidly with expand of Internet technology into business areas. To efficiently organize that great amount of data and promote searching required by business affairs, ontology is widening its use among different kind of domain, such as bank, stock and so on. Around ontology issue, there are many formats representing various mechanisms describing the existence of natural world. By these means, natural world and terms could be classified and the activities of some procedures could also be represented for machine readability. Since these describing tools among different developers don't adopt the same way outlining how an ontology structure is organized, the interoperability between them is cumbered. Core model, which is a kind of standards, stereotypes and patterns conforming its standard and is applied to activities including inter-enterprise connection and the stability and reusability of standard model, offers a guide principle to specify standard in target domains including ontology. Through inheriting Core model, MMF for ontology unifies the different concepts and eliminates conflicts existing among customers, trading partners and business partners. MMF for ontology builds a middleware neutral architecture, so the model conforming it would be easy to transferred from one platform into another one.

This standard represents a normative metamodel framework, which enables harmonized compliance to the normative identification and classification Schema because the sharing and the reusing of heterogeneous business objects among

different organizations have to rely upon the using common identification and classification Schema.

Today, a lot of consortia or organization defined those Schemas with their own manners in the term of the ontology. This standard provides a metamodel framework to enable both registering and using of a particular set of terminologies to represent an ontology Schema.

1.3 Scope-Exclusions

Followings are not covered in the scope of MMF for ontology

- Instruction of establishing an individual ontology Schema
- Standardization contents of ontology constructs
- Guarantee of the correctness of classifying objects

1.4 Overview of MMF Core Model

2 Normative references

The following referenced documents are indispensable for the application 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.

[ISO 8601:2000](#), Data elements and interchange formats – Information exchange – Representation of dates and times

[ISO/IEC 11179-1](#), Information technology – Metadata registries (MDR) - Part 1 : Framework

[ISO/IEC 11179-2](#), Information technology – Metadata registries (MDR) - Part 2 : Classification

[ISO/IEC 11179-4](#), Information technology – Metadata registries (MDR) - Part 4 : Formulation of data definitions

[ISO/IEC 11179-5](#), Information technology – Metadata registries (MDR) - Part 5 : Naming and identification principles

[ISO/IEC 11179-6](#), Information technology – Metadata registries (MDR) - Part 6 : Registration

[ISO/IEC 11404:1996](#), Information technology – Programming languages, their environments and system software interfaces – Language-independent datatypes

[ISO 12620:1999](#), Computer applications in terminology – Data categories

[ISO/IEC 19501-1:2002](#), Information technology – Unified Modeling Language (UML) – Part 1: Specification

3 Definition

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

3.1 defines metamodel constructs, used in specifying the registry metamodel.

3.2 lists terms, and their definitions, used in this document that are not included in 3.1.

An alphabetical list of terms from all three Clauses is provided in Annex A.

3.1 Definitions of Metamodel Constructs

This subclause defines the metamodel constructs used in specifying the registry metamodel in Clause 4.

3.1.1

association

⟨metamodel⟩ a semantic **relationship** between two **classes**

NOTE An **association** is a type of **relationship**.

[Adapted from ISO/IEC 19501-1:2001 2.5.2.3]

3.1.2

association class

⟨metamodel⟩ an **association** that is also a **class**

NOTE It not only connects a set of **classes**, but also defines a set of features that belong to the **relationship** itself.

[Adapted from ISO/IEC 19501-1:2001 2.5.2.4]

3.1.3

attribute

⟨metamodel⟩ a **characteristic** of an **object** or **entity**

3.1.4

class

⟨metamodel⟩ a description of a set of **objects** that share the same **attributes**, operations, methods, **relationships**, and semantics

[ISO/IEC 19501-1:2001 2.5.2.9]

3.1.5

composite attribute

⟨metamodel⟩ an **attribute** whose **datatype** is non-atomic

3.1.6

composite datatype

⟨metamodel⟩ a **datatype** that is also a **class**

NOTE A **composite datatype** is used as a **datatype** for a **composite attribute**.

3.1.7

generalization

⟨metamodel⟩ a **relationship** between a more general **class** (the parent) and a more specific **class** (the child) that is fully consistent with the first **class** (i.e. it has all of its **attributes** and **relationships**) and that adds additional information.

NOTE A **generalization** is a type of **relationship**.

[Adapted from ISO/IEC 19501-1:2001 2.5.2.24]

3.1.8

identifier (in Metadata Registry)

(metamodel) a sequence of characters, capable of uniquely identifying that with which it is associated, within a specified context

NOTE 1 A name should not be used as an identifier because it is not linguistically neutral.

3.1.9

relationship (in registry metamodel)

(metamodel) a connection among model elements

NOTE In ISO/IEC 11179-3, a relationship is either an association or a generalization.

[ISO/IEC 19501-1:2001 2.5.2.36]

3.2 Terms used in this part of ISO/IEC WD19763-3

AllDifferent

AnnotationProperty

DataRange

DatatypeProperty

DeprecatedClass

DeprecatedDatatypeProperty

DeprecatedObjectProperty

List

Literal

Nothing

ObjectProperty

Ontology

OntologyClass

OntologyConstruct

OntologyProperty

Property

Restriction

ReversedValue

Thing

4 Structure of a MetaModel Framework for Ontology

4.1 Overview of MMF for Ontology

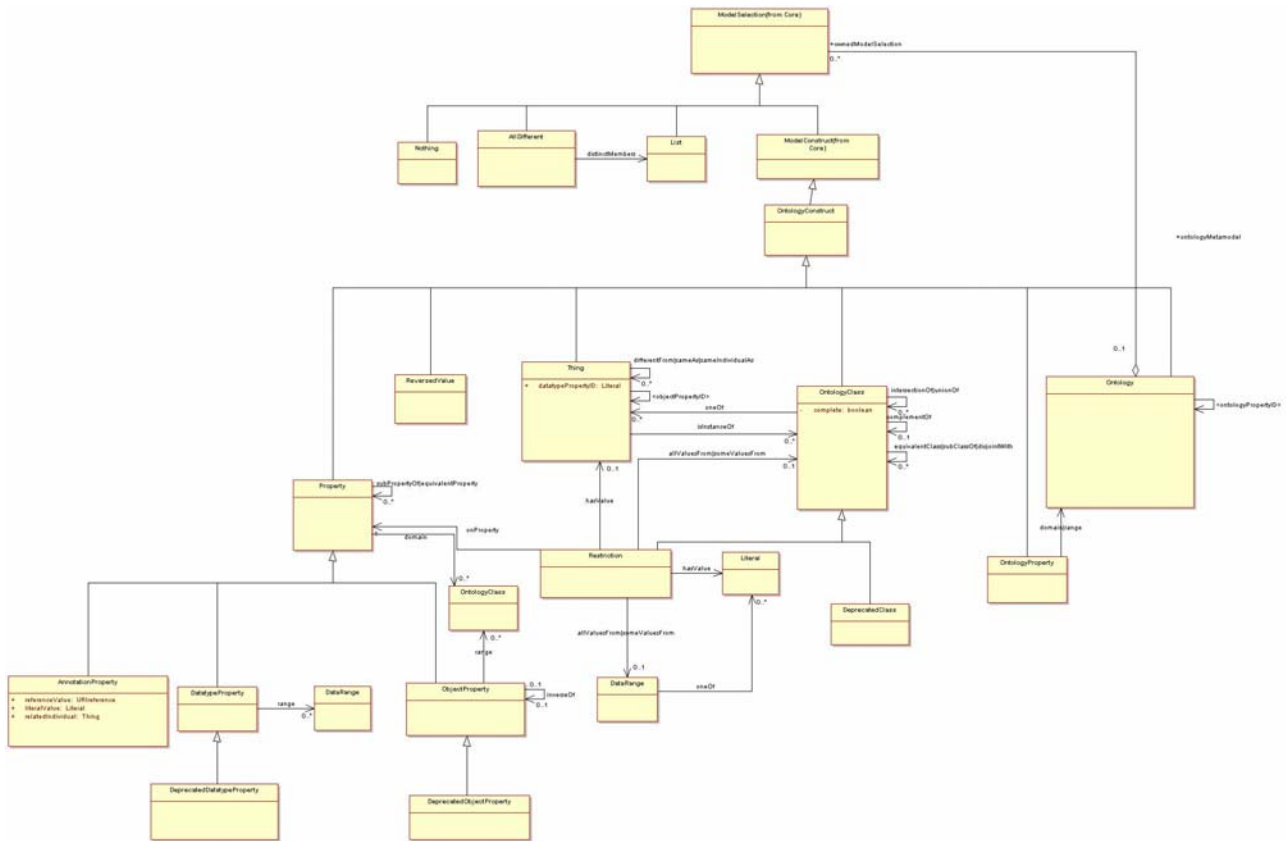


Fig.2 Metamodel Framework for Ontology

4.2 MMF for Ontology

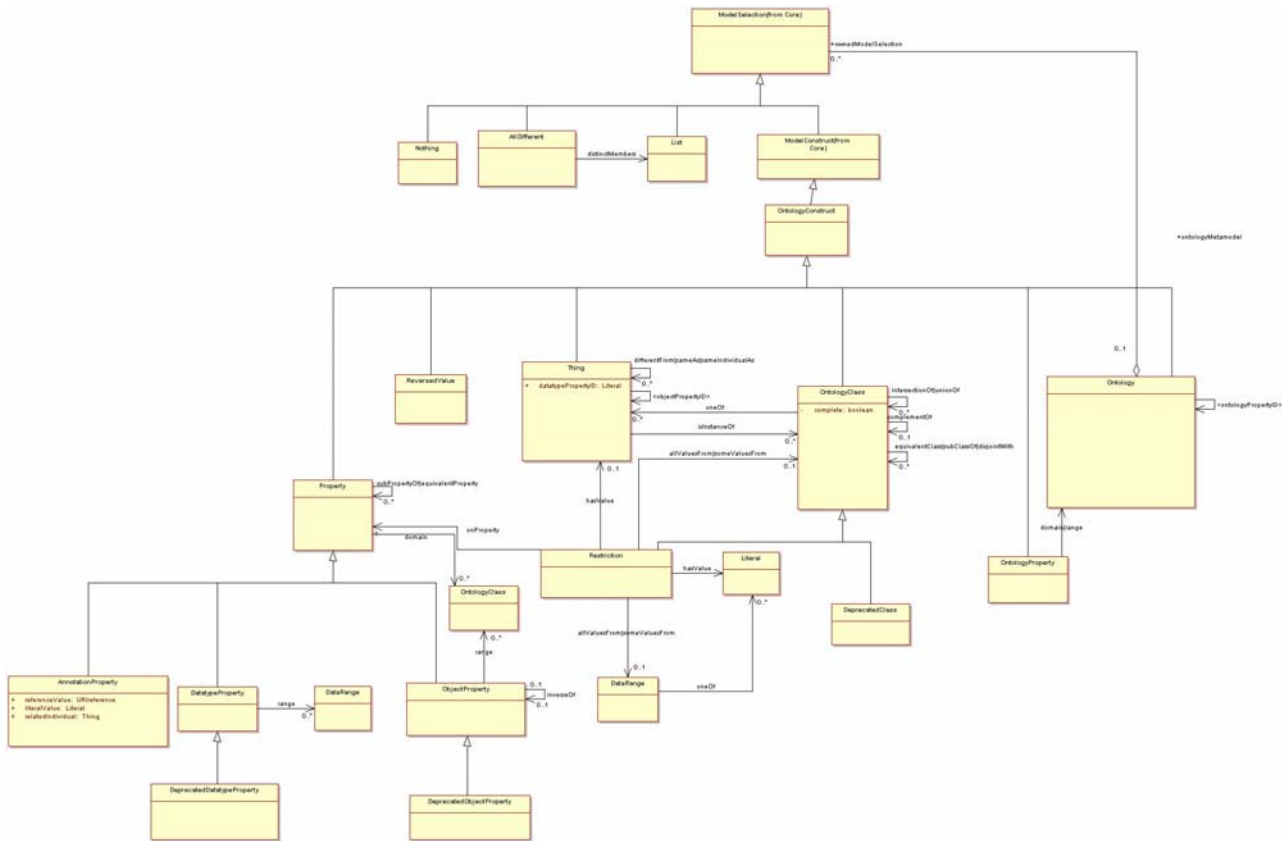


Fig.3 Metamodel Framework for Ontology

4.2.1 AllDifferent

<p>AllDifferent</p>	<p>For ontologies in which the unique-names assumption holds, all individuals have to be declared pairwise disjoint. This special element that links an instance of AllDifferent to a list of individuals. The intended meaning is that all individuals in the list are all different from each other. Its reference <i>distinctMembers</i> should point to a set of individuals' identifier. By thus, these individuals are different from each other.</p>		
Attribute or Reference	Occurrences	Datatype	Description
<i>distinctMembers</i>	0..1	List	<i>This List contains the Administration_Records of specified Things.</i>
constraints			

4.2.2 AnnotationProperty

AnnotationProperty	An annotation property relates a named element to an annotation. Annotation properties can be predefined or user defined. AnnotationProperty is the subclass of Property. The examples of AnnotationProperty are Label, Comment, VersionInfo and so on.		
Attribute or Reference	Occurrences	Datatype	Description
<i>literalValue</i>	<i>0..n</i>	<i>Literal</i>	
<i>referenceValue</i>	<i>0..n</i>	<i>Administration_Record</i>	<i>This is pointing to a individual or other OntologyConstruct owning a Administration Record</i>
<i>relatedIndividual</i>	<i>0..n</i>	<i>Thing</i>	
constraints			

4.2.3 DataRange

DataRange	A data range represents a range of data values. Data ranges are used to specify the range of datatype properties.		
Attribute or Reference	Occurrences	Datatype	Description
<i>oneOf</i>	<i>0..1</i>	<i>Literal</i>	This specifies the set of data values in literal format of this data range .
Constraints.			

4.2.4 DatatypeProperty

DatatypeProperty	A datatype property relates individuals to data values. Datatype properties provide relationships between instances of classes and instances of data ranges.		
Attribute or Reference	Occurrences	Datatype	Description
<i>range</i>	<i>0..n</i>	<i>DataRange</i>	This limits the value that this datatype property may have to the specified data ranges. Multiple data ranges are interpreted as stating that the range of this property is the intersection of the data ranges.

constraints

4.2.5 DeprecatedClass

DeprecatedClass	<p>A deprecated class is a subclass of <code>OntologyClass</code>. It represents an ontology class that is preserved for backward-compatibility purposes, but may be phased out in the future. So it means that <code>DeprecatedClass</code> should not be used in new document that commit to the <code>Ontology</code>.</p> <p><code>DeprecatedClass</code> designates the preferred <code>OntologyClass</code> by specifying <code>equivalentClass</code> inherited from <code>OntologyClass</code>.</p>		
Attribute or Reference	Occurrences	Datatype	Description
Constraints The id of a deprecated class must correspond to that of the ontology class that it deprecates.			

4.2.6 DeprecatedDatatypeProperty

DeprecatedDatatypeProperty	<p>A deprecated datatype property is a subclass of <code>DatatypeProperty</code>. It represents a datatype property that is preserved for backward-compatibility purposes, but may be phased out in the future. So it means that <code>DeprecatedDatatypeProperty</code> should not be used in new document that commit to the <code>Ontology</code>.</p> <p><code>DeprecatedDatatypeProperty</code> designates the preferred <code>DatatypeProperty</code> by specifying <code>equivalentProperty</code> inherited from <code>Property</code>.</p>		
Attribute or Reference	Occurrences	Datatype	Description
Constraints The id of a deprecated datatype property must correspond to that of the datatype property which it deprecates.			

4.2.7 DeprecatedObjectProperty

DeprecatedObjectProperty	<p>A deprecated object property is a subclass of <code>ObjectProperty</code>. It represents an object property that is preserved for backward-compatibility purposes, but may be phased out in the future. So it means that <code>DeprecatedObjectProperty</code> should not be used in new document that commit to the <code>Ontology</code>.</p> <p><code>DeprecatedObjectProperty</code> designates the preferred <code>ObjectProperty</code> by specifying <code>equivalentProperty</code> inherited from <code>Property</code>.</p>		
Attribute or Reference	Occurrences	Datatype	Description

Constraints The id of a deprecated object property must correspond to that of the object property that it deprecates.			

4.2.8 List

List	A list is an ordered collection of elements. User is allowed to decide how to define a List. Example ways include containing element identifier, referencing to its sub-list in recursion, and so on.		
Attribute or Reference	Occurrences	Datatype	Description
Constraints.			

4.2.9 Literal

Literal	Literals represent data values.		
Attribute or Reference	Occurrences	Datatype	Description
Constraints.			

4.2.10 Nothing

Nothing	Class extension of Nothing is the empty set.		
Attribute or Reference	Occurrences	Datatype	Description
Constraints. Nothing is the subclass of every class.			

4.2.11 ObjectProperty

ObjectProperty	An object property relates individuals to individuals. Object properties provide relationships between instances of two classes.		
Attribute or Reference	Occurrences	Datatype	Description
<i>range</i>	<i>0..n</i>	<i>OntologyClass</i>	This limits the individuals that this object property may have as its value to the class extension of the specified classes. Multiple range classes are interpreted as stating that the range of this property is the intersection of the class extension of the classes.
<i>inverseOf</i>	<i>0..1</i>	<i>ObjectProperty</i>	This states that the specified object property is the inverse of this object property. This association is symmetric. That is, if A is an inverse of B, then B is an inverse of A.
constraints			

4.2.12 Ontology

Ontology	An ontology includes definitions of basic concepts within a domain or across domains and the relationships among them, using the elements, such as classes, individuals, and properties etc.		
Attribute or Reference	Occurrences	Datatype	Description
<i>ontologyPropertyID</i>	<i>0..n</i>	<i>OntologyProperty.</i> <i>Administrator_Record</i>	This indicates that this ontology is related to the referenced ontology by a ontology property, which could be built-in or defined by user
Constraints.			

4.2.13 OntologyClass

OntologyClass	<p>OntologyClass defines a classification over a set of ontology instances by defining the behavior they exhibit. Every class is associated with a set of individuals, called the class extension.</p> <p>OntologyClass inherits the metaclass OntologyConstruct. OntologyClasses use Property to describe the relationship between them.</p> <p>OntologyClasses are described through “class descriptions”, which can be combined into “class axioms”. There are six types of class descriptions</p> <ul style="list-style-type: none"> • a class Administration_Record • an exhaustive enumeration of individuals • a property restriction • the complement of a class description • the intersection of two or more class descriptions • the union of two or more class descriptions 		
Attribute or Reference	Occurrences	Datatype	Description
<i>complete</i>	<i>0..1</i>	<i>boolean</i>	This specifies the modality of the class description, i.e., whether the class description is complete or partial. The default is “false”.
<i>complementOf</i>	<i>0..1</i>	<i>OntologyClass</i>	This describes a class for which the class extension contains exactly those individuals that do <i>not</i> belong to the class extension of the class description that is the object of the statement.
<i>disjointWith</i>	<i>0..n</i>	<i>OntologyClass</i>	This allows one to say that the class extension of a class description has no members in common with the class extension of another class description.
<i>equivalentClass</i>	<i>0..n</i>	<i>OntologyClass</i>	This links a class description to another class description. The meaning of it is that the class descriptions involved have the same class extension
<i>intersectionOf</i>	<i>0..n</i>	<i>OntologyClass</i>	This links a class to a list of class descriptions.intersectionOf describes a class for which the class extension contains precisely those individuals that are

			members of the class extension of all class descriptions in the list.
<i>oneOf</i>	<i>0..n</i>	<i>Thing</i>	This specify a list of individuals that are the instances of the class. This enables a class to be described by exhaustively enumerating its instances. The class extension of a class described with <i>oneOf</i> contains exactly the enumerated individuals, no more, no less.
<i>subClassOf</i>	<i>0..n</i>	<i>OntologyClass</i>	If the class description C1 is defined as a subclass of class description C2, then the set of individuals in the class extension of C1 should be a subset of the set of individuals in the class extension of C2. A class is by definition a subclass of itself (as the subset may be the complete set).
<i>unionOf</i>	<i>0..n</i>	<i>OntologyClass</i>	This describes an class for which the class extension contains those individuals that occur in at least one of the class extensions of the class descriptions in the list.
Constraints			

4.2.14 OntologyConstruct

OntologyConstruct	OntologyConstruct inherits ModelConstruct. It plays a role of unit to establish a complete Ontology. The instances of OntologyConstruct include class, property, individual and so on. Every OntologyConstruct should conform the definition of ModelConstruct, that is have a Administration_Record, and so on.		
Attribute or Reference	Occurrences	Datatype	Description

<i>annotationPropertyID</i>	<i>0..n</i>	<i>AnnotationProperty. Administrator_Record</i>	This indicates that this ontology construct is related to the referenced annotation, which may be a literal or individual type, and may be built-in or defined by user
Constraints. Exact one Administration_Record is mandatory required.			

4.2.15 OntologyProperty

OntologyProperty	<p>OntologyProperty is used to link two Ontologies which have some relation between them. OntologyProperty has two references: domain and range. They both point to a instance of a Ontology. The examples of OntologyProperty are like import, prior version, the ontology incompatible with and backcompatible with. User is permitted to define their own OntologyProperty if necessary.</p>		
Attribute or Reference	Occurrences	Datatype	Description
<i>domain</i>	<i>1..*</i>	<i>Ontology</i>	
<i>range</i>	<i>1..*</i>	<i>Ontology</i>	
constraints			

4.2.16 Property

Property	<p>Property is a metaclass designating the type of properties of an OntologyClass. Property is an instance of OntologyConstruct, it plays like a bridge connecting the OntologyClass and its property value. These values may be instances of other OntologyClass or just plain primitives, which are respectively defined by ObjectProperty and DatatypeProperty</p> <p>Every property is associated with a set of instances, called the property extension. Instances of properties are subject-object pairs. A property has an intensional meaning (the underlying concept) which is related but not equal to its property extension. Thus, two properties may have the same property extension, but still be different properties.</p>		
Attribute or Reference	Occurrences	Datatype	Description
<i>domain</i>	<i>0..n</i>	<i>OntologyClass</i>	
<i>subPropertyOf</i>	<i>0..1</i>	<i>Property</i>	This states that this property is the subproperty of the specified properties.
<i>equivalentProperty</i>	<i>0..n</i>	<i>Property</i>	This states that the specified properties have

			the same property extension with this property.
constraints			

4.2.17 Restriction

Restriction	Restriction is an anonymous class. It describes a class of all individuals which satisfy a particular property restriction.		
Attribute or Reference	Occurrences	Datatype	Description
<i>allValuesFrom</i>	<i>0..1</i>	<i>OntologyClass or DataRange</i>	It is used to describe a class of all individuals for which all values of the property under consideration are either members of the class extension of the class description or are data values within the specified data range.
<i>hasValue</i>	<i>0..1</i>	<i>Thing or Literal</i>	It describes a class of all individuals for which the property concerned has at least one value <i>semantically</i> equal to V (it may have other values as well). For datatypes "semantically equal" means that the lexical representation of the literals maps to the same value. For individuals it means that they either have the same Administration_Record or are defined as being the same individual
<i>someValuesFrom</i>	<i>0..1</i>	<i>OntologyClass or DataRange</i>	It describes a class of all individuals for which at least one value of the property concerned is an instance of the class description or a data value in the data range.

<i>onProperty</i>	<i>0..1</i>	<i>Property</i>	This indicates that the specified property is the property under consideration.
<i>cardinality minCardinality maxCardinality</i>	<i>0..1</i>	<i>NonNegativeInteger</i>	This is used to describe a class of all individuals that have exactly at least at most N semantically distinct values (individuals or data values) for the property under consideration, where N is the value of the cardinality constraint.
Constraints			

4.2.18 ReservedValue

ReservedValue	ReservedValue is some data value, such as individual, particular literal value, which are reserved terms defined by the administrator of Ontology. User is permitted to choose directly from instances of ReservedValue, but not allowed to create a new one to take the place it. That means everything defined by user should not conflict with existing ReservedValue instance. Instances of ReservedValue could be that of every OntologyConstruct		
Attribute or Reference	Occurrences	Datatype	Description
Constraints.			

4.2.19 Thing

Thing	Class extension of Thing is the set of all individuals.		
Attribute or Reference	Occurrences	Datatype	Description
<i>dataPropertyID</i>	<i>0..1</i>	<i>Literal</i>	This indicates that this individual is related to the specified data value by a user defined datatype property.
<i>objectPropertyID</i>	<i>0..1</i>	<i>ObjectProperty.</i>	This indicates that this individual is related to the

		<i>Administration_Record</i>	specified individual by a object property, user defined or built in
<i>differentFrom</i>	<i>0..n</i>	<i>Thing</i>	It links this individual to an individual on the specified list and indicates that the two individuals refer to two different things: the individuals are different.
<i>sameAs</i>	<i>0..n</i>	<i>Thing</i>	This links this individual to an individual on the specified list and indicates that the two individuals refer to the same thing: the individuals are identical.
<i>sameIndividualAs</i>	<i>0..n</i>	<i>Thing</i>	This links this individual to an individual on the specified list and indicates that the two individuals refer to the same thing: the individuals are identical.
<i>isInstanceOf</i>	<i>0..n</i>	<i>OntologyClass</i>	This relates this individual to the specified classes of which it is a member of the class extension.
Constraints. Every class is a subclass of Thing.			

5 Abstract Syntax (Normative)

The abstract syntax is specified here by means of a version of Extended BNF, very similar to the EBNF notation used for XML. Terminals are quoted; non-terminals are not quoted. Alternatives are either separated by vertical bars (|) or are given in different productions. Components that can occur at most once are enclosed in square brackets ([...]); components that can occur any number of times (including zero) are enclosed in braces ({...}). Whitespace is ignored in the productions here.

The EBNF notation to MMF for Ontology is divided into 3 parts, Ontologies, Facts and Axioms.

5.1 Ontologies

Define what an Ontology is consist of.

```
ontology ::= 'Ontology(' [ ontologyID ] { directive } ')'
directive ::= 'Annotation(' ontologyPropertyID ontologyID ')'
           | 'Annotation(' annotationPropertyID Administration_Record ')'
           | {axiom}
           | {fact}
```

```
datatypeID ::= Administration_Record
```

```
ontologyClassID ::= Administration_Record
```

```

individualID ::= Administration_Record
ontologyID ::= Administration_Record
datatypePropertyID ::= Administration_Record
objectPropertyID ::= Administration_Record
annotationPropertyID ::= Administration_Record
ontologyPropertyID ::= Administration_Record

```

```

annotation ::= 'annotation(' annotationPropertyID Administration_Record ')'
            | 'annotation(' annotationPropertyID Literal ')'
            | 'annotation(' annotationPropertyID individual ')'

```

5.2 Facts

Define how to describe the Facts appearing in Ontology.

```

fact ::= individual
individual ::= 'Individual(' [ individualID ] { annotation } { 'isInstanceOf' isInstanceOf } { value } ')'
value ::= 'value(' objectPropertyID individualID ')'
        | 'value(' objectPropertyID individual ')'
        | 'value(' datatypePropertyID Literal ')'

```

```

isInstanceOf ::= restriction

```

```

fact ::= 'SameAs(' individualID individualID {individualID} ')'
        | 'SameIndividualAs(' individualID individualID {individualID} ')'
        | 'DifferentFrom(' individualID individualID {individualID} ')'

```

5.3 Axiom

Define the production rule of the elements of Ontology.

```

axiom ::= 'OntologyClass(' OntologyClassID ['Deprecated'] modality { annotation } { description } ')'
modality ::= 'complete' | 'partial'

```

```

axiom ::= 'List(' OntologyClassID ['Deprecated'] { annotation } { individualID } ')'

```

```

axiom ::= 'DisjointWith(' description description { description } ')'
        | 'EquivalentClasses(' description { description } ')'
        | 'SubClassOf(' description description ')'

```

```

axiom ::= 'Datatype(' datatypeID ['Deprecated'] { annotation } ')'

```

```

description ::= OntologyClassID
            | restriction
            | 'unionOf(' { description } ')'
            | 'intersectionOf(' { description } ')'
            | 'complementOf(' description ')'
            | 'oneOf(' { individualID } ')'

```

```
restriction ::= 'restriction(' datatypePropertyID dataRestrictionComponent { dataRestrictionComponent } )'  
            | 'restriction(' objectPropertyID individualRestrictionComponent { individualRestrictionComponent }'  
            )'
```

```
dataRestrictionComponent ::= 'allValuesFrom(' dataRange )'  
                            | 'someValuesFrom(' dataRange )'  
                            | 'hasValue(' Literal )'  
                            | cardinality
```

```
individualRestrictionComponent ::= 'allValuesFrom(' description )'  
                                  | 'someValuesFrom(' description )'  
                                  | 'hasValue(' individualID )'  
                                  | cardinality
```

```
cardinality ::= 'minCardinality(' non-negative-integer )'  
              | 'maxCardinality(' non-negative-integer )'  
              | 'cardinality(' non-negative-integer )'
```

```
dataRange ::= datatypeID | Literal  
           | 'oneOf(' { Literal } )'
```

```
axiom ::= 'DatatypeProperty(' datatypePropertyID ['Deprecated'] { annotation }  
         { 'subPropertyOf(' datatypePropertyID ) }  
         { 'domain(' description ) } { 'range(' dataRange ) } )'  
        | 'ObjectProperty(' objectPropertyID ['Deprecated'] { annotation }  
         { 'subPropertyOf(' objectPropertyID ) }  
         [ 'inverseOf(' objectPropertyID ) ]  
         { 'domain(' description ) } { 'range(' description ) } )'  
        | 'AnnotationProperty(' annotationPropertyID { annotation } )'  
        | 'OntologyProperty(' ontologyPropertyID { annotation } )'
```

```
axiom ::= 'EquivalentProperties(' datatypePropertyID datatypePropertyID { datatypePropertyID } )'  
        | 'SubPropertyOf(' datatypePropertyID datatypePropertyID )'  
        | 'EquivalentProperties(' objectPropertyID objectPropertyID  
                               { objectPropertyID } )'  
        | 'SubPropertyOf(' objectPropertyID objectPropertyID )'
```