CIDOC OWL

Comparison of

Definition of the CIDOC Conceptual Reference Model
Version 4.2.4
January 2008
Editors: Nick Crofts, Martin Doerr, Tony Gill, Stephen Stead, Matthew Stiff
http://cidoc.ics.forth.gr/docs/cidoc_crm_version_4.2.4_March2008_.pdf

and

OWL Web Ontology Language Guide
W3C Recommendation 10 February 2004
Editors: Michael K. Smith, Chris Welty, Deborah L. McGuinness
http://www.w3.org/TR/owl-guide/

| | CIDOC CRM | OWL |
|---|---|---|
| | | |
| Class | A class is a category of items that share one or more common traits serving as criteria to identify the items belonging to the class. These properties need not be explicitly formulated in logical terms, but may be described in a text (here called a scope note) that refers to a common conceptualisation of domain experts. The sum of these traits is called the intension of the class. A class may be the domain or range of none, one or more properties formally defined in a model. The formally defined properties need not be part of the intension of their domains or ranges: such properties are optional. An item that belongs to a class is called an instance of this class. A class is associated with an open set of real life instances, known as the extension of the class. Here "open" is used in the sense that it is generally beyond our capabilities to know all instances of a class in the world and indeed that the future may bring new instances about at any time (Open World). Therefore a class cannot be defined by enumerating its instances. A class plays a role analogous to a grammatical noun, and can be completely defined without reference to any other construct (unlike properties, | owl:Class :A class defines a group of individuals that belong together because they share some properties. For example, Deborah and Frank are both members of the class Person. Classes can be organized in a specialization hierarchy using subClassOf. There is a built-in most general class named Thing that is the class of all individuals and is a superclass of all OWL classes. There is also a built-in most specific class named Nothing that is the class that has no instances and a subclass of all OWL classes. |

| | | |
|---|---|---|
| | which must have an unambiguously defined domain and range). In some contexts, the terms individual class, entity or node are used synonymously with class.<br><br>For example:<br>Person is a class. To be a Person may actually be determined by DNA characteristics, but we all know what a Person is. A Person may have the property of being a member of a Group, but it is not necessary to be member of a Group in order to be a Person. We shall never know all Persons of the past. There will be more Persons in the future. | |
| Subclass | A subclass is a class that is a specialization of another class (its superclass). Specialization or the IsA relationship means that:<br>1. all instances of the subclass are also instances of its superclass,<br>2. the intension of the subclass extends the intension of its superclass, i.e. its traits are more restrictive than that of its superclass and<br>3. the subclass inherits the definition of all of the properties declared for its superclass without exceptions (strict inheritance), in addition to having none, one or more properties of its own.<br>A subclass can have more than one immediate superclass and consequently inherits the properties of all of its superclasses (multiple inheritance). The IsA relationship or specialization between two or more classes gives rise to a structure known as a class hierarchy. The IsA relationship is transitive and may not be cyclic. In some contexts (e.g. the programming language C++) the term derived class is used synonymously with subclass.<br>For example:<br>Every Person IsA Biological Object, or Person is a subclass of Biological Object. Also, every Person IsA Actor. A Person may die. However other kinds of Actors, such as companies, don’t die (c.f. 2). Every Biological Object IsA Physical Object. A Physical Object can be moved. Hence a Person can be moved also (c.f. 3). | rdfs:subClassOf: Class hierarchies may be created by making one or more statements that a class is a subclass of another class. For example, the class Person could be stated to be a subclass of the class Mammal. From this a reasoner can deduce that if an individual is a Person, then it is also a Mammal. |

| | | |
|---|---|---|
| Property | A property serves to define a relationship of a specific kind between two classes. The property is characterized by an intension, which is conveyed by a scope note. A property plays a role analogous to a grammatical verb, in that it must be defined with reference to both its domain and range, which are analogous to the subject and object in grammar (unlike classes, which can be defined independently). It is arbitrary, which class is selected as the domain, just as the choice between active and passive voice in grammar is arbitrary. In other words, a property can be interpreted in both directions, with two distinct, but related interpretations. Properties may themselves have properties that relate to other classes (This feature is used in this model only in order to describe dynamic subtyping of properties). Properties can also be specialized in the same manner as classes, resulting in IsA relationships between subproperties and their superproperties. In some contexts, the terms attribute, reference, link, role or slot are used synonymously with property.<br>For example:<br>“Physical Man-Made Thing depicts CRM Entity” is equivalent to “CRM Entity is depicted by Physical Man-Made Thing”. | rdf:Property: Properties can be used to state relationships between individuals or from individuals to data values. Examples of properties include hasChild, hasRelative, hasSibling, and hasAge. The first three can be used to relate an instance of a class Person to another instance of the class Person (and are thus occurences of ObjectProperty), and the last (hasAge) can be used to relate an instance of the class Person to an instance of the datatype Integer (and is thus an occurence of DatatypeProperty). Both owl:ObjectProperty and owl:DatatypeProperty are subclasses of the RDF class rdf:Property.<br><br>will use owl:ObjectProperty |
| subPropertyOf | A subproperty is a property that is a specialization of another property (its superproperty). Specialization or IsA relationship means that:<br>1. all instances of the subproperty are also instances of its superproperty,<br>2. the intension of the subproperty extends the intension of the superproperty, i.e. its traits are more restrictive than that of its superproperty,<br>3. the domain of the subproperty is the same as the domain of its superproperty or a subclass of that domain,<br>4. the range of the subproperty is the same as the range of its superproperty or a subclass of that range, | rdf:Property: Property hierarchies may be created by making one or more statements that a property is a subproperty of one or more other properties. For example, hasSibling may be stated to be a subproperty of hasRelative. From this a reasoner can deduce that if an individual is related to another by the hasSibling property, then it is also related to the other by the hasRelative property. |

| | | |
|---|---|---|
| | 5. the subproperty inherits the definition of all of the properties declared for its superproperty without exceptions (strict inheritance), in addition to having none, one or more properties of its own.<br>A subproperty can have more than one immediate superproperty and consequently inherits the properties of all of its superproperties (multiple inheritance). The IsA relationship or specialization between two or more properties gives rise to the structure we call a property hierarchy. The IsA relationship is transitive and may not be cyclic. Some object-oriented languages, such as C++, have no equivalent to the specialization of properties. | |
| superproperty | A superproperty is a property that is a generalization of one or more other properties (its subproperties), which means that it subsumes all instances of its subproperties, and that it can also have additional instances that do not belong to any of its subproperties. The intension of the superproperty is less restrictive than any of its subproperties. The subsumption relationship or generalization is the inverse of the IsA relationship or specialization. | |
| intension | The intension of a class or property is its intended meaning. It consists of one or more common traits shared by all instances of the class or property. These traits need not be explicitly formulated in logical terms, but may just be described in a text (here called a scope note) that refers to a conceptualisation common to domain experts. In particular the so-called primitive concepts, which make up most of the CRM, cannot be further reduced to other concepts by logical terms. | |
| extension | The extension of a class is the set of all real life instances belonging to the class that fulfil the criteria of its intension. This set is “open” in the sense that it is generally beyond our capabilities to know all instances of a class in the world and indeed that the future may bring new instances about at any time (Open World). | |

| | | |
|---|---|---|
| | An information system may at any point in time refer to some instances of a class, which form a subset of its extension. | |
| scope note | A scope note is a textual description of the intension of a class or property. Scope notes are not formal modelling constructs, but are provided to help explain the intended meaning and application of the CRM’s classes and properties. Basically, they refer to a conceptualisation common to domain experts and disambiguate between different possible interpretations. Illustrative example instances of classes and properties are also regularly provided in the scope notes for explanatory purposes. | rdfs:comment |
| Instance / Individual | An instance of a class is a real world item that fulfils the criteria of the intension of the class. Note, that the number of instances declared for a class in an information system is typically less than the total in the real world. For example, you are an instance of Person, but you are not mentioned in all information systems describing Persons. | Individuals are instances of classes, and properties may be used to relate one individual to another. For example, an individual named Deborah may be described as an instance of the class Person and the property hasEmployer may be used to relate the individual Deborah to the individual StanfordUniversity. |
| domain | The domain is the class for which a property is formally defined. This means that instances of the property are applicable to instances of its domain class. A property must have exactly one domain, although the domain class may always contain instances for which the property is not instantiated. The domain class is analogous to the grammatical subject of the phrase for which the property is analogous to the verb. It is arbitrary, which class is selected as the domain and which as the range, just as the choice between active and passive voice in grammar is arbitrary. Property names in the CRM are designed to be semantically meaningful and grammatically correct when read from domain to range. In addition, the inverse property name, normally given in parentheses, is also designed to be semantically meaningful and grammatically correct when read from range to domain. | rdfs:domain: A domain of a property limits the individuals to which the property can be applied. If a property relates an individual to another individual, and the property has a class as one of its domains, then the individual must belong to the class. For example, the property hasChild may be stated to have the domain of Mammal. From this a reasoner can deduce that if Frank hasChild Anna, then Frank must be a Mammal. Note that rdfs:domain is called a global restriction since the restriction is stated on the property and not just on the property when it is associated with a particular class. See the discussion below on property restrictions for more information. |
| range | The range is the class that comprises all potential values of a property. That means that instances of the property can link only to instances of its range class. A property must have exactly one range, although the range class may always contain instances that are not the value of the property. The range class is analogous to | rdfs:range: The range of a property limits the individuals that the property may have as its value. If a property relates an individual to another individual, and the property has a class as its range, then the other individual must belong to the range class. For example, the property hasChild may be stated to have the range of |

| | | |
|---|---|---|
| | the grammatical object of a phrase for which the property is analogous to the verb. It is arbitrary, which class is selected as domain and which as range, just as the choice between active and passive voice in grammar is arbitrary. Property names in the CRM are designed to be semantically meaningful and grammatically correct when read from domain to range. In addition the inverse property name, normally given in parentheses, is also designed to be semantically meaningful and grammatically correct when read from range to domain. | Mammal. From this a reasoner can deduce that if Louise is related to Deborah by the hasChild property, (i.e., Deborah is the child of Louise), then Deborah is a Mammal. Range is also a global restriction as is domain above. Again, see the discussion below on local restrictions (e.g. AllValuesFrom) for more information. |
| Inheritance | Inheritance of properties from superclasses to subclasses means that if an item x is an instance of a class A, then<br>1. all properties that must hold for the instances of any of the superclasses of A must also hold for item x, and<br>all optional properties that may hold for the instances of any of the superclasses of A may also hold for item x. | holds for RDFS, OWL Lite |
| Strict Inheritance | Strict inheritance means that there are no exceptions to the inheritance of properties from superclasses to subclasses. For instance, some systems may declare that elephants are grey, and regard a white elephant as an exception. Under strict inheritance it would hold that: if all elephants were grey, then a white elephant could not be an elephant. Obviously not all elephants are grey. To be grey is not part of the intension of the concept elephant but an optional property. The CRM applies strict inheritance as a normalization principle. | |
| Multiple Inheritance | Multiple inheritance means that a class A may have more than one immediate superclass. The extension of a class with multiple immediate superclasses is a subset of the intersection of all extensions of its superclasses. The intension of a class with multiple immediate superclasses extends the intensions of all its superclasses, i.e. its traits are more restrictive than any of its superclasses. If multiple inheritance is used, the resulting “class hierarchy” is a directed graph and not a tree structure. If it is represented as an indented list, there are necessarily repetitions of the same class at different positions in the list. For example, Person is both, an Actor and a Biological Object. | holds for RDFS, OWL Lite |

| | | |
|---|---|---|
| instance | An instance of a class is a real world item that fulfils the criteria of the intension of the class. Note, that the number of instances declared for a class in an information system is typically less than the total in the real world. For example, you are an instance of Person, but you are not mentioned in all information systems describing Persons. | see above  Individuals |
| endurant, perdurant | “The difference between enduring and perduring entities (which we shall also call endurants and perdurants) is related to their behaviour in time. Endurants are wholly present (i.e., all their proper parts are present) at any time they are present. Perdurants, on the other hand, just extend in time by accumulating different temporal parts, so that, at any time they are present, they are only partially present, in the sense that some of their proper temporal parts (e.g., their previous or future phases) may be not present. E.g., the piece of paper you are reading now is wholly present, while some temporal parts of your reading are not present any more. Philosophers say that endurants are entities that are in time, while lacking however temporal parts (so to speak, all their parts flow with them in time). Perdurants, on the other hand, are entities that happen in time, and can have temporal parts (all their parts are fixed in time).” (Gangemi et al. 2002, pp. 166-181). | related to domain modeling |
| shortcut | A shortcut is a formally defined single property that represents a deduction or join of a data path in the CRM. The scope notes of all properties characterized as shortcuts describe in words the equivalent deduction. Shortcuts are introduced for the cases where common documentation practice refers only to the deduction rather than to the fully developed path. For example, museums often only record the dimension of an object without documenting the Measurement that observed it. The CRM allows shortcuts as cases of less detailed knowledge, while preserving in its schema the relationship to the full information. | |
| monotonic reasoning | Monotonic reasoning is a term from knowledge representation. A reasoning form is monotonic if an addition to the set of propositions making up the knowledge base never determines a decrement in the set of conclusions that may be derived from the | OWL is monotonic |

| | | |
|---|---|---|
| | knowledge base via inference rules. In practical terms, if experts enter subsequently correct statements to an information system, the system should not regard any results from those statements as invalid, when a new one is entered. The CRM is designed for monotonic reasoning and so enables conflict-free merging of huge stores of knowledge. | |
| disjoint | Classes are disjoint if the intersection of their extensions is an empty set. In other words, they have no common instances in any possible world. | owl:disjointWith [OWL DL]<br>The disjointness of a set of classes can be expressed using the `owl:disjointWith` constructor. It guarantees that an individual that is a member of one class cannot simultaneously be an instance of a specified other class. |
| primitive | The term primitive as used in knowledge representation characterizes a concept that is declared and its meaning is agreed upon, but that is not defined by a logical deduction from other concepts. For example, mother may be described as a female human with child. Then mother is not a primitive concept. Event however is a primitive concept. Most of the CRM is made up of primitive concepts. | |
| Open World | The "Open World Assumption" is a term from knowledge base systems. It characterizes knowledge base systems that assume the information stored is incomplete relative to the universe of discourse they intend to describe. This incompleteness may be due to the inability of the maintainer to provide sufficient information or due to more fundamental problems of cognition in the system's domain. Such problems are characteristic of cultural information systems. Our records about the past are necessarily incomplete. In addition, there may be items that cannot be clearly assigned to a given class. In particular, absence of a certain property for an item described in the system does not mean that this item does not have this property. For example, if one item is described as Biological Object and another as Physical Object, this does not imply that the latter may not be a Biological Object as well. Therefore complements of a class with respect to a superclass cannot be concluded in general from an information system using | |

| | | |
|---|---|---|
| | the Open World Assumption. For example, one cannot list “all Physical Objects known to the system that are not Biological Objects in the real world”, but one may of course list “all items known to the system as Physical Objects but that are not known to the system as Biological Objects”. | |
| complement | The complement of a class A with respect to one of its superclasses B is the set of all instances of B that are not instances of A. Formally, it is the set-theoretic difference of the extension of B minus the extension of A. Compatible extensions of the CRM should not declare any class with the intension of them being the complement of one or more other classes. To do so will normally violate the desire to describe an Open World. For example, for all possible cases of human gender, male should not be declared as the complement of female or vice versa. What if someone is both or even of another kind? | |
| query containment | Query containment is a problem from database theory: A query X contains another query Y, if for each possible population of a database the answer set to query X contains also the answer set to query Y. If query X and Y were classes, then X would be superclass of Y. | |
| interoperability | Interoperability means the capability of different information systems to communicate some of their contents. In particular, it may mean that<br>1. two systems can exchange information, and/or<br>2. multiple systems can be accessed with a single method.<br>Generally, syntactic interoperability is distinguished from semantic interoperability. Syntactic interoperability means that the information encoding of the involved systems and the access protocols are compatible, so that information can be processed as described above without error. However, this does not mean that each system processes the data in a manner consistent with the intended meaning. For example, one system may use a table called “Actor” and another one called “Agent”. With syntactic interoperability, data from both tables may only be retrieved as distinct, even though they may have exactly the same meaning. To overcome this situation, semantic interoperability has to be added. | |

| | | |
|---|---|---|
| | The CRM relies on existing syntactic interoperability and is concerned only with adding semantic interoperability. | |
| semantic interoperability | Semantic interoperability means the capability of different information systems to communicate information consistent with the intended meaning. In more detail, the intended meaning encompasses<br>1. the data structure elements involved,<br>2. the terminology appearing as data and<br>3. the identifiers used in the data for factual items such as places, people, objects etc.<br>Obviously communication about data structure must be resolved first. In this case consistent communication means that data can be transferred between data structure elements with the same intended meaning or that data from elements with the same intended meaning can be merged. In practice, the different levels of generalization in different systems do not allow the achievement of this ideal. Therefore semantic interoperability is regarded as achieved if elements can be found that provide a reasonably close generalization for the transfer or merge. This problem is being studied theoretically as the query containment problem. The CRM is only concerned with semantic interoperability on the level of data structure elements. | |
| property quantifiers | We use the term property quantifiers for the declaration of the allowed number of instances of a certain property that an instance of its range or domain may have. These declarations are ontological, i.e. they refer to the nature of the real world described and not to our current knowledge. For example, each person has exactly one father, but collected knowledge may refer to none, one or many. | |
| universal | The fundamental ontological distinction between universals and particulars can be informally understood by considering their relationship with instantiation: particulars are entities that have no instances in any possible world; universals are entities that do | |

| | | |
|---|---|---|
| | have instances. Classes and properties (corresponding to predicates in a logical language) are usually considered to be universals. (after Gangemi et al. 2002, pp. 166-181). | |