

# AN INTRODUCTION TO MVCO

## MEDIA VALUE CHAIN ONTOLOGY

**Víctor Rodríguez, Jaime Delgado**

**DMAG (Distributed Multimedia Applications Group)**

**Universitat Politècnica de Catalunya**

**5th November 2008**



**DMAG**

DISTRIBUTED MULTIMEDIA APPLICATIONS GROUP



## TABLE OF CONTENTS

<b>Part I.</b>	<b>Model Description.....</b>	<b>3</b>
	The Intellectual Property Value Chain.....	6
	IP Entities.....	7
	Users and Roles .....	8
	Actions.....	9
	Permissions .....	10
<b>Part II.</b>	<b>Ontology Representation .....</b>	<b>11</b>
	Ontology Representation .....	11
	Reference of Classes.....	13
	Reference of Properties .....	24
	Annex I – Definitions .....	26
	Annex II – Matching Samples .....	29
	References.....	32

# Part I. Model Description

## INTRODUCTION

The Media Value Chain Ontology (MVCO) is an ontology for formalizing the representation of the Media Value Chain. It couples naturally with the MPEG-21 multimedia framework, and its standardization as Part 19 of this ISO/IEC standard is underway (on Committee Draft (CD) International Standard balloting at the editing time of this document). The proposal has been edited by members of the DMAG (Distributed Multimedia Applications Group) of the Universitat Politècnica de Catalunya (UPC), Jaime Delgado and Víctor Rodríguez, the Digital Spanish Author's Collecting Society (sDae), Marc Gauvin, and the Electronics and Telecommunications Research Institute (ETRI), Miran Choi.

Although hooked on the broader framework of the MPEG-21 standard, the MVCO ontology is complete and useful on its way. The MVCO is quite a relative small ontology, (less than 60 classes and 20 properties), simple to understand and accompanied by a forthcoming Java API. It has been coded as an OWL ontology.

## ONTOLOGY SCOPE

The MVCO represents the **Intellectual Property** (IP) along the Value Chain.

There are different kinds of objects of the Intellectual Property (we call them IP Entities) and different actions that are performed on them, what defines the different roles that users can play regarding these IP Entities. These elements, along the permissions to execute the actions, constitute the essence of the MVCO. The most important IP Entities, Actions and User roles are introduced here (the list is not exhaustive):

**IP Entities:** Work, Adaptation, Manifestation, Instance, Copy, Product

**User roles:** Creator, Adaptor, Instantiator, Producer, Distributor, EndUser

**Actions:** CreateWork, MakeAdaptation, MakeManifestation, MakeInstance, MakeCopy, Produce, Distribute, EndUserAction

Some examples of IP Entities for different domains are given in the following table:

	Music domain	Image domain	Video domain
<b>Work</b>	A Song	An artistic photo	A video sequence
<b>Manifestation</b>	The Score	The photo's original JPEG	The video's original AVI
<b>Instance</b>	First fixation (e.g. Studio record)	The photo's JPEG, possibly at a different resolution, size or cropping.	The video's AVI, possibly at a different resolution, size, etc.
<b>Product</b>	Marketable CD	The photo in a catalogue to be sold	Distributable product

Table 1. Examples in several domains

An Adaptation is to all respects a Work, but which is classified as derivative work of an original Work. Works (and Adaptations) are not physical objects but abstractions –a media file will never be classified as a Work. Manifestations are the expression of Works, while Instances are executions or renditions of the Work. Copies are reproductions of the Instance. A Product is the final IP Entity, and it carries or is subject to licensing terms.

## WHY A SINGLE VIEW OF THE IP VALUE CHAIN?

Any notion of Intellectual Property implies the existence of a minimum and necessary set of entities, roles and actions, each a corollary of the other. The objective and scope of the MVCO is to represent this minimum set in a machine readable fashion.

While there are clear differences in the legal treatment of IP between different jurisdictions, this does not mean that a common core between them does not exist. This is reflected by the numerous International World Intellectual Property Organization treaties such as the Berne convention where if it were not for a clear understanding of common terms such as work, adaptation, performance etc., it would not be possible for such a treaty to be assumed by 163 different national jurisdictions.

This common IP model lacks a computer representation and MVCO has the vocation of filling this gap.

The WIPO (**World Intellectual Property Organization**) is an intergovernmental organization with head-quarters in Geneva, Switzerland. It is one of the 16 specialized agencies of the United Nations system of organizations. WIPO is responsible for the promotion of the protection of intellectual property throughout the world through cooperation among States, and for the administration of various multilateral treaties dealing with the legal and administrative aspects of intellectual property. WIPO defines Intellectual Property as *the creations of the mind: inventions, literary and artistic works, and symbols, names, images, and designs used in commerce*. [1]

## SOME CONVENTIONS

**Special terms.** Terms beginning with a capital letter (like “Work”) are used according to definitions given in this document while terms referred to ontology elements are given with its namespace (e.g. `mvco:Work`).

**Namespace conventions.** MVCO includes an initial `rdf:RDF` component with some XML namespace declarations. Namespaces provide a means to unambiguously interpret identifiers and make the rest of the ontology presentation much more readable. It also contains the base URI for this document and the default namespace, which has been chosen as the URI of the document containing the ontology. The ontology URI is `http://purl.oclc.org/NET/mvco.owl`. This is a permanent URI deemed to provide a permanent reference regardless of the actual location of the MVCO ontology. The following namespace conventions have been adopted:

**Table 2 — Namespace prefixes**

Namespace prefix	Namespace
None (default namespace)	<code>http://purl.oclc.org/NET/mvco.owl#</code>
xsd	<code>http://www.w3.org/2001/XMLSchema#</code>
rdf	<code>http://www.w3.org/1999/02/22-rdf-syntax-ns#</code>
rdfs	<code>http://www.w3.org/2000/01/rdf-schema#</code>
owl	<code>http://www.w3.org/2002/07/owl#</code>
dc	<code>http://purl.org/dc/elements/1.1/</code>
dii	<code>urn:mpeg:mpeg21:2002:01-DII-NS#</code>
mx	<code>urn:mpeg:mpeg21:2003:01-REL-MX-NS#</code>
daml	<code>http://www.daml.org/2001/03/daml+oil#</code>

## THE INTELLECTUAL PROPERTY VALUE CHAIN

The origin of any IP Value Chain necessarily consists of the creation and subsequent manifestation of the original IP Entity referred to as Work, the rights over which are exclusively of the Work's author(s). This original IP Entity is subject to be used to create new dependent IP Entities leading to what is referred to as a value chain. Rights for the exploitation of IP Entities can be transferred along this value chain.

A Value Chain can be defined as: *a group of interacting Users, connecting (and including) Creators to End-Users with the purpose of Delivering Content to End-Users* (as given by the Digital Media Project (DMP<sup>1</sup>) and sketched in Figure 1 [2].

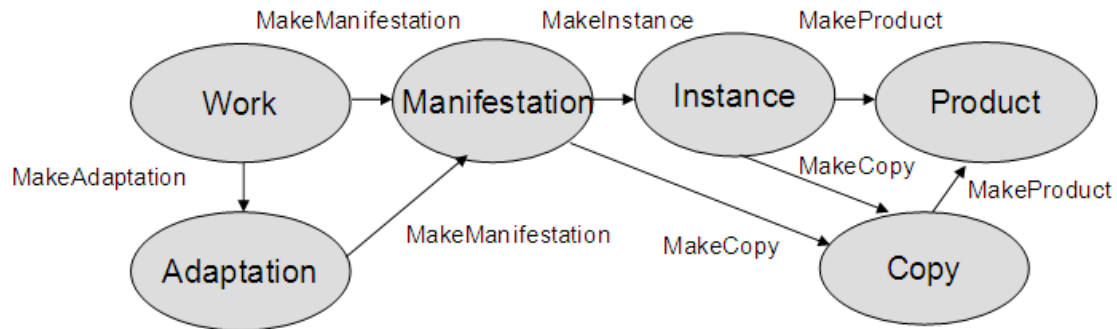


Figure 1. Value Chain as represented in DMP AD#2

<sup>1</sup> The DMP is a not-for-profit organisation with the mission to "promote continuing successful development, deployment and use of Digital Media that respect the rights of creators and rights holders to exploit their works, the wish of end users to fully enjoy the benefits of Digital Media and the interests of various value-chain players to provide products and services". (<http://www.dmpf.org>)

## IP ENTITIES

All entities subject to Intellectual Property are said to be IP Entities. The origin of an IP based value chain is the IP Entity “Work” and the end of the chain is the Product, these having as intermediate IP Entities Adaptation, Manifestation, Instance and corresponding Copies, as can be seen in Figure 1. They are defined as follows:

- Work. A creation that retains intellectual or artistic attributes independently of its Manifestations.
- Adaptation. A Work that is derived from another Work
- Manifestation. An object or event which is an expression of a Work
- Instance. An object or event which is an example of an Identified Manifestation (e.g. a File)
- Copy. A mechanical reproduction of analogue or digital representations of a given IP Entity. In the case of digital Copies the result is virtually identical while in the case of analogue Copies the results can vary considerably in quality.
- Product. A Content Item that adds value to IP Entities by including them with an appropriate Licence for the purpose of publishing.

Further specialization is given so each term can be more refined: For example, “Manifestation” can be either of an Adaptation or of a Work thus giving rise to the concepts “Adaptation-Manifestation” or “Work-Manifestation”

**Composite IP Entities.** On occasions, IP Entities are formed as the result of the joint composition of two or more IP Entities –of the same kind. We speak then about Composite IP Entities. A Composite IP Entity has as a rights owner that consists of the Collective made of the individual Creators of each component. In the case of an audiovisual work, the Director is also included as Creator of the composite work.

## USERS AND ROLES

The concept of User, sometimes referred as Agent, includes individuals, consumers, communities, organisations, corporations, consortia, governments or any other agent acting in the Value Chain.

The relationship between a User and a particular IP Entity type is specified through the concept of Role. The actions that a User takes on a given IP Entity determine the Role of that User with respect to the IP Entity in question. Thus, any given User may take on any number of Roles within a given Value Chain.

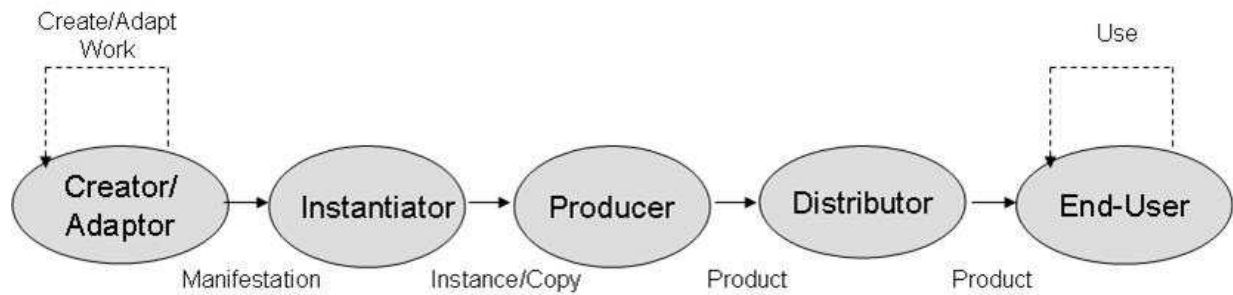


Figure 2 — User along the Value Chain

## ROLES

Figure 2 illustrates the core Roles, i.e. types of Users or actors that perform actions over IP Entities. The following core Roles are defined in MVCO:

- Creator. A User who generates a Work and makes its first Manifestation, also referred to as author.
- Adaptor. A User who produces an Adaptation.
- Instantiator. A User who produces an Instance.
- Producer. A User that makes Products.
- Distributor. A User who distributes a Product. Note: Any Service transferring Products along the value chain can be considered to be a Distributor.
- EndUser. A User in a Value-Chain who ultimately consumes content.

This set of core Roles can be extended to include further specialisations. For example, from Distributor, two specialized Roles could be distinguished: ContentAggregator who may operate in a B2B oriented model, and ContentProviders operating in a B2C oriented model. Both roles would be specializations of the generic Distributor Role and their definition would be wholly consistent with the MVCO Core model.

**Collective Users.** In practice, it happens that a given User sometimes acts as part of a Collective of Users, in the sense that several decision-enabled parties share jointly the responsibility for a given IP Entity of a work or hold collectively certain rights. The MVCO model explicitly acknowledges this fact.

**Content Handlers.** On occasions, Users delegate tasks in other Users called Content Handlers for which they are ultimately responsible. This Role has no rights associated with it other than those conferred solely by virtue of its “surrogate” function under the auspices of the User under which it operates on Content. The MVCO model explicitly acknowledges this figure.



## ACTIONS

Actions are the process of doing something over IP Entities. Actions can be applied over the IP Entities themselves or over their representations including both analogue and digital.

The result of some Actions may imply the creation of another IP Entity (for example, a MakeAdaptation Action generates a new IP Entity of the kind Adaptation) while others do not as in the case of Render.

Each action can be exercised over only one kind of IP Entity, and it can only be performed by one Role. The User who performs an Action has to be the rights owner of the IP Entity. Further Permissions may be required by virtue of the Creator's moral rights.

The main actions are listed below:

- CreateWork. The action of creating a Work.
- MakeAdaptation. The action of making an Adaptation. (Note: Adapt and MakeAdaptation are not equivalent. The result of MakeAdaptation has an IP Entity (Adaptation) as a result, while Adapt represents a transient change of the Resource
- MakeManifestation. The Action of making a Manifestation
- MakeInstance. The Action of making an Instance from a Manifestation.
- MakeCopy. The Function by which Device A Stores Content in Device B, preserving the original Content in Device A. Similar to MechanicalCopy.
- Produce. The Function of making Products
- Distribute. The Right to sell, rent and lend.
- PublicCommunication. The action of publicly displaying/performing, e.g. live performance, radio, television, internet. A specialization of PublicCommunication with special interest is Broadcast (*The Function that Delivers Content to a Device in a point-to-multipoint modality*).
- Synchronise. Concurrent performance/display of two distinct Works or Adaptation Instances each for a different sense e.g. text and audio or video and song. Synchronization implies the simultaneous rendition of two IP Entities requiring the additional Permission of the Creator.
- EndUserAction. The Action performed by an EndUser

## PERMISSIONS

Transfers of Rights are represented in Permissions. A Permission relates an IP Entity with the transmitted right, the original rights owner and the new rights owner. A Permission may require the prior satisfaction of the conditions. Requirements in Permissions are expressed as Facts, which are simply defined as positive propositions with a binary truth value. A prohibition is thus expressed as the negation of a particular Fact.

**Copyright exceptions.** Some rights can be invoked if certain conditions are met. For example, complete quotes are allowed for scientific purposes. The MVCO Ontology provides mechanisms for specifying such copyright exceptions, although the exceptions themselves are not specified. Permissions from one User to another are not needed to invoke a CopyrightException, CopyrightExceptions are given based on the existence of the corresponding CopyrightExceptionFact.

# Part II. Ontology Representation

## ONTOLOGY REPRESENTATION

The core model presented so far has been represented as a Web Ontology Language (OWL) ontology. The Web Ontology Language is a specification from the World Wide Web Consortium (W3C) based on the Resource Description Framework (RDF), keystone of the Semantic Web. Its coding can be as simple as a single XML file which can be universally accessed. By using this model supported by existing technology standards, organizations can create end-to-end applications based on the MVCO Core Model. The MVCO Core Model is fully described by giving a set of classes and properties of those classes.

## ONTOLOGY GENERAL FEATURES

OWL ontologies can be categorised into three species or sub-languages: OWL-Lite, OWL-DL and OWL-Full, with increasing level of expressivity. OWL-DL is based on *Description Logic*, a decidable fragment of First Order Logic which grants automated reasoning in a finite time. The MVCO is consistent and uses the DL subset of the OWL language, granting thus its decidability and its usability in practical applications. MVCO expressivity level is *SIF(D)*.

## GRAPHICAL REPRESENTATION

**Actions, Users and IP Entities.** Users have Roles associated to them (Creator, Adaptor, Instantiator, Producer...) that attribute to them rights over Actions that can be exercised on the specific corresponding IP Entities. Figure 3 shows these relations.



Figure 3 — Relations between Actions, Users and IP Entities

Figure 4 shows the main relations regarding the Actions.

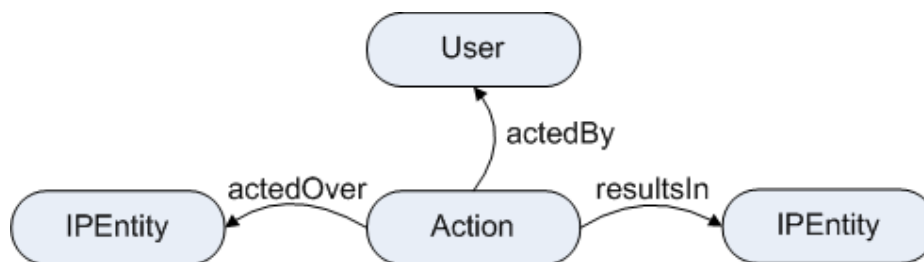


Figure 4. Main relations around Action

Figure 5 shows the main relations involved in the permission mechanism. Permissions are given to execute Actions over existing IPEntities (excepting for the mvco:CreateWork action). The issuers of these Permissions must be the right owners of the IPEntities who suffer the action. In addition, some Actions require the extra consent of the Creator of the original Work, which can be expressed with the mvco:rightGivenBy relation.

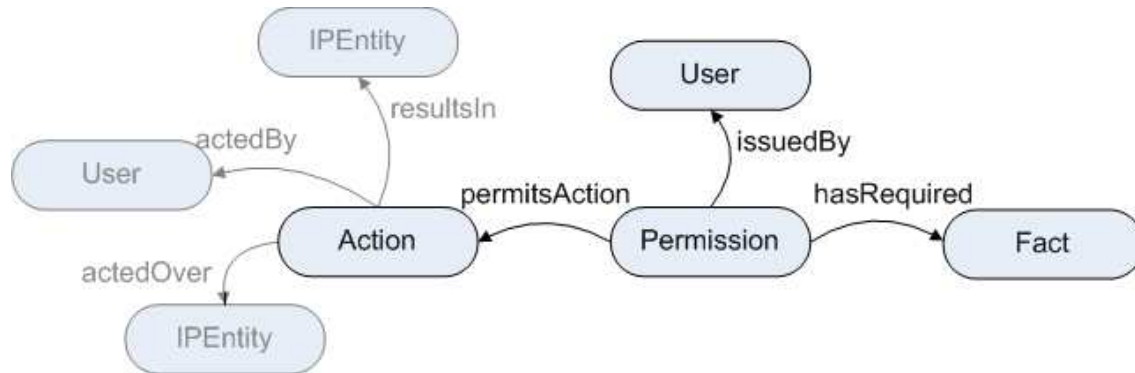
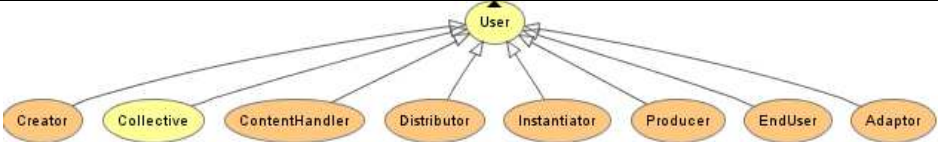


Figure 5. Main relations around rights ownership and transmission

## REFERENCE OF CLASSES

## USER-RELATED CLASSES

Class	<b>User</b>	
rdfs:comment	Any person or legal entity in a Value-Chain connecting (and including) Creator and EndUser.	
rdfs:subClassOf	owl:Thing	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\subseteq \forall \text{ acts} . \text{Action}$	A User may only act Actions
	$\subseteq \forall \text{ belongsTo} . \text{Collective}$	A User may only belong to Collectives
in-domain-of	isRightsOwnerOf (IPEntity) belongsTo (Collective) acts (Action) socialTag (String)	
in-range-of	actedBy hasRightsOwner issuedBy actOnBehalfOf rightGivenBy	
Known subclasses		
Individuals	Anonymous	

Class	<b>Collective</b>
rdfs:comment	Set of two or more Users.
rdfs:subClassOf	mvco:User
Restrictions	None

Class	<b>ContentHandler</b>	
rdfs:comment	A User who is appointed to act on Content on behalf of another User and within the scope and responsibility of that second User's rights.	
rdfs:subClassOf	mvco:User	
Domain	actOnBehalfOf (User)	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv \exists \text{ actOnBehalfOf} . \text{User}$	A ContentHandler is any User who acts on behalf a User.

Class	<b>Creator</b>	
rdfs:comment	A mvco:User who generates a mvco:Work and makes its first mvco:Manifestation, also referred to as author	
rdfs:subClassOf	mvco:User	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv \exists \text{ acts} . \text{CreateWork}$	A Creator is a User who has created a Work

Class	<b>Adaptor</b>	
rdfs:comment	A User who produces an Adaptation	
rdfs:subClassOf	mvco:User	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv \exists \text{ acts . MakeAdaptation}$	An Adaptor is a User who performs a MakeAdaptation


Class	<b>Instantiator</b>	
rdfs:comment	A User who interprets a Manifestation of a Work making an Instance	
rdfs:subClassOf	mvco:User	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv \exists \text{ acts . MakeInstance}$	An Instantiator is a User who has executed a MakeInstance

Class	<b>Producer</b>	
rdfs:comment	A User who produces a Product from one or more Instances.	
rdfs:subClassOf	mvco:User	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv \exists \text{ acts . Produce}$	A Producer is a User who has executed a Produce Action.

Class	<b>Distributor</b>	
rdfs:comment	A User who distributes a Product	
rdfs:subClassOf	mvco:User	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv \exists \text{ acts . Distribute}$	A Distributor is a User who has executed a Distribute Action


Class	<b>EndUser</b>	
rdfs:comment	A User in a Value-Chain who ultimately consumes Content.	
rdfs:subClassOf	mvco:User	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv \exists \text{ acts . EndUserAction}$	An EndUser is a User who has executed any of the EndUserActions.


## IP ENTITIES AND USEDATA

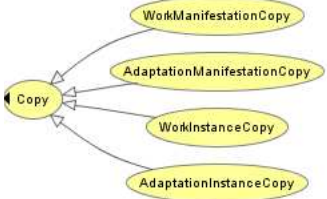
Class	<b>IPEntity</b>	
rdfs:comment	Types of IP Represented as Content: Work, Adaptation, Manifestation, Instance...	
rdfs:subClassOf	owl:Thing	
in-domain-of	hasRightsOwner (User) isMadeUpOf (IPEntity) resultedFrom (Action) dii:RelatedIdentifier (URI) socialTag (String)	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\subseteq \forall \text{ hasRightsOwner User}$ $\subseteq \text{ hasRightsOwner exactly } 1$	An IP Entity has one and only one User as rights owner. Note: anonymous Users are allowed by using the defined mvco:Anonymous individual.
	$\subseteq \forall \text{ isMadeUpOf IPEntity}$	A Composite IP Entity may be made of several IPEntities.
	$\subseteq \forall \text{ resultedFrom Action}$	IPEntities are created as results of Actions
Known subclasses	 <pre> graph BT     Product --&gt; IPEntity     Work --&gt; IPEntity     Adaptation --&gt; IPEntity     Manifestation --&gt; IPEntity     Instance --&gt; IPEntity     Copy --&gt; IPEntity </pre>	

Class	<b>Work</b>	
rdfs:comment	A creation that retains intellectual or artistic attributes independently of its Manifestations	
rdfs:subClassOf	mvco:IPEntity	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\subseteq \forall \text{ resultedFrom CreateWork}$	A Work is only the result of a CreateWork Action

Class	<b>Adaptation</b>	
rdfs:comment	A Work that is derived from another Work	
rdfs:subClassOf	mvco:IPEntity	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\subseteq \forall \text{ resultedFrom MakeAdaptation U Synchronise}$	An Adaptation derives only from the execution of a MakeAdaptation or a Synchronise Action

Class	<b>Manifestation</b>	
rdfs:comment	An object or event which is an expression of a Work.	
rdfs:subClassOf	mvco:IPEntity	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\subseteq \forall \text{ resultedFrom MakeManifestation}$	A Manifestation derives from the execution of a MakeManifestation Action
Known subclasses	 <pre> graph BT     WorkManifestation --&gt; Manifestation     AdaptationManifestation --&gt; Manifestation </pre>	

Class	<b>Instance</b>	
rdfs:comment	An object or event which is an example of an Identified Manifestation (e.g. a File).	
rdfs:subClassOf	mvco:IPEntity	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\subseteq \forall \text{ resultedFrom MakeInstance}$	An Instance derives from the execution of a MakeInstance Action
Known subclasses		

Class	<b>Copy</b>	
rdfs:comment	A mechanical reproduction of analogue or digital representations of a given IP Entity. In the case of digital Copies the result is virtually identical while in the case of analogue Copies the results can vary considerably in quality.	
rdfs:subClassOf	mvco:IPEntity	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\subseteq \forall \text{ resultedFrom MakeCopy}$	A Copy derives from the execution of a MakeCopy Action
Known subclasses		

Class	<b>Product</b>	
rdfs:comment	A Content Item that adds value to IP Entities by including them with an appropriate Licence for the purpose of Publishing	
rdfs:subClassOf	mvco:IPEntity	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\subseteq \exists \text{ resultedFrom. Produce}$	A Product derives from the execution of a Produce Action

Class	<b>UseData</b>	
rdfs:comment	Data documenting the Actions performed by a User on a content item and the associated context	
rdfs:subClassOf	owl:Thing	

Class	<b>AdaptationManifestation</b>	
rdfs:comment	An object or event which is an expression of an Adaptation	
rdfs:subClassOf	mvco:Manifestation	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{ resultedFrom MakeAdaptationManifestation} \cup \text{Manifestation})$	An AdaptationManifestation is every Manifestation resulting from executing a MakeAdaptationManifestation Action.



Class	<b>WorkManifestation</b>	
rdfs:comment	An object or event which is an expression of a Manifestation of a Work	
rdfs:subClassOf	mvco:Manifestation	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{resultedFrom MakeWorkManifestation} \sqcap \text{Manifestation})$	A WorkManifestation is the Manifestation resulting of executing a MakeWorkManifestation action.

Class	<b>WorkInstance</b>	
rdfs:comment	An object or event which is an example of an Identified Manifestation of a Work (e.g. a File)	
rdfs:subClassOf	mvco:Instance	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{resultedFrom MakeWorkInstance} \sqcap \text{Instance})$	A WorkInstance is every Instance coming from a WorkManifestation.

Class	<b>AdaptationInstance</b>	
rdfs:comment	An object or event which is an example of an Identified Adaptation Manifestation (e.g. a File)	
rdfs:subClassOf	mvco:Instance	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{resultedFrom MakeAdaptationInstance} \sqcap \text{Instance})$	An AdaptationInstance is every Instance coming from a AdaptationManifestation.

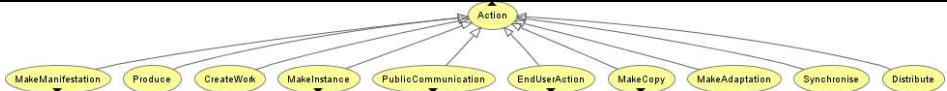
Class	<b>AdaptationInstanceCopy</b>	
rdfs:comment	A copy of an AdaptationInstance	
rdfs:subClassOf	mvco:Copy	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\sqsubseteq \exists \text{resultedFrom MakeAdaptationInstanceCopy}$	An AdaptationInstanceCopy is the result of executing a MakeAdaptationInstanceCopy Action.

Class	<b>AdaptationManifestationCopy</b>	
rdfs:comment	A copy of an AdaptationManifestation	
rdfs:subClassOf	mvco:Copy	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{resultedFrom MakeAdaptationManifestationCopy} \sqcap \text{Copy})$	An AdaptationInstanceCopy is every copy resulting of executing a MakeAdaptationManifestationCopy.

Class	<b>WorkInstanceCopy</b>	
rdfs:comment	A copy of WorkInstance	
rdfs:subClassOf	mvco:Copy	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{resultedFrom MakeWorkInstanceCopy} \sqcap \text{Copy})$	A WorkInstanceCopy is every copy resulting of executing a MakeWorkInstanceCopy.

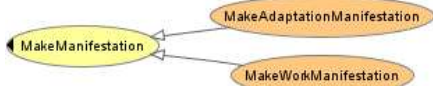
Class	<b>WorkManifestationCopy</b>	
rdfs:comment	A copy of an WorkManifestation	
rdfs:subClassOf	mvco:Copy	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{resultedFrom } \text{MakeWorkManifestationCopy}$ $\cap \text{Copy})$	A WorkManifestationCopy is every copy resulting of executing a MakeWorkManifestationCopy.

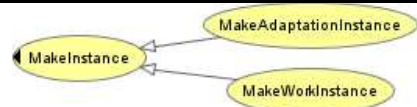
## ACTIONS AND RIGHTS

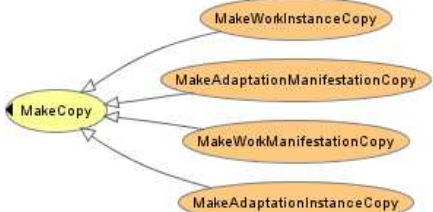
Class	<b>Action</b>	
rdfs:comment	The exercise of a Right.	
rdfs:subClassOf	owl:Thing	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\subseteq \exists \text{ actedBy User}$	Actions are acted by one and only one User.
	$\subseteq \text{actedBy exactly } 1$	
	$\subseteq \exists \text{ impliesAlso Action}$	Actions may imply other Actions
	$\subseteq \exists \text{ rightsGivenBy User}$	Only Users can give rights to execute Actions
in-domain-of	actedOver (IPEntity) actedBy (User) rightGivenBy (User) resultsIn (IPEntity) impliesAlso (Action)	
Known subclasses		

Class	<b>CreateWork</b>	
rdfs:comment	The Action of creating a Work.	
rdfs:subClassOf	mvco:Action	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\subseteq (\neg (\exists \text{actedOver IPEntity}))$	Cannot be executed over any IPEntity
	$\subseteq (\exists \text{resultsIn Work})$	The result of creating a work is at least one Work (and at most one, because <i>resultsIn</i> is a functional relation)
	$\subseteq \text{rightGivenBy exactly } 0$	No one can give the right to create a Work

Class	<b>MakeAdaptation</b>	
rdfs:comment	The Action of making an Adaptation	
rdfs:subClassOf	mvco:Action	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\subseteq (\exists \text{ actedOver Work})$	An Adaptation is made over some Work
	$\subseteq (\exists \text{ rightGivenBy Creator})$	Only Creators give the right to make an Adaptation
	$\subseteq (\exists \text{resultsIn Adaptation})$	The act of making an adaptation results in one Adaptation.

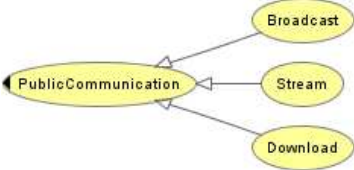
Class	<b>MakeManifestation</b>	
rdfs:comment	The action of making a Manifestation.	
rdfs:subClassOf	mvco:Action	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\sqsubseteq (\exists \text{actedOver} \text{ (Work U Adaptation)})$	A Manifestation has to be done over some Work or Adaptation
	$\sqsubseteq (\exists \text{resultsIn} \text{ . Manifestation})$	The act of making a manifestation results in one Manifestation.
Known subclasses	 <pre> graph TD     MM([MakeManifestation]) --&gt; MAM([MakeAdaptationManifestation])     MM --&gt; MW([MakeWorkManifestation]) </pre>	

Class	<b>MakeInstance</b>	
rdfs:comment	The Action of making an Instance from a Manifestation.	
rdfs:subClassOf	mvco:Action	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\sqsubseteq (\exists \text{actedOver} \text{ . Manifestation})$	Instances are created from Manifestations
	$\sqsubseteq (\exists \text{resultsIn} \text{ . Instance})$	The act of making an Instance results in one Instance.
Known subclasses	 <pre> graph TD     MI([MakeInstance]) --&gt; MAI([MakeAdaptationInstance])     MI --&gt; WI([MakeWorkInstance]) </pre>	

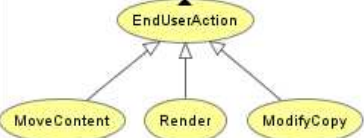
Class	<b>MakeCopy</b>	
rdfs:comment	The Action of making a Copy	
rdfs:subClassOf	mvco:Action	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\sqsubseteq (\exists \text{actedOver} \text{ . Instance})$	Copies are created from Instances
	$\sqsubseteq (\exists \text{resultsIn} \text{ . Copy})$	The act of making a Copy results in Copy.
Known subclasses	 <pre> graph TD     MC([MakeCopy]) --&gt; MWIC([MakeWorkInstanceCopy])     MC --&gt; MAMC([MakeAdaptationManifestationCopy])     MC --&gt; MWMC([MakeWorkManifestationCopy])     MC --&gt; MAIC([MakeAdaptationInstanceCopy]) </pre>	

Class	<b>Produce</b>	
rdfs:comment	The function of making Products	
rdfs:subClassOf	mvco:Action	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\sqsubseteq (\exists \text{actedOver} \text{ . Copy U Instance})$	Products are created from Copies or Instances
	$\sqsubseteq (\exists \text{resultsIn} \text{ . Product})$ $\sqsubseteq (\text{resultsIn exactly } 1)$	The act of making a Product results in exactly 1 Product.
	$\sqsubseteq (\exists \text{rightsGivenBy Creator})$	Producing requires the additional permission of the Creator.

Class	<b>Synchronise</b>	
rdfs:comment	The Action of concurrently perform/display two distinct IP Entities each for a different human sense e.g. text and audio or video and song	
rdfs:subClassOf	mvco:Action	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\sqsubseteq (\exists \text{actedOver} . \text{Work})$	Synchronization is made over Works
	$\sqsubseteq (\exists \text{rightGivenBy} . \text{Creator})$	Making a Synchronization requires the additional permission given by the Creator.
	$\sqsubseteq (\forall \text{impliesAlso} . (\text{Render U ModifyCopy}))$	Any User who has the right to Synchronise, can also invoke the right to Render or Modify a Copy.

Class	<b>PublicCommunication</b>	
rdfs:comment	The Function of publicly displaying/performing, e.g. live performance, radio, television, internet streaming, multicast of Instances and Manifestations, and download	
rdfs:subClassOf	mvco:Action	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\sqsubseteq (\exists \text{actedOver} . \text{Copy U Product})$	Public Communication can be performed on Copies or Products
	$\sqsubseteq (\exists \text{rightGivenBy} . \text{Creator})$	Making PublicCommunication requires the additional permission given by the Creator.
	$\sqsubseteq (\forall \text{impliesHaving} . \text{Render})$	Any User who has the right to make a PublicCommunication, can indeed invoke the right to Render.
Known subclasses	 <pre> graph BT     Broadcast --&gt; PublicCommunication     Stream --&gt; PublicCommunication     Download --&gt; PublicCommunication </pre>	

Class	<b>Distribute</b>	
rdfs:comment	The Function of selling, renting and lending	
rdfs:subClassOf	mvco:Action	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\sqsubseteq (\exists \text{actedOver} . \text{Product})$	There must be a Product to distribute
	$\sqsubseteq (\text{resultsIn exactly } 0)$	The act of Distribute cannot generate any additional IP Entity
	$\sqsubseteq (\exists \text{rightsGivenBy} . \text{Creator Producer})$	Distributing requires the additional permission of the Creator.

Class	<b>EndUserAction</b>	
rdfs:comment	Action performed by an EndUser	
rdfs:subClassOf	mvco:Action	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\sqsubseteq (\exists \text{actedOver} . \text{Product})$	End Users only deal with Products
	$\sqsubseteq (\text{resultsIn exactly } 0)$	EndUserActions cannot generate any additional IP Entity
Known subclasses	 <pre> graph BT     MoveContent --&gt; EndUserAction     Render --&gt; EndUserAction     ModifyCopy --&gt; EndUserAction </pre>	

Class	<b>Broadcast</b>
rdfs:comment	The Function that Delivers Content to a Device in a point-to-multipoint modality
rdfs:subClassOf	mvco:PublicCommunication
Restrictions	none

Class	<b>Download</b>
rdfs:comment	The Action of transferring a file or program from a central computer to a smaller computer or to a computer at a remote location
rdfs:subClassOf	mvco:PublicCommunication
Restrictions	none

Class	<b>Stream</b>
rdfs:comment	The Function of Delivering Content to a Device where the transferred Content is Processed for Rendering only and not Stored
rdfs:subClassOf	mvco:PublicCommunication
Restrictions	none

Class	<b>MakeAdaptationInstanceCopy</b>	
rdfs:comment	The Action of making an AdaptationInstanceCopy	
rdfs:subClassOf	mvco:MakeCopy	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{ actedOver } \text{AdaptationInstance} \cap \text{MakeCopy})$	MakeCopy executed over an AdaptationInstance.
	$\subseteq (\exists \text{ resultsIn } \text{AdaptationInstanceCopy})$	The result must be an AdaptationInstanceCopy

Class	<b>MakeAdaptationManifestationCopy</b>	
rdfs:comment	The Action of making an AdaptationManifestationCopy	
rdfs:subClassOf	mvco:MakeCopy	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{ actedOver } \text{AdaptationManifestation} \cap \text{MakeCopy})$	MakeCopy executed over an AdaptationManifestation.
	$\subseteq (\exists \text{ resultsIn } \text{AdaptationManifestationCopy})$	The result must be an AdaptationManifestationCopy

Class	<b>MakeWorkInstanceCopy</b>	
rdfs:comment	The Action of making an WorkInstanceCopy	
rdfs:subClassOf	mvco:MakeCopy	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{ actedOver } \text{WorkInstance} \cap \text{MakeCopy})$	MakeCopy executed over a WorkInstance.
	$\subseteq (\exists \text{ resultsIn } \text{WorkInstanceCopy})$	The result must be an WorkInstanceCopy


Class	<b>MakeWorkManifestationCopy</b>	
rdfs:comment	The Action of making a WorkManifestationCopy	
rdfs:subClassOf	mvco:MakeCopy	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{ actedOver } \text{WorkManifestation} \sqcap \text{MakeCopy})$	MakeCopy executed over a WorkManifestation.
	$\sqsubseteq (\exists \text{ resultsIn } \text{WorkManifestationCopy})$	The result must be an WorkManifestationCopy

Class	<b>MakeAdaptationInstance</b>	
rdfs:comment	The Action of making an Instance from an AdaptationManifestation	
rdfs:subClassOf	mvco:MakeInstance	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{ actedOver } \text{AdaptationManifestation} \sqcap \text{MakeInstance})$	IMakeInstance executed over one AdaptationManifestation.
	$\sqsubseteq (\exists \text{ resultsIn } \text{AdaptationInstance})$	The result must be an AdaptationInstance

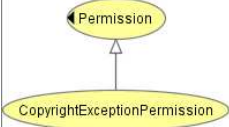
Class	<b>MakeWorkInstance</b>	
rdfs:comment	The Action of making an Instance from a WorkManifestation	
rdfs:subClassOf	mvco:MakeInstance	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{ actedOver } \text{WorkManifestation} \sqcap \text{MakeInstance})$	It is a MakeInstance executed over one WorkManifestation.
	$\sqsubseteq (\exists \text{ resultsIn } \text{WorkInstance})$	The result must be an WorkInstance

Class	<b>MakeWorkManifestation</b>	
rdfs:comment	The Action of making a Manifestation from Work.	
rdfs:subClassOf	mvco:MakeManifestation	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\exists \text{ actedOver } \text{Work} \sqcap \text{MakeManifestation})$	MakeManifestation executed over a Work.
	$\sqsubseteq (\exists \text{ resultsIn } \text{WorkManifestation})$	The result must be a WorkManifestation

## PERMISSION AND RELATED CLASSES

Class	<b>Fact</b>	
rdfs:comment	Positive proposition.	
rdfs:subClassOf	owl:Thing	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\equiv (\text{truth exactly } 1)$	Any individual belongs to the mvco:Fact class as long as its truth or falsehood can be asserted.
Known subclasses		

Class	<b>CopyrightExceptionFact</b>
rdfs:comment	Fact related to the invocation of a CopyrightException.
rdfs:subClassOf	mvco:Fact

Class	<b>Permission</b>	
rdfs:comment	Authorisation from one RightsOwner to one or more Users to perform one or more Actions on a given IPEntity. (Note: This includes the defacto application of CopyrightExceptions via existence of corresponding Facts.)	
rdfs:subClassOf	owl:Thing	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\sqsubseteq \exists \text{permitsAction Action}$	Permissions permit at least one Action and only Actions.
	$\sqsubseteq \forall \text{permitsAction Action}$	
	$\sqsubseteq \exists \text{issuedBy User}$	Permissions are issued by exactly one User
	$\sqsubseteq \text{issuedBy exactly 1}$	
	$\sqsubseteq \exists \text{hasRequired Fact}$	Permissions have as requirements only Facts
in-domain-of	hasRequired (Fact) issuedBy (User) permitsAction (Action)	
Known subclasses		

Class	<b>CopyrightExceptionPermission</b>	
rdfs:comment	Permission to invoke one right exceptionally.	
rdfs:subClassOf	mvco:Permission	
Restrictions	<b>Expression</b>	<b>Meaning</b>
	$\sqsubseteq (\exists \text{hasRequired . CopyrightExceptionFact})$	Permissions must have at least one CopyrightExceptionFact as requirement

## REFERENCE OF PROPERTIES

Two kinds of properties can be defined in an OWL Ontology:

- Object Properties, which link two class individuals.
- Datatype Properties, which attributes a literal value to a class individual.

A *property axiom* defines characteristics of a property. In its simplest form, a property axiom just defines the existence of a property but they can also define additional characteristics of properties. OWL supports the following constructs for property axioms:

- *RDF Schema constructs*: `rdfs:subPropertyOf`, `rdfs:domain` and `rdfs:range`. In MVCO there is no hierarchy of properties, but *domains* and *ranges* are given for each property. By giving a *domain* we assert that the property only applies to instances of the given class (i.e. `mvco:Action` is the domain of the property `mvco:resultsIn` because it can only be applied on `mvco:Action` individuals). By giving a *range* we assert that the property only assumes values that are instances of the given class (e.g. executing a `mvco:Action`, results in an `mvco:IPEntity`)
- *Relations to other properties*: `owl:equivalentProperty` and `owl:inverseOf`. MVCO defines 4 pairs of property - inverse property.
- *Global cardinality constraints*: `owl:FunctionalProperty`, `owl:InverseFunctionalProperty`. MVCO defines some properties as functionals, (properties that can have only one (unique) value for each instance).
- *Logical property characteristics*: `owl:SymmetricProperty` and `owl:TransitiveProperty`. Two properties are defined to be transitive in the MVCO ontology, and none to be symmetric.

Table 3 lists the object properties in the MVCO ontology, including their features (T stands for transitive, F for functional).

Table 3 — Object Properties in the MVCO Ontology.

Relation	Definition	Domain	Range	P	Inverse
<code>actedOver</code>	Specifies which IP Entity is the object of the Action	Action	IPEntity		
<code>resultsIn</code>	Declares which IP Entity arises as a result of the execution of an Action. It is a functional relation.	Action	IPEntity	F	<code>resultedFrom</code>
<code>actedBy</code>	Declares the User who has executed the Action	Action	User	F	
<code>Acts</code>	Performance of an Action by a User	User	Action		
<code>hasRequired</code>	For a Permission to be valid, the Fact has to hold	Permission	Fact		
<code>impliesAlso</code>	Explicit link between one and any other number of Actions	Action	Action		
<code>rightGivenBy</code>	Declares from which User rights are given	Action	User		
<code>permitsAction</code>	Relation used to express the Actions that are allowed to be performed.	Permission	Action		
<code>actOnBehalfOf</code>	Relates a ContentHandler with the User under the auspices of which the ContentHandler operates.	Content Handler	User	T	



issuedBy	Declares who has issued a Permission	Permission	User		
hasRightsOwner	Defines the owner of the Rights over an IP Entity.	IPEntity	User	F	<i>isRightsOwnerOf</i>
isMadeUpOf	Relates a composite IP Entity with its constituent IP Entities	IPEntity	IPEntity		
belongsTo	Relates a User with a Collective	User	Collective	T	

Some datatype properties also have been defined in the MVCO ontology. They are listed in Table 4, along with its definition, its domain, and its type (the kind of value it can assume, which is one of the XML Schema types).

Table 4 — Datatype properties in the MVCO Ontology

Property	Domain	type	Definition
dii:RelatedIdentifier	owl:Thing	xsd:string	It allows the identification information that is related to a Digital Item (or parts thereof).
isDigital	IPEntity	xsd:boolean	Specifies whether the IP Entity is digital or not
hasSocialTag	IPEntity U User	xsd:String	Specifies an entry point for social tags to be added
isTrue	Fact	xsd:boolean	Truth

## ANNEX I – DEFINITIONS

The definitions given in this Annex are mainly taken from Approved Document 6 of the Interoperable DRM Platform version 3.0 of DMP [3]. These are given here as a help to interpret the requirements in a consistent fashion. The definition of some other “REL” terms can be found in ISO/IEC 21000-5 (MPEG-21 Rights Expression Language) and its Amendments.

Action	The exercise of a Right
Adaptation	A Work that is derived from another Work
AdaptationInstance	An object or event which is an example of an Identified Adaptation Manifestation (e.g. a File)
AdaptationInstanceCopy	A copy of an AdaptationInstance
AdaptationManifestation	An object or event which is an expression of an Adaptation
AdaptationManifestationCopy	A copy of an AdaptationManifestation
Adaptor	A User who produces an Adaptation
Broadcast	The function that delivers Content to a Device in a point-to-multipoint modality
Collective	Set of two or more Users.
Content	A defined structured of Content Elements e.g. Digital Item
Content Elements	Any of the following types of data: Resource, metadata, nested Content, license, IPMP data, IPMP tools and Use Data
ContentHandler	User who is appointed to take Action on Content on behalf and within the scope and responsibility of another User’s rights.
Copy (verb)	The function by which Device A Stores content in Device B, preserving the original content in Device A
Copy (noun)	A mechanical reproduction of analogue or digital representation of a given IP Entity. In the case of a digital Copy the result is virtually identical while in the case of an analogue Copy the result can vary considerably in quality.
CopyrightException	Permission to invoke a right under exceptional circumstance (e.g. when a particular Fact is true).
CreateWork	The action of creating a Work without any previous IP Entity
Creator	A User who generates a Work and makes its first Manifestation, also referred to as author
Distribute	Sell, rent or lend
Distributor	A User who distributes a Product
Device	A combination of hardware and/or software that allows a User to execute functions over Content and/or IP Entities
Download	The Action of transferring a file or program from a central computer to a smaller computer or to a computer at a remote location
End-User	A User in a Value-Chain who ultimately consumes content
End-UserAction	Action performed by an EndUser
Fact	Positive proposition.
Identify	The function of assigning a unique signifier that establishes the identity of entities, Devices, Content and Content Elements
Instance	An object or event which is an example of an Identified Manifestation (e.g. a File)
Instantiator	A User who produces an Instance

Intellectual Property (IP)	Any identifiable product of the mind attributable to any person(s) or legal entitie(s) that can be represented or communicated physically and protectable by copyright or similar laws.
IP Entities	Types of IP Represented as content: Work, Adaptation, Manifestation, Instance
MakeAdaptation	The action of making an Adaptation
MakeAdaptationManifestationCopy	The action of making a ManifestationCopy
MakeAdaptationInstanceCopy	The action of making an Adaptation InstanceCopy
MakeAdaptationInstance	The action of making an Instance from an Adaptation
MakeAdaptationManifestation	The action of making an Adaptation Manifestation.
MakeCopy	The right to make a copy of a Representation
MakeInstance	The action of making an Instance from a Manifestation.
MakeManifestation	The action of making a Manifestation.
MakeWorkInstanceCopy	The action of making a Work InstanceCopy
MakeWorkManifestationCopy	The action of making a Work ManifestationCopy
MakeWorkInstance	The action of making an Instance from a Work Manifestation.
MakeWorkManifestation	The action of making a Manifestation from Work.
Manifestation	An object or event which is an expression of a Work
Permission	Authorisation from one Rights owner to one or more Users to perform one or more Actions on a given IPEntity.
Private Copy	The function of storing content and hold it private for non commercial purposes
Produce	The function of making Products
Producer	A User that makes Products
Product	A content item that adds value to IP Entities by including them with an appropriate licence for the purpose of publishing
Public Communication	The function of publicly displaying/performing, e.g. live performance, radio, television, internet streaming, multicast of Instances and Manifestations, and download
Render	The function of converting a Resource to a human-perceivable form
Represent	Expressing information in a form that can be processed by either a digital or analogue Device
Resource	Data, whose Representation is not specified by DMP (e.g. an MP3 file or an executable code), that can be processed by a Device
Right	A consequence of ownership of Intellectual Property yielding the ability of performing one or more functions over the Intellectual Property
Role	A defined set of actions and corresponding conditions attributed to and required of a User
Synchronise	The function of concurrently performing/displaying two or more distinct IP Entities each for a different human sense e.g. text and audio or video and song
Use Data	Data documenting the functions performed by a Device on a content item and the associated context
User	Any person or legal entity in a Value-Chain connecting (and including) Creator and End-User.
Value-Chain	A group of interacting Users, connecting (and including) Creators to End-Users
Work	A creation that retains intellectual or artistic attributes independently of its Manifestations
WorkInstance	An object or event which is an example of an Identified Manifestation of a Work (e.g. a file)
WorkInstanceCopy	A copy of a WorkInstance
WorkManifestation	An object or event which is an expression of a Manifestation of a Work

WorkManifestationCopy	A copy of a WorkManifestation
-----------------------	-------------------------------

## ANNEX II – MATCHING SAMPLES

The Semantic Web foresees and encourages interoperation between different semantic models.

The MVCO Ontology can be extended as any other ontology by adding new derived terms, by adding new relations, etc.

The MVCO can also be used in conjunction with other ontologies, provided that a matching of some key terms is adequate. In this section some examples follow showing these possible alignments.

### ALIGNMENT WITH MPEG-21 REL RIGHTS

MPEG-21 REL defines 14 rights in its Multimedia Extension (whose namespace prefix is usually denoted mx) [4]. An alignment of terms could be done as follows:

Table 5. Alignment between MVCO and MPEG-21 REL terms

<b>MPEG-21 REL – XML Schemata</b>	<b>Relation</b>	<b>MVCO</b>
mx:adapt	subclass Of	mvco:ModifyCopy
mx:delete	subclass Of	mvco:MoveContent
mx:diminish	subclass Of	mvco:ModifyCopy
mx:embed	subclass Of	mvco:Synchronize
mx:enhance	subclass Of	mvco:ModifyCopy
mx:enlarge	subclass Of	mvco:ModifyCopy
mx:execute	subclass Of	mvco:Render
mx:install	subclass Of	mvco:MoveContent
mx:modify	subclass Of	mvco:ModifyCopy
mx:move	subclass Of	mvco:MoveContent
mx:play	subclass Of	mvco:Render
mx:print	subclass Of	mvco:Render
mx:reduce	subclass Of	mvco:Copy
mx:uninstall	subclass Of	mvco:MoveContent

### ALIGNMENT WITH CREATIVE COMMONS

Creative Commons provides a RDF representation of the well known CreativeCommons licenses [5]. This description includes 16 classes and 10 properties, and for each of them, according to their definition a possible mapping is given here.

<b>CreativeCommons RDF</b>	<b>Relation</b>	<b>MVCO</b>
cc:Work	equivalent to	mvco:IPEntity
cc:License	equivalent to	mvco:Permission
cc:Jurisdiction	subclass Of	mvco:Fact
cc:Permission	equivalent to	mvco:Action isPermittedIn mvco:Permission
cc:Reproduction	equivalent to	mvco:MakeCopy
cc:Distribution	equivalent to	mvco:Distribute U mvco:PublicRepresentation
cc:Derivative	equivalent to	mvco:MakeAdaptation
ccHighIncomeNationUse	subclass of	mvco:Fact

cc:Sharing	subclass of	mvco:Distribute
cc:Requirement	equivalent to	mvco:Fact isRequiredIn mvco:Permission
cc:Prohibition	equivalent to	mvco:Fact NOT requiredIn mvco:Permission
cc:permits	equivalent to	mvco:permitsAction
cc:requires	equivalent to	mvco:hasRequired
cc:prohibits	equivalent to	mvco:hasRequired NOT mvco:Fact

## ALIGNMENT WITH MUSIC ONTOLOGY

The Music Ontology (MO) [6] is an attempt to link all the information about musical artists, albums and tracks in the Music world. From its revision 1.11 in March 2007, the MO includes the music creation workflow. The next tables show the possible equivalences regarding Users:

Music Ontology	Comment	Media Value Chain Ontology
foaf:Agent	equivalent to	mvco:User
mo:Composer	equivalent to	mvco:Creator
mo:Arranger	subclass of	mvco:Creator
mo:Conductor	subclass of	mvco:Instantiator
mo:SoundEngineer	subclass of	mvco:Producer
mo:Listener	equivalent to	mvco:EndUser

While the next table shows the relations about IPEntities.

Music Ontology	Comment	Media Value Chain Ontology
frbr:Endeavour	equivalent to	mvco:IPEntity
mo:MusicalWork	subclass of	mvco:Work
mo:MusicalExpression	superclass of	mvco:Manifestation and mvco:Instance
mo:MusicalManifestation	equivalent to	mvco:Copy
mo:MusicalItem	equivalent to	mvco:Product

Some actions can also be mapped:

Music Ontology	Comment	Media Value Chain Ontology
mo:Composition	subclass of	mvco:MakeWork
mo:Performance	subclass of	mvco:MakeInstance
mo:Recording	subclass of	mvco:MakeCopy



## REFERENCES

- [1] WIPO Intellectual Property Handbook: Policy, Law and Use. WIPO Publication 489e, ISBN 92-805-1291-7
- [2] The Digital Media Project, “Approved Document No 2, WD1.1 – Technical Reference: Architecture, Version 3.0”, No. 1002/GA15, July 2007
- [3] The Digital Media Project, “Approved Document No 6, WD1.1 – Technical Reference: Terminology, Version 3.0”, No. 1006/GA15, July 2007
- [4] ISO/IEC IS 21000-5 – Information technology – Multimedia Framework (MPEG-21) – Part 5: Rights Expression Language
- [5] Describing CreativeCommons in RDF [online] <http://creativecommons.org/ns>
- [6] The Music Ontology, Frédérick Giasson, Yves Raimond, [online] <http://www.musicontology.com>