

# EBU

OPERATING EUROVISION AND EURORADIO

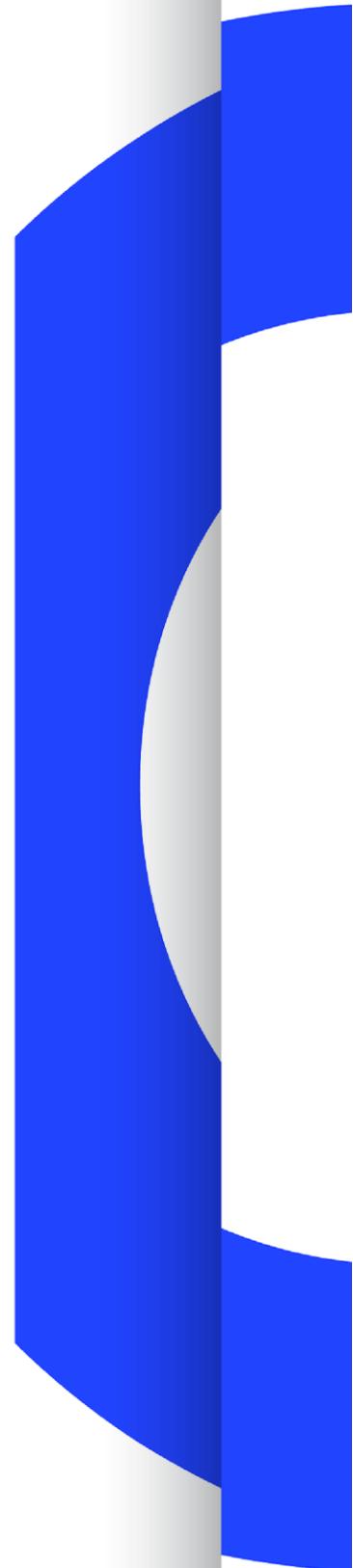
## Tech 3351

# EBU CLASS CONCEPTUAL DATA MODEL (CCDM)

SOURCE: MIM-AI

Version 2.2

Geneva  
April 2020



This page and others in the document are intentionally left blank to maintain pagination for two-sided printing

## Introduction

The EBU Class Conceptual Data Model (CCDM) is an ontology defining a basic set of classes and properties as a common vocabulary to describe business objects, e.g. programmes, articles and other types of content, and their relations in the business processes of media enterprises. Examples are programmes in their different phases of creation from commissioning to delivery, their associated rights or publication events, etc.

CCDM is a common framework and users are invited to, and should, further enrich the model with classes and properties fitting their needs more specifically. Properties for describing each of the objects can be found in EBUCore, or you are welcome to define your own.

This is version 2.2 of the "CCDM".

The CCDM has been purposefully designed as a minimum and flexible set of classes for a wide range of broadcasting applications, including archives, exchange and media service-oriented production, semantic web and linked data.

The CCDM specification combines several aspects from existing models and specifications into a common framework. It has been built over several EBU attempts to represent broadcasting as a simple logical model. It has benefited from EBU work in metadata modelling (P META and EBUCore) and semantic web developments. The distribution part has been designed to seek maximum mapping to TV Anytime and the "BBC Programmes Ontology".

The CCDM ontology is represented in RDF/OWL and associated class diagrams.

More information on EBU metadata activities is provided on the EBU TECHNICAL website (<http://tech.ebu.ch/metadata>).

### **Terms and Conditions of Use**

This EBU CCDM is freely available for all to use, but you should take note of the following:

© EBU 2020.

REDISTRIBUTION AND USE OF THIS SPECIFICATION AND ASSOCIATED RESOURCES IS PERMITTED PROVIDED THAT THE FOLLOWING CONDITIONS ARE MET:

REDISTRIBUTIONS MUST RETAIN THE ABOVE COPYRIGHT NOTICE, THIS LIST OF CONDITIONS AND THE FOLLOWING DISCLAIMER IN THE DOCUMENTATION AND/OR OTHER MATERIALS PROVIDED WITH THE DISTRIBUTION;

NEITHER THE NAME OF THE EBU NOR THE NAMES OF ITS CONTRIBUTOR(S) MAY BE USED TO ENDORSE OR PROMOTE PRODUCTS DERIVED FROM THIS SPECIFICATION AND ASSOCIATED RESOURCES WITHOUT SPECIFIC PRIOR WRITTEN PERMISSION.

DISCLAIMER: THIS SPECIFICATION AND ASSOCIATED RESOURCES IS PROVIDED BY THE COPYRIGHT OWNER "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS [SOFTWARE], EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.



# Contents

Introduction .....	3
<b>1. Scope .....</b>	<b>7</b>
1.1 Rationale.....	7
<b>2. Class Conceptual Data Model .....</b>	<b>8</b>
2.1 Main principles .....	8
2.2 Classes, Relationships and Properties.....	9
2.2.1 Legal, Commercial and Regulatory domain .....	9
2.2.1.1 Asset.....	10
2.2.1.1.1 <i>AssetValue</i> .....	12
2.2.1.2 <i>Rights</i> .....	12
2.2.1.3 <i>Contract</i> .....	13
2.2.1.3.1 <i>ContractCost</i> .....	13
2.2.1.4 <i>Rule</i> .....	14
2.2.2 Editorial Domain .....	14
2.2.2.1 <i>EditorialObject</i> .....	15
2.2.2.2 <i>TimelineTrack</i> .....	18
2.2.2.3 <i>Location</i> .....	18
2.2.2.4 <i>Event</i> .....	18
2.2.3 Entity domain.....	19
2.2.3.1 <i>Agent</i> .....	19
2.2.3.2 <i>Person</i> .....	20
2.2.3.3 <i>Organisation</i> .....	20
2.2.3.4 <i>Crew</i> .....	21
2.2.3.5 <i>Role</i> .....	21
2.2.3.6 <i>Artefact</i> .....	21
2.2.4 Production Domain.....	22
2.2.4.1 <i>Resource</i> .....	22
2.2.4.2 <i>MediaResource</i> .....	23
2.2.4.3 <i>Track</i> .....	24
2.2.4.4 <i>Format</i> .....	25
2.2.4.5 <i>Essence</i> .....	27
2.2.4.6 <i>PhysicalResource</i> .....	27
2.2.4.7 <i>Artefact</i> .....	28
2.2.4.8 <i>ProductionJob</i> .....	28
2.2.4.9 <i>ProductionDevice</i> .....	29
2.2.4.10 <i>OnStagePosition</i> .....	30
2.2.5 Distribution Domain .....	31
2.2.5.1 <i>PublicationEvent</i> .....	32

- 2.2.5.2 *Service* ..... 33
- 2.2.5.3 *ConsumptionDeviceProfile* ..... 33
- 2.2.6 Consumption Domain ..... 34
  - 2.2.6.1 *ConsumptionEvent* ..... 35
  - 2.2.6.2 *ConsumptionDevice* ..... 36
  - 2.2.6.3 *ConsumptionLicence* ..... 37
  - 2.2.6.4 *Consumer* ..... 37
  - 2.2.6.5 *Account* ..... 38
  - 2.2.6.6 *ResonanceEvent* ..... 38
- 2.2.7 Planning Domain ..... 39
  - 2.2.7.1 *Campaign* ..... 40
  - 2.2.7.2 *PublicationPlan* ..... 40
  - 2.2.7.3 *ProductionOrder* ..... 41
  - 2.2.7.4 *Audience* ..... 42
  - 2.2.7.5 *Resonance* ..... 42
- 2.2.8 Financial Domain ..... 43
  - 2.2.8.1 *AssetValue* ..... 44
  - 2.2.8.2 *ContractCost* ..... 44
- 2.2.9 Audit and Assessment Domain ..... 44
  - 2.2.9.1 *AuditJob* ..... 45
  - 2.2.9.2 *Measure* ..... 46
  - 2.2.9.3 *AuditReport* ..... 46
- 3. Implementation Guidelines / Questions & Answers ..... 47**
  - 3.1 General remarks ..... 47
  - 3.2 Examples provided by SRG SSR, Swiss Confederation ..... 47
    - 3.2.1 Modelling Different Viewpoints with CCDM ..... 47
    - 3.2.2 CCDM as a Comprehensive Representation of Business Objects ..... 50
  - 3.3 Example provided by TV2, Norway ..... 52
  - 3.4 The total class diagram ..... 53
  - 3.5 The RDF ontology ..... 53
  - 3.6 Further questions? ..... 53
- 4. CCDM Compliance ..... 53**
- 5. Download Zone ..... 53**
- 6. Licensing regime ..... 54**
- 7. Maintenance ..... 54**
- 8. Useful links ..... 54**
- Annex A: EBU CCDM ontology ..... 55**

## EBU Class Conceptual Data Model (EBU CCDM)

EBU Committee	First Issued	Revised	Re-issued
TC	October 2012	April 2020	

**Keywords:** Class, Model, Metadata, Business, Object, Radio, Television, Production, SOA, Semantic Web, Linked Data, Internet, Web Publishing.

### 1. Scope

The EBU Class Conceptual Data Model (CCDM) is an ontology defining a basic set of classes and properties as a common vocabulary to describe business objects in their different phases of creation from commissioning to delivery, i.e. the full lifecycle of a business process. CCDM is a common framework and users are welcome to further enrich the model with Classes and properties fitting their needs more specifically.

The CCDM has deliberately been designed as a minimum and flexible set of classes for a wide range of applications including but not restricted to archives, exchanges, media service oriented production, broadcasting, Internet delivery, Semantic Web modelling and Linked Open Data (LOD).

This specification is a class model, an ontology, and not a metadata specification. Metadata properties and datatypes (other than the relationships between Classes) are **indicative**. Users willing to adapt the CCDM model to their needs are invited to describe CCDM classes and custom extensions either using properties from EBU Tech 3293 (EBUCore metadata set) or other metadata specifications (e.g. TV-Anytime or in-house metadata schemes).

The CCDM specification is combining several aspects from existing models and specifications into a common framework. It has been built over several EBU attempts to represent broadcasting as a simple logical model. It has benefited from EBU work in metadata modelling (P-META and EBUCore) and semantic web developments. The distribution part has specifically been designed to seek maximum mapping to TV-Anytime and the "BBC Programmes ontology".

The CCDM ontology is represented in RDF/OWL.

#### 1.1 Rationale

It is vital for content providers and broadcasters to have a well-defined class model. This is a necessary step towards:

- Greater understanding of the business models and workflows;
- Process optimisation with easier and more reliable data exchange;
- A simpler and rationalised description of Media Classes;
- The easier implementation of media service-oriented production architectures;
- The adoption of new information management models such as Semantic Web and Linked Data (enrichment, improved searching and ubiquity).

The CCDM has been designed to let implementers adapt the names of the Classes and their Relationships to their respective modelling needs. Each organisation is encouraged to make its proper analysis and to create its own model starting from the CCDM framework as a common basis for comparison with models from other CCDM implementers.

## 2. Class Conceptual Data Model

### 2.1 Main principles

The EBU CCDM is composed of:

- **Classes:** directly related (e.g. a programme, a part, a clip, a track) or associated (e.g. a person, a location) to media.
  - Note: equivalent to the notion of class used in semantic web modelling (see RDF & OWL Primers), also referred to as 'Business Objects' or 'concepts' in certain projects, see also [http://protege.stanford.edu/publications/ontology\\_development/ontology101.pdf](http://protege.stanford.edu/publications/ontology_development/ontology101.pdf) . W3C's Media-Ontology (MA-ONT) is based on the CCDM class model (<http://www.w3.org/ns/ma-ont.rdf>).
- **Relationships:** linking Classes (e.g. 'Programme hasContributor Person')
  - Note: equivalent to the notion of *objectProperties* used in semantic web modelling (see RDF and OWL Primers)
- **Properties:** defining intrinsic characteristics of Classes (e.g. 'bitrate' expressed as an integer or a person 'name' expressed as a string)

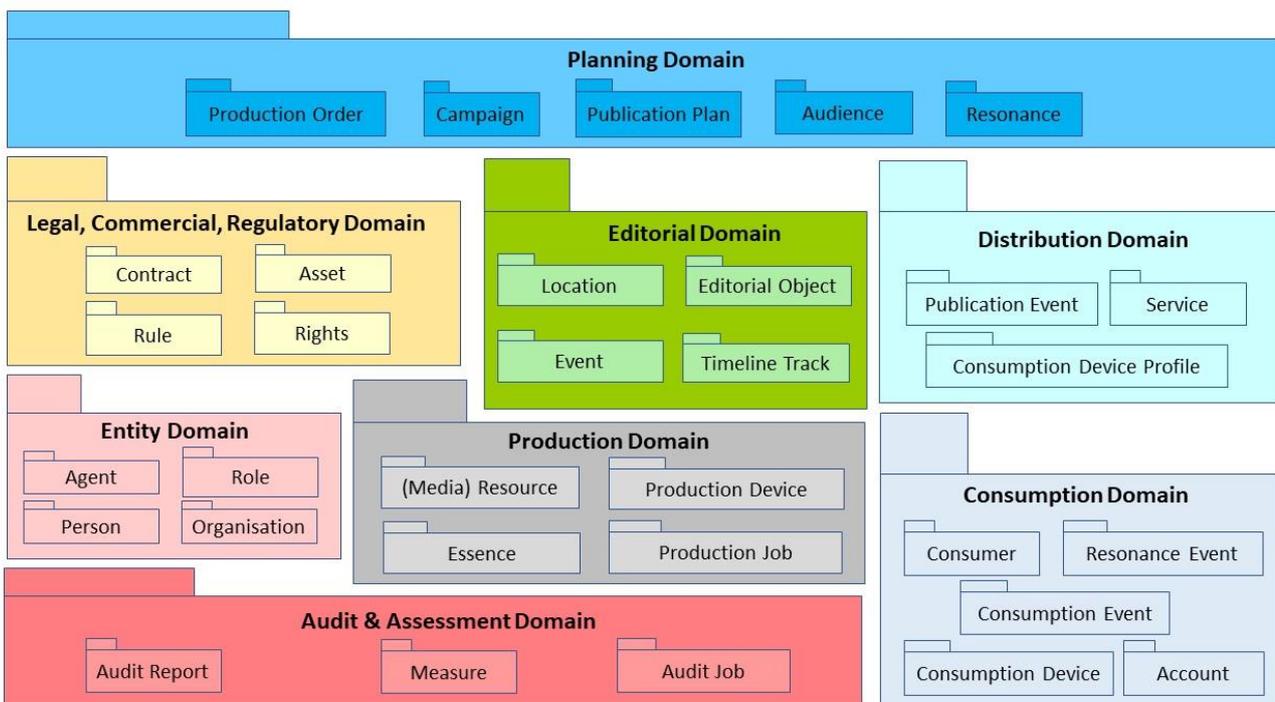


Figure 1: CCDM domains

As shown in Figure 1, the model is defined around seven main domains:

- **Planning Domain** is where *Resonance* from the *Audience* is analysed to understand the demand which in turn is met by a strategy in form of a *PublicationPlan*, leading to the

commissioning of production (*ProductionOrders*) and coordination of publication (*Campaigns*).

- **Legal, Commercial and Regulatory domain** is where *Contracts*, intellectual property and other rights associated to content and its manifestations are being managed. The central class of the Legal Domain is the *Asset*, which establishes the association of an *EditorialObject* with Intellectual Property and *Rights* related information.
- **Distribution Domain** is where any form of publishing, play-out or distribution is covered. The central Class is the *PublicationEvent* that plays out an *Essence*, (i.e. the media object that was the result of the *ProductionJob*.) through a *Service* that is consumable on a type of device represented by a *ConsumptionDeviceProfile*.
- **Editorial Domain** is where concept related, and content related information is being managed. Furthermore, all editing decisions are represented here. The *EditorialObject* is the central class of the domain. It can be grouped, and it can be ordered on a timeline through *TimelineTrack*. Associated objects like *Location* or *Event* are represented.
- **Entity domain** is a where actors/contributors, like persons and companies are described through *Agent*, *Role*, *Person* and *Organisation*.
- **Production Domain** is where *Production Orders* are realised through the acquisition of the necessary *MediaResources* (e.g. manufacturing an object through the *ProductionJob* with *ProductionDevices*, purchase or retrieval of material) according to the *ProductionPlan*. *MediaResources* ready for publication use the *Essence* class for connecting the content to a certain publication.
- **Consumption Domain** is where the consumption of media is modelled. Important classes in this domain is the *ConsumptionEvent*, that corresponds with the *PublicationEvent* in the *DistributionDomain*. A *Consumer* uses a *Consumption Device* to access the *Service* and possibly create *ResonanceEvents*. *Account* is a *Consumer's* registration with the media enterprise to handle authorisation, personalisation, monetization, etc.
- **Audit and Assessment Domain** is where auditing is defined to measure the quality of content (editorial or technical), against contractual rules and expressed through *Measures*. An *AuditJob* results in an *AuditReport*.

The EBU CCDM has been designed to let users adapt the names of Classes and relationships to their respective modelling needs. For example, a class *EditorialObject* can be of type *Programme*, *Item* or *Shot*, but it can also represent a group *Series*, *Serial* or *Season*. The definition of appropriate properties is left to the user. A core set of classes and properties is proposed in EBU Tech 3293, EBUCore, or in other metadata specifications (e.g. TV-Anytime or in-house metadata schemes).

## 2.2 ***Classes, Relationships and Properties***

See **Figure 1**, which illustrates the relationships between domains and objects.

### 2.2.1 **Legal, Commercial and Regulatory domain**

It is the domain in which intellectual property, rights, regulations, legal constraints, compliance standards, and contracts are being managed and associated to a *MediaResource* and / or an *EditorialObject*, and by inference to a *PublicationEvent* (incl. exploitation and distribution conditions), to define an *Asset*. The domain also covers the commissioning of productions and material.

The central class of the domain is the *Asset* that acts like a conjunction between a set of *Rights* or legal constraints and an *EditorialObject*.

### 2.2.1.1 Asset

Definition:

The class *Asset* is an object to which an identifier will be associated at commissioning. It will serve as a central reference point to manage rights associated to *EditorialObjects*, *MediaResources* or *Essences*, and - by inference - *PublicationEvents* (distribution and exploitation conditions).

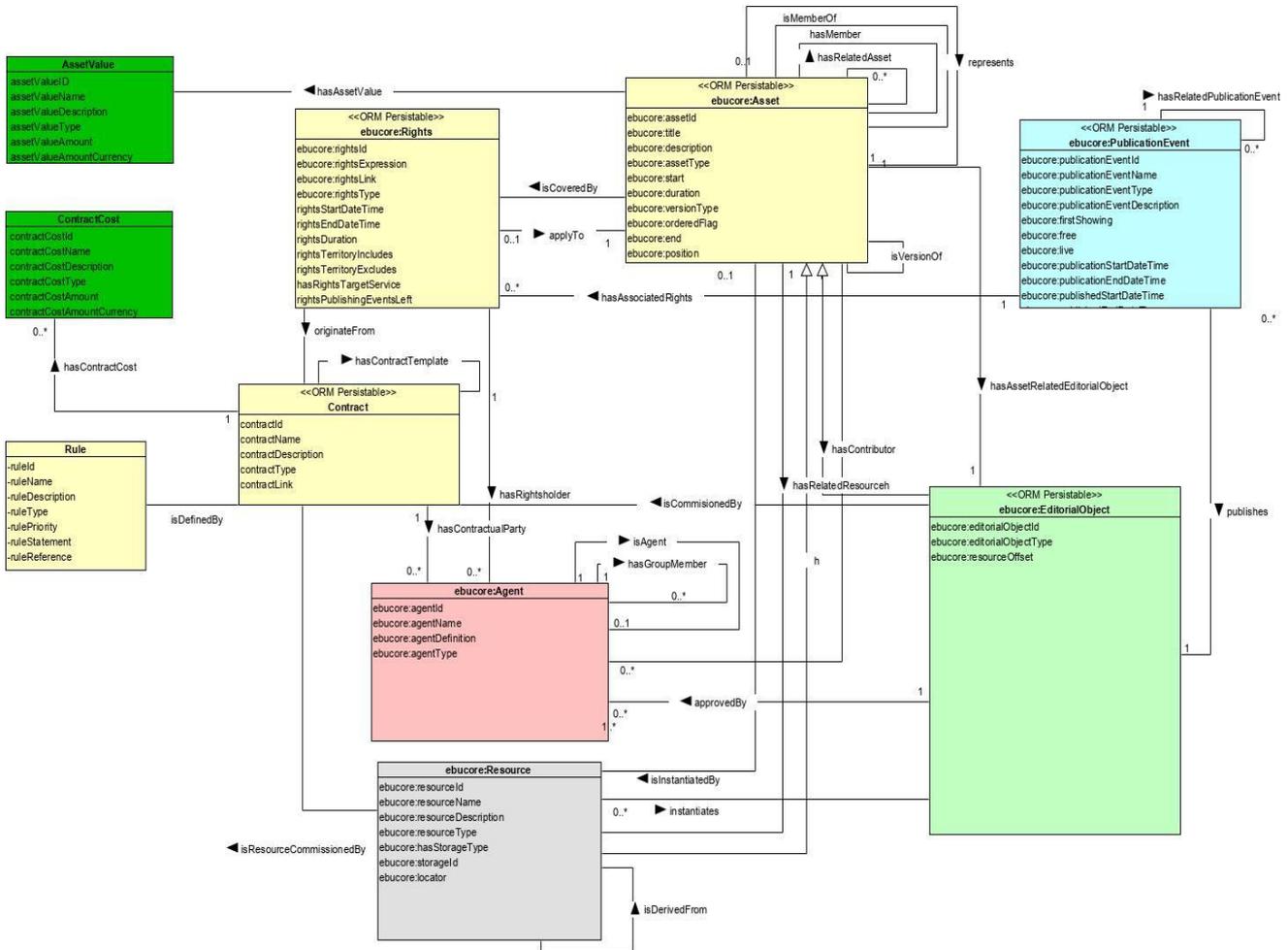


Figure 2: The Asset

Remember that the *MediaResources* or *Essences* will, in this model, always be the representation/instantiation of an *EditorialObject*.

The *Asset* class is also a superclass for *EditorialObject* and *Resource* in the way that Rights information can be added to those classes for a simple representation.

Example:

The CCDM model allows the association of Rights to an *EditorialObject* representing an *Essence*.

Class relations	
hasAssetRelatedEditorialObject	A pointer to the <i>EditorialObject</i> that the <i>Asset</i> links to its <i>Rights</i>
hasAssetRelatedResource	A pointer to the <i>Resource</i> that the <i>Asset</i> links to its <i>Rights</i>
hasRelatedAsset	A pointer to another asset (e.g. a TV Series) that the <i>Asset</i> links to

isCoveredBy	A pointer to the <i>Rights</i> associated to the <i>EditorialObject</i>
hasAssetValue	A pointer to the <i>AssetValue</i> associated with the <i>Asset</i>
hasRelatedResource	A relationship to identify a <i>Resource</i> that are related to the <i>EditorialObject</i>
isInstantiatedBy	A relationship to identify the <i>Resource</i> that instantiates the <i>EditorialObject</i>
isVersionOf	To identify <i>EditorialObjects</i> presenting alternative version of the content.
existsAs	To identify <i>EditorialObjects</i> representing alternative representations of the content
approvedBy	An <i>Agent</i> , like the editor of the day, that approves the <i>Asset</i> for publication
hasRelatedLocation	Optionally, one (or more) <i>Location</i> related to the <i>EditorialObject</i> characterised by its type (e.g. shooting or fictional).
hasRelatedEvent	Optionally, one (or more) <i>Event</i> related to the <i>EditorialObject</i> characterised by its type (e.g. sport event / meeting).
represents	An <i>EditorialObject</i> represents an <i>Asset</i> .
hasRelatedArtefact	A relationship to an <i>Artefact</i> related to the <i>EditorialObject</i>
hasRelatedAuditReport	To associate an <i>AuditReport</i> with an <i>Asset</i> .
hasContributor	To identify <i>Agents</i> contributing to the <i>Asset</i> .
Etc.	Other class relationships can be associated to an <i>Asset</i> . See EBU Tech 3293, EBUCore
<b>Class Properties</b>	
assetId	An identifier associated with the <i>Asset</i>
title	The main Title by which of the <i>EditorialObject</i> is known. As an example.
description	Optionally one (or more) description of the <i>EditorialObject</i>
assetType	The type assigned to the <i>Asset</i>
editUnit	The unit used to express start, duration and <i>resourceOffset</i>
orderedFlag	If 'true', a flag which indicates that the members of the <i>EditorialObject</i> are ordered (e.g. membership is subject to a strict sequence such as episodes in a series)
versionType	A string to optionally identify the version of the <i>EditorialObject</i> such as lengthened, shortened, signed, closed-captioned, etc.
position	The position or index of the <i>EditorialObject</i> in an <i>EditorialObject</i> of type 'rundown', or in an ordered <i>Group</i>
start	The starting point of the member, i.e. the part, in an <i>EditorialObject</i> or in a <i>TimelineTrack</i>
duration	The duration of the member in an <i>EditorialObject</i> or in a <i>TimelineTrack</i>
end	The ending point of the member, i.e. the part, in an <i>EditorialObject</i> or in a <i>TimelineTrack</i>
Etc.	Other properties can be associated to an <i>Asset</i> . See EBU Tech 3293, EBUCore.

### 2.2.1.1.1 AssetValue

Definition:

The class *AssetValue* is an object that is used to specify the value of an *Asset*.

Class Properties	
assetValueId	An identifier associated with the <i>AssetValue</i>
assetValueName	A name given to the <i>AssetValue</i>
assetValueDescription	A description of what the <i>AssetValue</i> represents
assetValueType	The type assigned to the <i>AssetValue</i>
assetValueAmount	The actual estimated value of the <i>Asset</i>
assetValueAmountCurrency	The currency in which the value is expressed
Etc.	Other properties can be associated to an <i>AssetValue</i>

### 2.2.1.2 Rights

Definition:

The class *Rights* defines rights that originate from a contract. The *Rights* are associated to a *MediaResource* through the definition of an *Asset*.

Class relations	
applyTo	A pointer to the <i>Asset</i> , which in turn has <i>EditorialObject</i> , to which the <i>Rights</i> apply
originateFrom	A pointer to the <i>Contract</i> granting the <i>Rights</i>
hasRightsholder	The <i>Agent</i> related to the <i>Rights</i> . Can be sub-classed to specify the kind of relationship
Etc.	Other class relationships can be associated to <i>Rights</i> . See EBU Tech 3293, EBUCore
Class Properties	
rightsID	An identifier associated with the <i>Rights</i>
rightsExpression	The expression of <i>Rights</i>
rightsLink	A link to e.g. a web resource where the <i>Rights</i> terms can be found
rightsType	A type associated to <i>Rights</i> e.g. licensing terms
rightsStartDateTime	The start of the time interval where the <i>Rights</i> is valid
rightsEndDateTime	The end of the time interval where the <i>Rights</i> is valid
rightsDuration	The extend of a <i>Rights</i> period, when it is not expressed using <i>rightsEndDateTime</i>
rightsTerritoryIncluding	Territory covered by the <i>Rights</i>
rightsTerritoryExcluding	Territory excluded from the <i>Rights</i>
hasRightsTargetService	The <i>Service</i> associated with the <i>Rights</i>
rightsPublishingEventsLeft	The number of publishing events left covered by the <i>Rights</i>
rightsUsageRestriction	Restrictions and other constraints defining how the material can be used
Etc.	Other properties can be associated to <i>Rights</i> . See EBU Tech 3293, EBUCore.

### 2.2.1.3 Contract

Definition:

The class *Contract* represents any legal document covering *Rights* - or commissioning issues. This object/class covers the production order and sales order combined. The *Contract* connects the *Rights* to any *RightsHolders*. A *Contract* defines one or more set of *Rights*.

Class relations	
hasContractualParty	A list of the parties involved with the <i>Contract</i> . Can be specified by a sub-property or a subclass to describe the relationship in more detail
hasContractTemplate	Relation to the template the <i>Contract</i> is derived from
hasContractRelatedCost	A pointer to the <i>ContractCost</i> associated with the <i>Contract</i>
Etc.	Other class relationships can be associated to a <i>Contract</i> . See EBU Tech 3293, EBUCore
Class properties	
contractID	An Identifier associated with the <i>Contract</i>
contractName	The name given to a <i>Contract</i>
contractDescription	A description of the <i>Contract</i>
contractType	The type of <i>Contract</i>
contractLink	URL pointing to a document describing the <i>Contract</i>
Etc.	Other properties can be associated to a <i>Contract</i> . See EBU Tech 3293, EBUCore.

#### 2.2.1.3.1 ContractCost

Definition: The class *ContractCost* is an object that is used to specify the cost associated with a *Contract*.

Class Properties	
contractCostId	An identifier associated with the <i>ContractCost</i>
contractCostName	A name given to the <i>ContractCost</i>
contractCostDescription	A description of what the <i>ContractCost</i> represents
contractCostType	The type assigned to the <i>ContractCost</i>
contractCostAmount	The actual cost figure associated with the <i>ContractCost</i>
contractCostAmountCurrency	The currency in which the cost figure is expressed
Etc.	Other properties can be associated to a <i>ContractCost</i>

### 2.2.1.4 Rule

Definition:

The class Rule is an object used to specify contractual requirements. Rules are assessed during an audit and associated measures.

Class relations	
isDefinedBy	A link to a Contract from which the <i>Rule</i> was established
Etc.	Other relations can be established with a <i>Rule</i>
Class Properties	
ruleId	An identifier associated with the <i>Rule</i>
ruleName	A name given to the <i>Rule</i>
ruleDescription	A description of what the <i>Rule</i> represents
ruleType	The type assigned to the <i>Rule</i>
ruleStatement	A statement to further specify the <i>Rule</i>
ruleReference	A reference associated with a <i>Rule</i> .
rulePriority	To establish ranking and priorities between <i>Rules</i>
Etc.	Other properties can be associated to a Rule

### 2.2.2 Editorial Domain

The Editorial Domain is the domain within which a concept is defined and commissioned before fabrication and distribution. All metadata related to the idea of a programme (e.g. content, format, purpose, audience, schedule window), related to the content of the programme (e.g. titles, subjects, contributors, locations, events) and all editing decisions are represented in the respective classes.

The central class in the Editorial Domain is the *EditorialObject*.

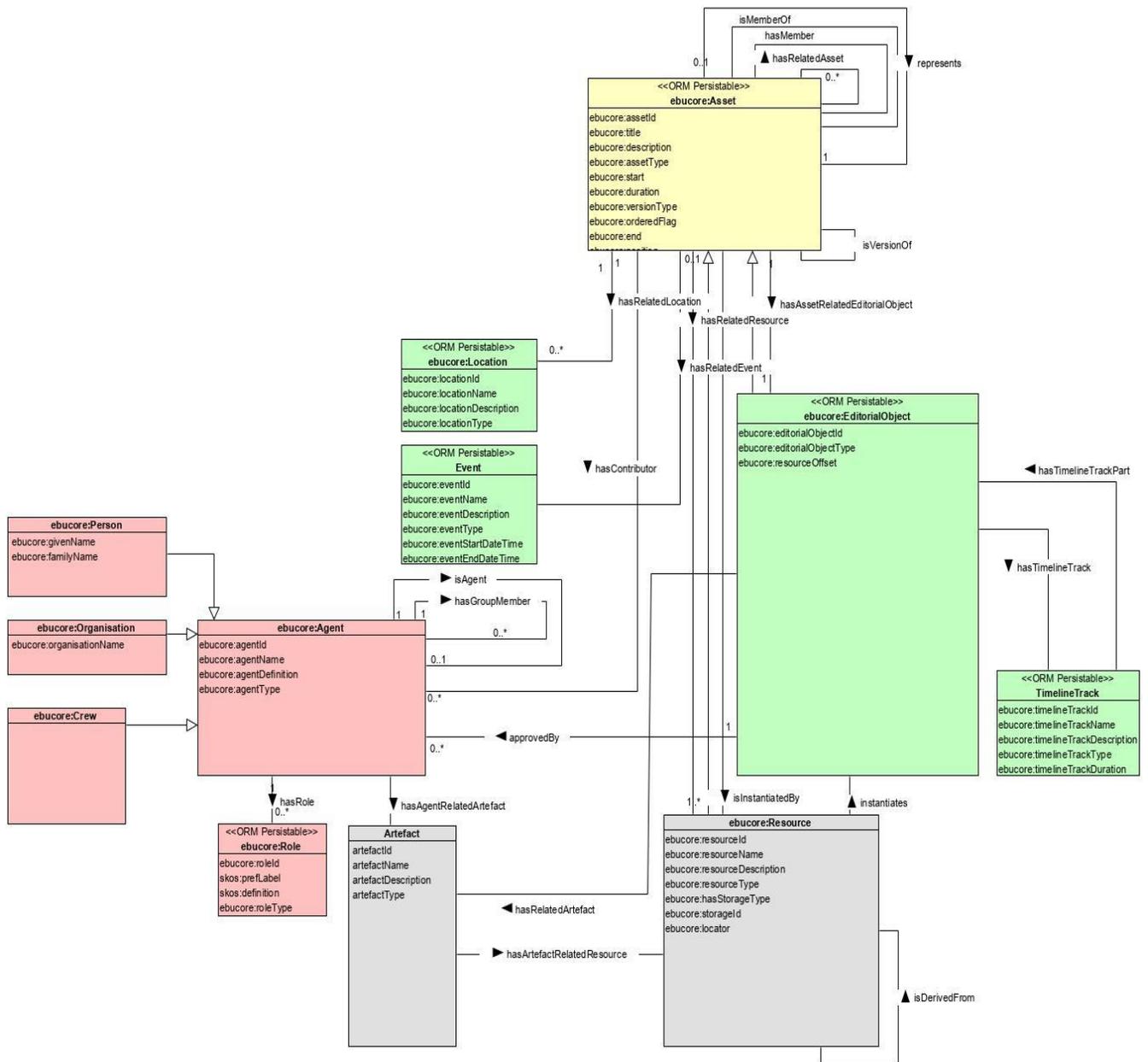


Figure 3: Classes around the *EditorialObject*

### 2.2.2.1 EditorialObject

Definition:

The class *EditorialObject* describes an idea or story and will be used to transform a concept into an editorial definition of a *MediaResource* before fabrication (Production Domain) and Distribution (Distribution Domain). An *EditorialObject* is associated with a set of descriptive metadata summarising e.g. editing decisions.

An *EditorialObject* can be a *Group*.

An *EditorialObject* can also be a part of another *EditorialObject*, which is defined by its start time and duration.

*EditorialObjects* can be ordered either as *Groups* or as items on a timeline.

Examples:

*Programme, Item, Shot, Part, Chapter, Segment*, and where the group properties are in use: *Series, Serial*, compilation, collection, item group, item block.

A simplified use-case:

A TV news broadcast consists of two news items. One news item contains the last ten seconds of a one-minute long interview taken from another source (i.e. from 50'' to 60''). This could be modelled as follows:

- The *NewsBroadcast* is linked to a *MediaResource* using the *instantiates*-property
- The *NewsItems* are linked to the *NewsBroadcast* using a *TimelineTrack*.
- The *InterviewPart* is linked to the *NewsItem* using the *hasMember*-property. *Start* and *Duration* are properties within the *InterviewPart* indicating its appearance within the *NewsItem2*.
- The *InterviewPart* is linked to its original source using the *existsAs*-property
- The *Interview* instantiates a *MediaResource*, which in turn is linked from the *MediaResource* of the *NewsBroadcast* using the *hasSource*-property
- Representation of segmentation: *TimelineTracks* are preferred over *hasPart*-properties, when a rundown is needed, e.g. for playout.

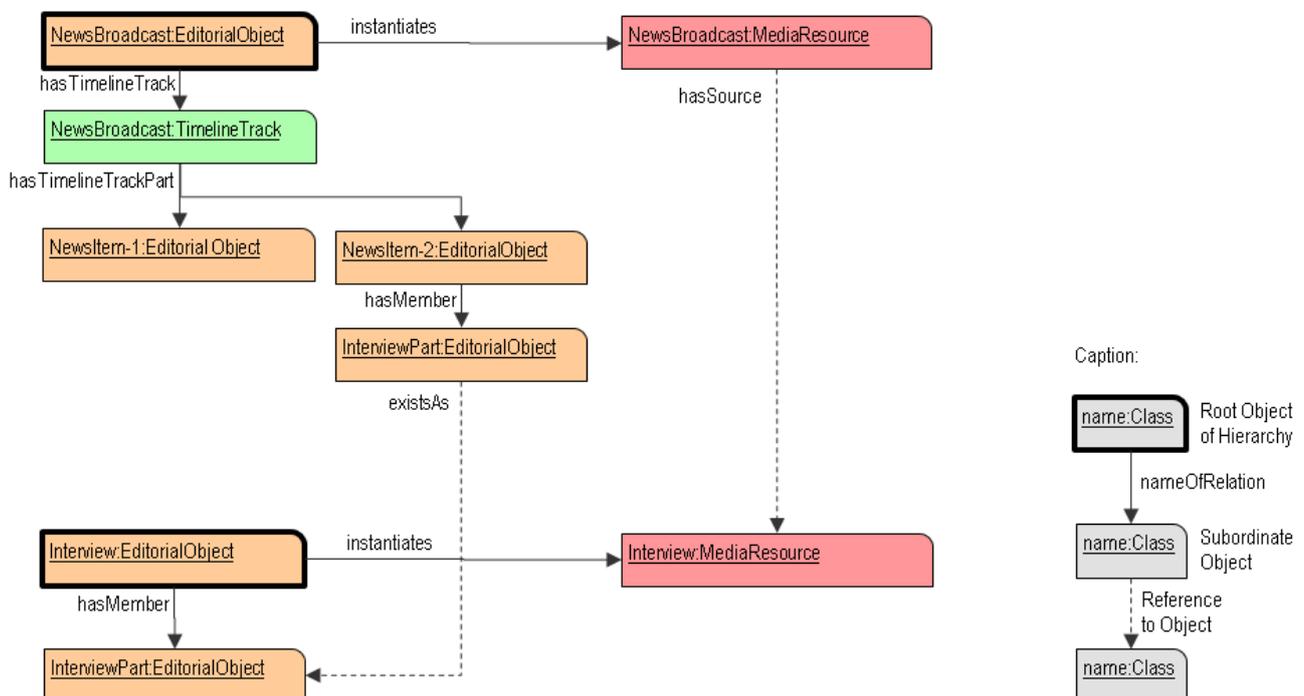


Figure 4: Illustration of use-case

Class relations	
hasAssociatedProductionJob	A <i>ProductionJob</i> represents a production process through which an <i>EditorialObject</i> is being instantiated into a <i>MediaResource</i> and / or <i>Essence</i> .
hasTimelineTrack	To associate a <i>TimelineTrack</i> , e.g. a <i>RunDown</i> , with an <i>EditorialObject</i> itself constituted of other <i>EditorialObjects</i> .
isCommisionedBy	The <i>Contract</i> that commissions the <i>EditorialObject</i>

hasRelatedResonanceEvent	Used when e.g. an interactive Tweet from a consumer is being used on-screen in a television show, - a <i>ResonanceEvent</i> triggers and is the base for the creation a new <i>EditorialObject</i> .
Etc.	Other class relationships can be associated with an <i>EditorialObject</i> . See EBU Tech 3293, EBUCore.
<b>Class relations inherited from Asset</b>	
isMemberOf	A list of Groups that the <i>EditorialObject</i> is a member of.
hasMember	A list of <i>EditorialObjects</i> that the <i>EditorialObject</i> contains that is not a part of a timeline. Series-episode is an example of such a relationship
hasRelatedResource	A relationship to identify a <i>Resource</i> that are related to the <i>EditorialObject</i>
isInstantiatedBy	A relationship to identify the <i>Resource</i> that instantiates the <i>EditorialObject</i>
isVersionOf	To identify <i>EditorialObjects</i> presenting alternative version of the content.
existsAs	To identify <i>EditorialObjects</i> representing alternative representations of the content
approvedBy	An <i>Agent</i> , like the editor of the day, that approves the <i>EditorialObject</i> for publication
hasRelatedLocation	Optionally, one (or more) <i>Location</i> related to the <i>EditorialObject</i> characterised by its type (e.g. shooting or fictional).
hasRelatedEvent	Optionally, one (or more) <i>Event</i> related to the <i>EditorialObject</i> characterised by its type (e.g. sport event / meeting).
represents	An <i>EditorialObject</i> represents an <i>Asset</i> .
hasRelatedArtefact	A relationship to an <i>Artefact</i> related to the <i>EditorialObject</i>
<b>Class hierarchy</b>	
subclass	<i>EditorialObject</i> is a subclass of <i>Asset</i>
<b>Class Properties</b>	
editorialObjectType	The type of <i>EditorialObject</i> e.g. Programme, Item
editorialObjectId	Optionally one (or more) identifier attributed to the <i>EditorialObject</i>
resourceOffset	The start offset of the related resource, used if the related resource is not used from its start
Etc.	Many other properties can be associated with an <i>EditorialObject</i> . See EBU Tech 3293, EBUCore.

### 2.2.2.2 TimelineTrack

Definition:

A *TimelineTrack* is used to define timelines, i.e. a time related sequence of *EditorialObjects* (or *Part of EditorialObjects*).

Class relations	
hasTimelineTrackPart	To identify the Parts of a <i>TimelineTrack</i> . I. e. <i>EditorialObjects</i> with a start time and duration.
Etc.	Many other relationships can be associated with a <i>TimelineTrack</i> . See EBU Tech 3293, EBUCore.
Class properties	
timelineTrackID	The identifier attributed to a <i>TimelineTrack</i> .
timelineTrackType	E.g. rundown or other types not defined as subclass in the specification
timelineTrackName	The name given to the timeline
timelineTrackDescription	The description of a <i>TimelineTrack</i>
timelineTrackduration	The duration of the <i>TimelineTrack</i> in the <i>EditorialObject</i>
Etc.	Many other properties can be associated with an <i>TimelineTrack</i> . See EBU Tech 3293, EBUCore.

### 2.2.2.3 Location

Definition:

The class *Location* is used to define the locations, e.g. spatial coverage of the story or recording locations like studios or in the field, associated with the *EditorialObjects* (or *Part of EditorialObjects*).

Class relations	
hasLocationRelatedEvent	An <i>Event</i> related to a <i>Location</i> .
Etc.	Many other relationships can be associated with a <i>Location</i> . See EBU Tech 3293, EBUCore.
Class properties	
locationId	To identify a <i>Location</i> in a system of defined locations.
locationName	The name of a <i>Location</i> .
locationDescription	The description of a <i>Location</i> .
locationType	The type of <i>Location</i> .
Etc.	Many other properties can be associated with a <i>Location</i> . See EBU Tech 3293, EBUCore (incl. GPS coordinates) or <i>GeoNames</i> .

### 2.2.2.4 Event

Definition:

The class *Event* is used to define the event that the *EditorialObject* covers.

Examples:

Olympic Games 1994, General election, etc.

Class relations	
hasEventRelatedLocation	A <i>Location</i> related to an <i>Event</i> .
Etc.	Many other relationships can be associated with an <i>Event</i> . See EBU Tech 3293, EBUCore.
Class properties	
eventId	To identify the <i>Event</i>
eventName	The name of an <i>Event</i>
eventDescription	The description of an <i>Event</i>
eventType	The type of an <i>Event</i>
eventStartDateTime	The time where an <i>Event</i> starts
eventEndDateTime	The time where an <i>Event</i> ends
eventDuration	The duration of an <i>Event</i>
Etc.	Many other properties can be associated with an <i>Event</i> . See EBU Tech 3293, EBUCore.

### 2.2.3 Entity domain

This is where actors, like persons and companies are described. The classes can be connected to any other class in the model where there is a need for describing ownership or contribution to data. The Agent class is specialized into *Person*, *Organisation* and *Crew*, used for needs of description of the data.

E.g. the in the planning stage we like to describe the need for the job functions in the production crew. At this stage the jobs are not assign to any people yet. So, we are using the *Crew* class for describing the functions that are needed for a production. As the planning evolves further, each of the *Crew* will be assigned an isAgent relation to a *Person*, containing the real name.

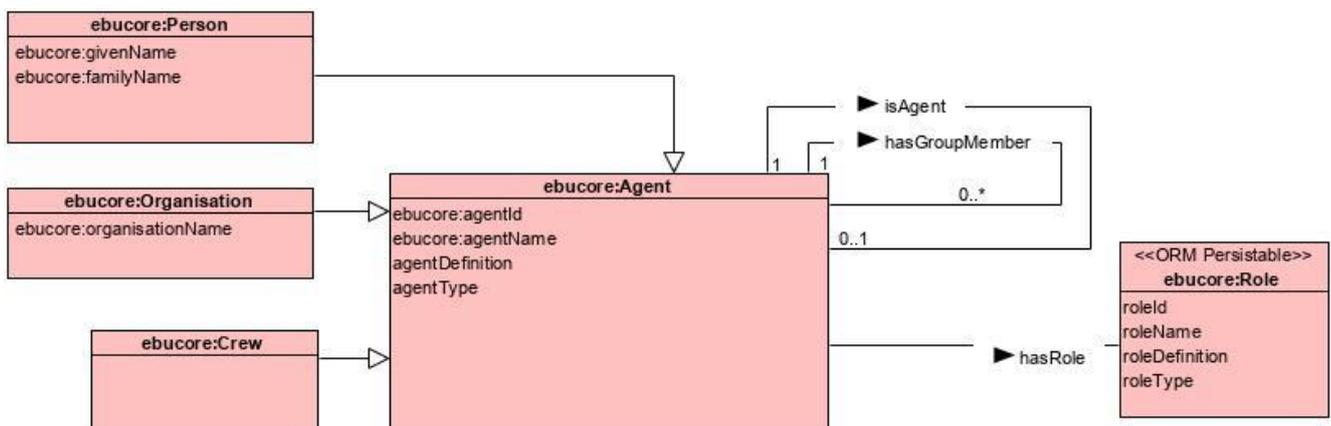


Figure 5: Entity Domain

#### 2.2.3.1 Agent

Definition:

The class *Agent* is either a *Contact/Person/Crew* or *Organisation* to which is associated a *Role* corresponding to the contribution the *Agent* brings to the realisation of a *MediaResource* or *EditorialObject*.

Examples:

Examples of *Agent’s Roles* are ‘producer’, ‘cameraman’ or ‘actor’.

Class relations	
hasRole	The <i>Role</i> of the <i>Agent</i> . <i>Role</i> refines “hasContributor”. Alternatively, a user can decide to add new class and associated relationships as contributions to an <i>EditorialObject</i> e.g. “hasContributor Creator”, “hasContributor Composer”, etc., which in turn will be refined with “hasRole” <i>Role</i>
isAgent	The relation is used for connecting the <i>Person, Organisation and Crew</i> part of the <i>Agent</i> data
hasGroupMember	Used for connecting a team or a group to its members
hasAssociatedArtefact	Relation to an Artefact associated with the Agent, e. g. a costume
hasContact	To link to another Agent/contact.
hasAgentOnStagePosition	To associate a StagePosition with an Agent
Etc.	Other class relationships can be associated with an <i>Agent</i> . See EBU Tech 3293, EBUCore.
Class Properties	
agentId	An identifier for the <i>Agent</i>
agentName	The display name given to the <i>Agent</i>
agentDescription	A description of the <i>Agent</i>
agentType	A type associated with the <i>Agent</i>
Etc.	Other class Properties can be associated with an <i>Agent</i> . See EBU Tech 3293, EBUCore.

**2.2.3.2 Person**

Definition:

The class *Person* stores the personal data such as name for an Agent. The class can be extended with *Contact* data from EBU Core.

Class relations	
Subclass	The <i>Person</i> class is a subclass of <i>Agent</i> .
Class Properties	
givenName	The name given to a Person. This is an example of how properties from EBUCore are used in CCDM
familyName	The family name of a Person.
Etc.	Other class Properties can be associated with a <i>Person</i> . See EBU Tech 3293, EBUCore.

**2.2.3.3 Organisation**

Definition:

The class *Organisation* stores the name and other data for a company. The class can be extended with *Contact* data from EBU Core.

Class relations	
Subclass	The <i>Organisation</i> class is a subclass of <i>Agent</i>
Class Properties	
organisationName	A name associated with an Organisation
Etc.	Other class Properties can be associated with an <i>Organisation</i> . See EBU Tech 3293, EBUCore.

#### 2.2.3.4 Crew

Definition:

The class *Crew* stores the job function of an unspecified crew member. The class is typically used for resource planning. *Crew* is a subclass of *Agent* and uses *Agent*'s *hasRole* to specify the job function.

Examples:

Examples of *Crew* are 'producer', 'cameraman' etc.

Class relations	
Subclass	The <i>Crew</i> class is a subclass of <i>Agent</i> .
hasRole	To define the job function of a Crew member.
Class Properties	
Etc.	Other class Properties can be associated with a <i>Crew</i> . See EBU Tech 3293, EBUCore.

#### 2.2.3.5 Role

Definition:

The *Role* played by an *Agent*. A *Role* will be identified e.g. by a concept from a SKOS Classification Scheme. *Role* is therefore to be considered as a class, i.e. a subclass of SKOS Concept.

Example:

A Contact may be an actor.

Class hierarchy	
subclass	<i>Role</i> is a subclass of skos:Concept
Class Properties	
roleId	Identifier attributed to a <i>Role</i> , preferably from a defined list of <i>Roles</i> (e.g. a SKOS ConceptId)
skos:prefLabel	A name associated with a Role
skos:definition	The definition of a Role
roleType	A type of Role
Etc.	Other class Properties can be associated with a <i>Role</i> . See EBU Tech 3293, EBUCore.

#### 2.2.3.6 Artefact

See § 2.2.4.7.

## 2.2.4 Production Domain

The Production Domain is the domain, within which production orders are realised through the acquisition of *MediaResource* (e.g. manufacturing an object through a *ProductionJob*, purchase or retrieval of material).

The central class in the Production Domain is the *MediaResource* and its *Essence* subclass.

*MediaResources* ready for publication use the *Essence* class for connecting the content to a certain publication.

A *MediaResource* has always a relation to an *EditorialObject* (Editorial Domain) describing its content. The *Essence* is a manifestation of a *MediaResource* in a particular Format that is destined for publication. The *Essence* is the result of a *ProductionJob* and is a subclass of *MediaResource* and inherits all its properties such as *Format*, *Location* and *ProductionDevice*.

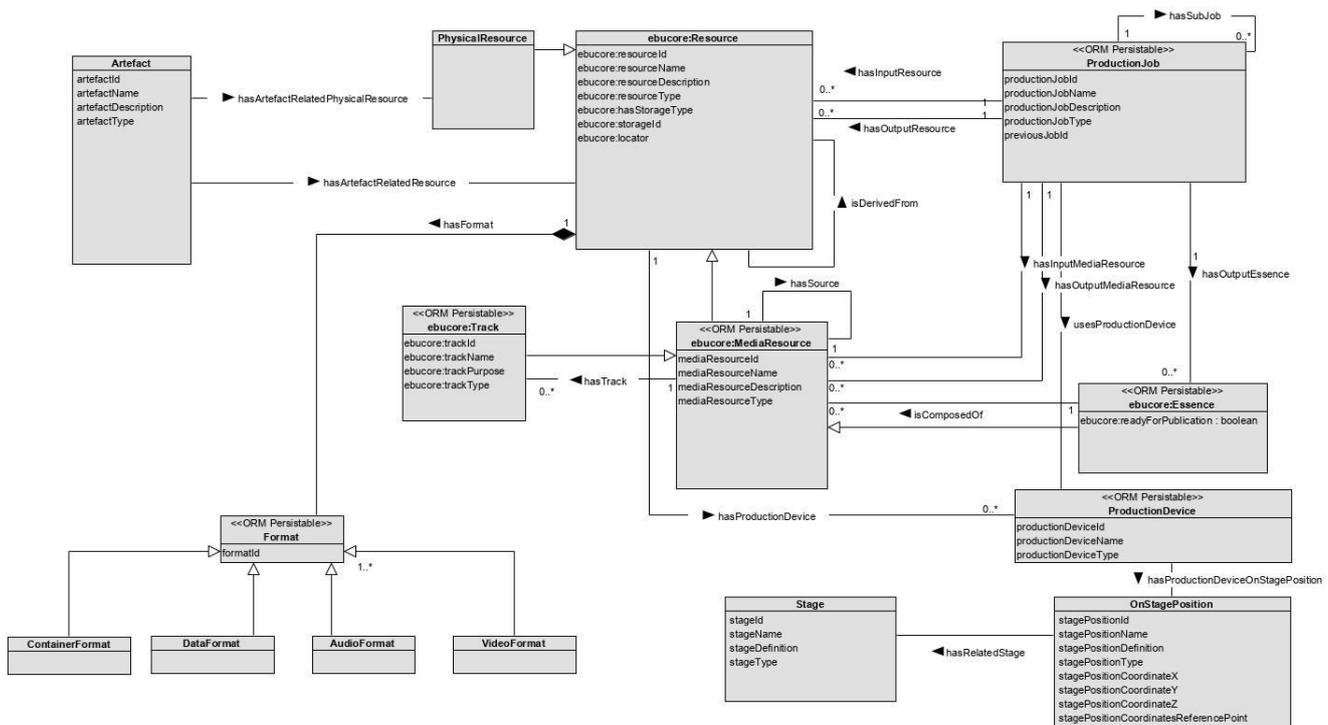


Figure 6: MediaResource

### 2.2.4.1 Resource

Definition:

*Resource* is a generic concept used in relation to a production and going beyond the notions of *MediaResource* or *Essence*. It is defined by an *EditorialObject* (Editorial Domain). It has an associated Locator where the *Resource* can be retrieved.

The class *Resource* is a subclass of *Asset*.

Examples:

A pdf file used as part of the research; a manuscript stored in a repository etc.

Class relations	
hasFormat	E.g. the composition of a <i>Resource</i> . A <i>Resource</i> can exist in one or more <i>Formats</i>
instantiates	Relation to the <i>EditorialObject</i> that describes the <i>Resource</i>
isResourceCommissionedBy	The <i>Contract</i> through which the creation of the <i>Resource</i> has been commissioned
hasProductionDevice	The <i>ProductionDevice</i> used for the creation of the <i>Resource</i>
hasResourceRelatedAuditReport	A link to an <i>AuditReport</i> associated with a <i>Resource</i>
isDerivedFrom	A link to a <i>Resource</i> from which the current <i>Resource</i> has been derived. This information can be used to track provenance
Etc.	Other class relationships can be associated with a <i>Resource</i> . See EBU Tech 3293, EBUCore.
Class Hierarchy	
subclass	<i>Resource</i> is a subclass of <i>Asset</i>
Often used subclasses	
Subclass	<i>MediaResource</i> is a sub-class of <i>Resource</i> , used to specify additional attributes typical for media files.
Subclass	<i>PhysicalResource</i> is a sub-class of <i>Resource</i> , used where the object that instantiates the <i>EditorialObject</i> is a physical thing.
Class Properties	
resourceId	Unique Identifier e.g. a UUID, UMID, URI etc. It can be generated or assigned by the business process or it can be extracted from the content
resourceName	The name given to a <i>Resource</i>
resourceDescription	A description of a <i>Resource</i>
resourceType	The type of <i>Resource</i>
storageId	The identifier of the storage where the <i>Resource</i> has been stored
hasStorageType	A definition of the type / structure of storage where the <i>Resource</i> is stored
locator	This indicates where a particular <i>Resource</i> can be found and accessed
Etc.	Many other properties can be associated to a <i>Resource</i> . See EBU Tech 3293, EBUCore.

#### 2.2.4.2 MediaResource

##### Definition:

“*MediaResource*” is commissioned for production. It is defined by an *EditorialObject* (Editorial Domain). It can be represented by one or more *Essences* e.g. in a particular *Format* for distribution on a specific delivery media. The *MediaResource* is a subclass of *Resource*.

Many properties can be found under the format element of EBUCore for describing the technical metadata of a *MediaResource*

Class relations	
hasSource	The relation to a <i>MediaResource</i> acting as a source of the <i>MediaResource</i> . E.g. an analogue tape that is the source of a file
hasTrack	The relation to the <i>Tracks</i> that the <i>MediaResource</i> are divided into
Etc.	Other class relationships can be associated with a <i>MediaResource</i> . See EBU Tech 3293, EBUCore.
Often used subclasses	
subclass	<i>Track</i> is a sub-Class of <i>MediaResource</i> , used to specify how a file is divided in <i>Tracks</i>
subclass	<i>Essence</i> is a sub-Class of <i>MediaResource</i> , used to specify a <i>MediaResource</i> ready for publication.
Class Hierarchy	
subclass	<i>MediaResource</i> is a subclass of <i>Resource</i>
Class Properties	
mediaResourceId	Unique identifier e.g. a UUID, UMID etc. It can be generated or assigned by the business process or it can be extracted from the content.
mediaResourceName	The name of the <i>MediaResource</i>
mediaResourceDescription	A description of a <i>MediaResource</i>
mediaResourceType	The type of <i>MediaResource</i>
Etc.	Many other properties can be associated with a <i>MediaResource</i> . See EBU Tech 3293, EBUCore.

### 2.2.4.3 Track

Definition:

A *Track* is both a part and a subclass of a *MediaResource*. A *MediaResource* is potentially composed of any combination of audio, video and data *Tracks*.

Examples:

Examples of video *Tracks* are different camera angles or an additional signing *Track*.

Examples of audio *Tracks* are stereo pairs, multichannel audio e.g. surround, international sound, etc.

Examples of data *Tracks*: ancillary data, captioning, etc.

Class relations	
Etc.	Other class relationships can be associated to a <i>Track</i> . See EBU Tech 3293, EBUCore.
Class Hierarchy	
subclass	Track is a subclass of <i>MediaResource</i>
Class properties	
trackId	The identifier attributed to a <i>Track</i> .
trackType	The type of <i>Track</i> .
trackName	A name associated to a <i>Track</i> .
trackPurpose	A short description of what the <i>Track</i> is used for.
Etc.	Many other properties can be associated with a <i>Track</i> . See EBU Tech 3293, EBUCore.

#### 2.2.4.4 Format

##### Definition:

*Format* is a structure of technical metadata. A *Format* can be defined as the composition of audio, video and or data components and the description of their respective *Formats*. The *ContainerFormat* defines the file / package structure of the *MediaResource*. A streaming format can also be defined as a specific *ContainerFormat* for streaming or a custom combination of an *AudioFormat* and *VideoFormat*...

##### Example:

A *Format* for an audio *MediaResource* will define the audio encoding format, the sampling frequency, etc.

Class hierarchy	
subclass	Format is a subclass of skos:Concept
Often used subclasses	
Subclass	<i>AudioFormat</i> is a sub-class of <i>Format</i> , used to list all the characteristics of the audio signal. See e.g. 'audioFormat' in EBU Tech 3293, EBUCore for more information.
Subclass	<i>VideoFormat</i> is a sub-class of <i>Format</i> , used to list all the characteristics of the video signal. See e.g. 'videoFormat' in EBU Tech 3293, EBUCore for more information.
Subclass	<i>DataFormat</i> is a sub-class of <i>Format</i> , used to list all the characteristics of the data signal. See e.g. 'dataFormat' in EBU Tech 3293, EBUCore for more information.
Subclass	<i>ContainerFormat</i> is a sub-class of <i>Format</i> , used to list all the characteristics of the container. It provides information on the container / wrapper format in complement to the stream encoding information provided in 'channel', (e.g. mp3, wave, Quicktime, ogg). See, e.g., 'containerFormat' in EBU Tech 3293, EBUCore for more information.
Subclass	<i>StreamFormat</i> is a sub-class of <i>Format</i> , used to list all the characteristics of a stream.

Class Properties	
formatId	An identifier associated to the <i>Format</i> .
skos:prefLabel	A name associated to the <i>Format</i> .
skos:definition	A definition of the <i>Format</i> .
Etc.	Many other properties can be associated with a <i>Format</i> . See EBU Tech 3293, EBUCore.

**2.2.4.4.1 AudioFormat**

Definition:

A class to provide definitions about the “*AudioFormat*” (e.g. encoding format, sampling rate).

Class relations	
Etc.	Other class relationships can be associated with an <i>AudioFormat</i> . See EBU Tech 3293, EBUCore. This standard defines the Audio Definition Model
Class Properties	
Etc.	Other data properties can be associated with an <i>AudioFormat</i> . See EBU Tech 3293, EBUCore. This standard defines the schema of the Audio Definition Model (ADM).

**2.2.4.4.2 VideoFormat**

Definition:

A class to provide definitions about the “*VideoFormat*” (e.g. encoding format, frame rate).

Class relations	
Etc.	Other class relationships can be associated with a <i>VideoFormat</i> . See EBU Tech 3293, EBUCore.
Class Properties	
Etc.	Other data properties can be associated with a <i>VideoFormat</i> . See EBU Tech 3293, EBUCore.

**2.2.4.4.3 DataFormat**

Definition:

A class to provide definitions about the “*DataFormat*” (e.g. captioning format).

Class relations	
Etc.	Other class relationships can be associated with a <i>DataFormat</i> . See EBU Tech 3293, EBUCore.
Class Properties	
Etc.	Other data properties can be associated with a <i>DataFormat</i> . See EBU Tech 3293, EBUCore.

#### 2.2.4.4.4 *ContainerFormat*

Definition:

A class to provide definitions about the “*ContainerFormat*” (e.g. container type).

Class relations	
Etc.	Other class relationships can be associated with a <i>ContainerFormat</i> . See EBU Tech 3293, EBUCore.
Class Properties	
Etc.	Other data properties can be associated with a <i>ContainerFormat</i> . See EBU Tech 3293, EBUCore.

#### 2.2.4.5 *Essence*

Definition:

The *Essence* is a physical representation of a *MediaResource* in a particular *Format* destined for play-out or publishing. *Essence* is a subclass of a *MediaResource* and inherits the *MediaResource* properties. An *Essence* can be available in a form of a simple file or complex packages (e.g. as delivered by cameras of different brands).

Examples:

An AAC file is an example of audio *Essence*. A P2 file structure (audio, video clip, voice, icon, proxy directories) is an example of package.

Class relations	
isComposedOf	A list of <i>MediaResources</i> that composes the <i>Essence</i> .
Etc.	Other class relationships can be associated with an <i>Essence</i> . See EBU Tech 3293, EBUCore.
Class Properties	
readyForPublication	A flag that is set if the <i>Essence</i> is ready for publication.
Etc.	Many other properties can be associated with an <i>Essence</i> . See EBU Tech 3293, EBUCore.

#### 2.2.4.6 *PhysicalResource*

Definition:

A physical manifestation of the *EditorialObject* it instantiates.

Examples:

This can be a paper document, a book or any other physical object that manifest someone's idea.

Class relations	
Etc.	Other class relationships can be associated with a <i>Resource</i> . See EBU Tech 3293, EBUCore.
Class hierarchy	
superclass	Resource is the superclass for PhysicalResource
Class Properties	
Etc.	Many other properties can be associated to a <i>Resource</i> . See EBU Tech 3293, EBUCore.

**2.2.4.7 Artefact**

Definition:

An object in use, e.g. in a production.

Class relations	
hasArtefactRelatedPhysicalResource	Relation to a <i>PhysicalResource</i> associated with the <i>Artefact</i>
hasArtefactRelatedResource	Relation to a <i>Resource</i> associated with the <i>Artefact</i>
Etc.	Other class relationships can be associated with a <i>Resource</i> . See EBU Tech 3293, EBUCore.
Class Properties	
artefactId	Unique identifier e.g. a UUID, UMID, URI etc. It can be generated or assigned by the business process or it can be extracted from the content.
artefactName	The name given to an <i>Artefact</i> .
artefactDescription	A description of an <i>Artefact</i> .
artefactType	The type of <i>Artefact</i> .
Etc.	Many other properties can be associated to an <i>Artefact</i> . See EBU Tech 3293, EBUCore.

**2.2.4.8 ProductionJob**

Definition:

The “*ProductionJob*” is a process to produce an *Essence* for publication. It uses *MediaResources* as inputs, based on an *EditorialObject* describing the process in detail. It is ordered by a *Contract*.

Where a production is described in several steps, the output can be a *MediaResource* that is not ready for publishing but will be used as input of other *ProductionJobs*.

Class relations	
basedOn	Relation to the <i>EditorialObject</i> that is produced by the <i>ProductionJob</i>
hasSubJob	Relation to a breakdown of the <i>ProductionJob</i> , i.e. a separate task of a workflow
hasInputMediaResource	A list of <i>MediaResources</i> that are used for composing the <i>Essence</i>
hasInputResource	A list of <i>Resources</i> that are used for composing the <i>Essence</i> .
hasOutputMediaResource	Relation to a <i>MediaResource</i> that is the result of the

	<i>ProductionJob</i>
hasOutputResource	Relation to a <i>Resource</i> that is the result of the <i>ProductionJob</i>
hasOutputEssence	Relation to the <i>Essence</i> that is the result of the <i>ProductionJob</i>
hasPJContributor	Information about <i>Agents</i> contributing to the <i>ProductionJob</i>
isOrderedBy	Relation to the <i>Contract</i> through which the <i>ProductionJob</i> is ordered.
hasProductionJobLocation	Relation to the location of the <i>ProductionJob</i> . This can be a studio or another recording Location
hasProductionJobEvent	Relation to the time information associated with the <i>ProductionJob</i> . Can be used for model production plans.
usesProductionDevice	To identify <i>ProductionDevices</i> used for the <i>ProductionJob</i>
Etc.	Other class relationships can be associated with a <i>ProductionJob</i>
<b>Class Properties</b>	
productionJobId	Identifier for the <i>ProductionJob</i>
productionJobName	The name of a <i>ProductionJob</i> .
productionJobdescription	The description of a <i>ProductionJob</i> .
productionJobType	The type of <i>ProductionJob</i> .
previousRelatedProductionJobId	To provide an identifier of a previous <i>ProductionJob</i> . This information can be used to track provenance.
Etc.	Many other properties can be associated with a <i>ProductionJob</i> .

#### 2.2.4.9 ProductionDevice

Definition:

A “*ProductionDevice*” is a device used during a *ProductionJob*.

Example:

An example of a *ProductionDevice* is a tapeless camcorder.

<b>Class relations</b>	
hasUsageContract	Relation to a <i>Contract</i> regulating the usage of the <i>ProductionDevice</i> .
hasProductionDeviceOnStagePosition	To associate a <i>StagePosition</i> with a <i>ProductionDevice</i> .
Etc.	Other class relationships can be associated to a <i>ProductionDevice</i> .
<b>Class Properties</b>	
productionDeviceId	An identifier associated to a <i>ProductionDevice</i> .
productionDeviceType	The type of the <i>ProductionDevice</i> e.g. a camcorder.
productionDeviceName	The name of the <i>ProductionDevice</i> .
productionDeviceDescription	A description of the <i>ProductionDevice</i> .
Etc.	Many other class properties can be associated with a <i>ProductionDevice</i> . Examples of additional properties for a camcorder can be found in EBU Tech 3349 (Acquisition Metadata).

**2.2.4.10 OnStagePosition**

Definition:

A “OnStagePosition” allows to specify the position of a ProductionDevice on Stage.

Class relations	
hasRelatedStage	Relation to a <i>Stage</i> where the <i>ProductionDevice</i> is being used and related to the <i>OnStageLocation</i>
Etc.	Other class relationships can be associated to a <i>OnStagePosition</i>
Class Properties	
onStagePositionId	An identifier associated to a <i>OnStagePosition</i>
onStagePositionType	The type of <i>OnStagePosition</i> e.g. floor/ceiling/wall position
onStagePositionName	The name of the <i>OnStagePosition</i>
onStagePositionDescription	A description of the <i>OnStagePosition</i>
onStagePositionCoordinateX	The x coordinates within a 3 axis spatial coordinates space
onStagePositionCoordinateY	The y coordinates within a 3 axis spatial coordinates space
onStagePositionCoordinateZ	The z coordinates within a 3 axis spatial coordinates space
onStagePositionCoordinatesReferencePoint	The origin of the reference 3 axis spatial coordinates space
Etc.	Other class properties to be associated with a <i>OnStagePosition</i>

**2.2.4.10.1 Stage**

Definition:

The **stage** is a designated space for creating, performing and producing content.

Class relations	
hasStageLocation	Relation to a <i>Location</i> where the <i>Stage</i> is located
Etc.	Other class relationships can be associated to a <i>Stage</i>
Class Properties	
stageId	An identifier associated to a <i>Stage</i> .
stageType	The type of <i>Stage</i> .
stageName	The name of the <i>Stage</i> .
stageDescription	A description of the <i>Stage</i> .
Etc.	Other class properties to be associated with a <i>Stage</i> .

### 2.2.5 Distribution Domain

The Distribution Domain covers any form of publishing, play-out or distribution.

The central class is the *PublicationEvent* that plays out an *Essence*, i.e. the media object that was the result of the *ProductionJob*.

Other classes can be added to suit a specific need in play-out or distribution.

A *PublicationEvent* can be, for example:

- A broadcast event, i.e. an isolated event such as for last minutes news reports, etc. This content can be available via over the air broadcast or streaming.
- A scheduled event, i.e. each event being identified in a particular timeslot. This content can be available via over the air broadcast or streaming.
- An on-demand event, i.e. content is made available for immediate viewing or for download. It generally has a certain window of time availability. Catch-up TV is considered as an on-demand event. On-demand events can also be linked to broadcast and schedule events.
- An on-line event, i.e. content is made available for download/fruition on some web repository (e.g. on a web site)

According to the type of *PublicationEvent*, *MediaResource* is available in different Formats instantiated in *Essence* files or packages.

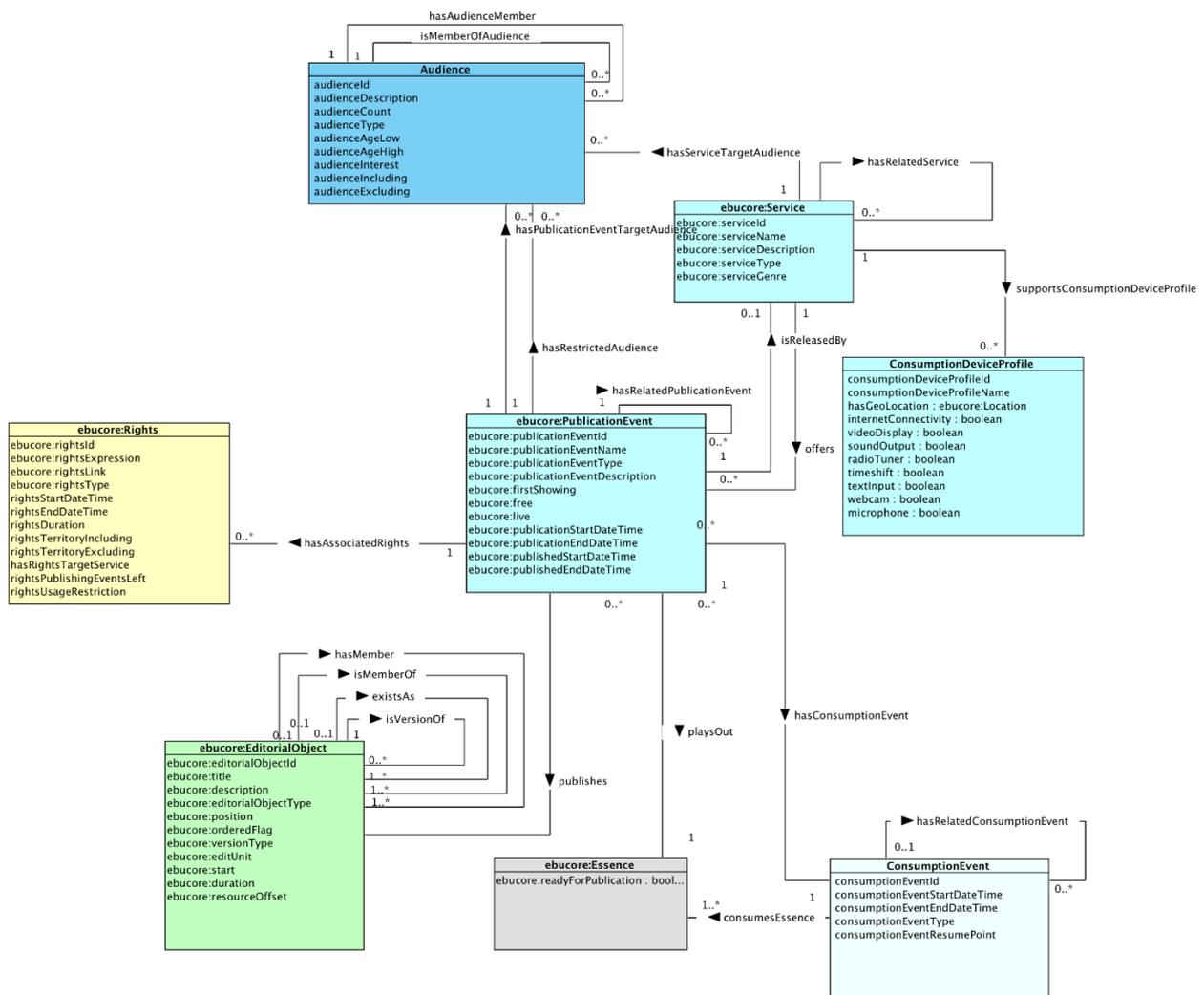


Figure 7: Publication Event

**2.2.5.1 PublicationEvent**

Definition:

The publication of an *EditorialObject* for user consumption is realised by releasing an *Essence*.

Example:

A *PublicationEvent* that is, for example, a scheduled event i.e. a time slot in a schedule associated with a *PublicationChannel*. A *PublicationEvent* can also be a broadcast event not in a preliminary schedule, such as a live special news report. A *PublicationEvent* can also be a streaming event or a VoD publication event.

Class relations	
publishes	A relation to an <i>EditorialObject</i> representing the story that will be published.
playsOut	To allow the ordered publication of a time related sequence of <i>MediaResource</i> / <i>Essence</i> as a <i>TimelineTrack</i> of an <i>EditorialObject</i> .
hasAssociatedRights	To identify the Rights directly associated with a <i>PublicationEvent</i> in addition to inferred rights associated with the related <i>EditorialObjects</i> , <i>MediaResources</i> and/or <i>Essences</i> .
hasRelatedPublicationEvent	To establish a link between two <i>PublicationEvents</i> (e.g. linking an on-demand event triggered from a broadcast event).
hasPublicationEventTargetAudience	The publication targets this particular audience represented by the <i>Audience</i> class.
hasRestrictedAudience	The content is forbidden for this audience.
isReleasedBy	The channel or service platform that releases the content
hasConsumptionEvent	Relation to <i>ConsumptionEvents</i> in relation to a <i>PublicationEvent</i> .
Etc.	Other class relationships can be associated to a <i>PublicationEvent</i> . See e.g. ETSI TS 102 822 (TV-Anytime) or the BBC Programme Ontology.
Class Properties	
publicationEventId	An identifier associated with the <i>PublicationEvent</i> .
publicationEventName	The name of the <i>PublicationEvent</i> .
publicationEventDescription	A description of the <i>PublicationEvent</i> .
publicationStartDateTime	The date and time at which the programme is scheduled to start or when content is made available / can be accessed or consumed.
publishedStartDateTime	The scheduled start date and time of publication.
publicationEndDateTime	The date and time at which the programme is scheduled to end or after which content is no longer available / accessible / consumable.
publishedEndDateTime	The scheduled end date and time of publication.
publicationEventType	The type of the <i>PublicationEvent</i> , e.g. publishing on web or play-out on radio
live	If set, a flag to indicate that the content is "Live".

free	If set, a flag to indicate that content can be accessed / consumed without subscription.
firstShowing	If set, a flag to indicate that this is the first time that this content is available on this <i>PublicationChannel</i> . This is just an indication, the collection of the <i>PublicationEvents</i> one <i>Essence</i> have will tell the real publishing history.
Etc.	Many other properties can be used to define a <i>PublicationEvent</i> . See e.g. ETSI TS 102 822 (TV-Anytime) or the BBC Programme Ontology.

### 2.2.5.2 Service

#### Definition:

A *Service* is a channel or publishing platform that releases the content to a targeted audience through a *ConsumptionDevice*.

Class relations	
hasRelatedService	Relation to some related publishing <i>Service</i> .
Offers	A list of <i>PublicationEvents</i> the <i>Service</i> offers, i.e. like an EPG
hasServiceTargetAudience	The <i>Audience</i> the <i>Service</i> has been designed for
supportsConsumptionDeviceProfile	A list of devices the <i>Service</i> supports, described using instances of the <i>ConsumptionDeviceProfile</i> class
Etc.	Other Class relationships can be associated to a <i>Service</i> . See e.g. ETSI TS 102 822 (TV-Anytime)
Sub-Classes	
PublicationChannel	A specific type of <i>Service</i>
Class Properties	
serviceId	An identifier attributed to the <i>Service</i>
serviceName	The name given to the <i>Service</i>
serviceDescription	A description of the <i>Service</i>
serviceType	Description of the type of <i>Service</i>
serviceGenre	The genre of <i>Service</i> .
Etc.	Many other properties can be used to define a <i>Service</i> .

### 2.2.5.3 ConsumptionDeviceProfile

Describes technical capabilities and requirements of a *ConsumptionDevice* that are needed for accessing a *Service*.

Class relations	
hasGeoLocation	The device is currently within the boundary of a (geo) location. This can assist finding the closest and best CDN service for the device. It might also be used to restrict geo-location access to content.
Etc.	Other class relationships can be associated to a <i>ConsumptionDeviceProfile</i> .

Class Properties	
consumptionDeviceProfileId	An identifier associated with the <i>ConsumptionDeviceProfile</i> .
consumptionDeviceProfileName	A name given to the <i>ConsumptionDeviceProfile</i>
internetConnectivity	The device is capable of accessing the Internet
videoDisplay	The device is capable of displaying video picture frames
soundOutput	The device is capable of outputting sound
radioTuner	The device has a radio tuner
timeshift	The device has a time shift capacity
textInput	The device has a keyboard or another means of text input
webcam	The unit can record video
microphone	The device can record audio
Etc.	Many other properties can be used to define a <i>ConsumptionDeviceProfile</i> .

### 2.2.6 Consumption Domain

In the same way, the Consumption Domain covers aspects of the access and consumption of *Essence*, including any response or *Resonance* this may trigger by the consumer.

The central class in the Consumption Domain is the *ConsumptionEvent*. For linear publishing, this will happen at the same time as the *PublicationEvent*, but for on-line publishing this event will occur one or more times, during the lifecycle of the *PublicationEvent*.

To help adapting the content to the right device and *Consumer*, this domain has a class to describe the *ConsumptionDevice* in detail, but also the *Consumer* via his *Account* information.

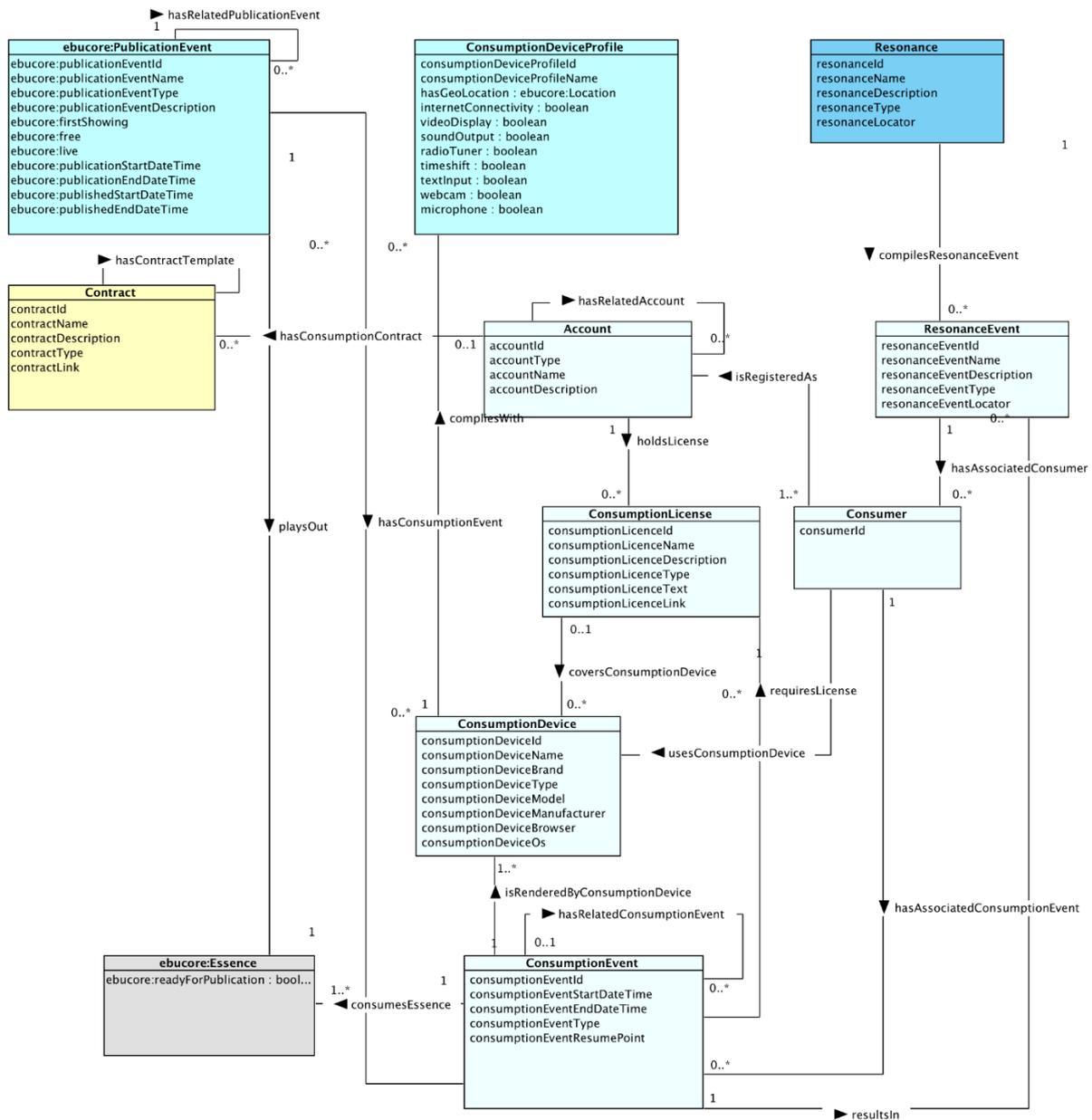


Figure 8: Consumption Domain

### 2.2.6.1 ConsumptionEvent

Definition:

Represents the event of a user consuming a published content.

A *ConsumptionEvent* follows publication *but is no longer related to the PublicationEvent*. The link to the *PublicationEvent* is represented via the *Essence* it consumes.

For linear services the *ConsumptionEvent* and *PublicationEvent* happen at the same time (well, almost, when respecting signal transport and transformation time). For non-linear services, the *Consumer* decides about the time of the *ConsumptionEvent*.

The *ConsumptionEvent* can be followed by a *ResonanceEvent*, if the consumer reacts in a countable or noticeable way.

Example:

- - reading a news article on a public service broadcaster's web site

- - watching a TV programme
- - listening to a radio programme

Class relations	
isRenderedByConsumptionDevice	Relation to the device used as a media render at the moment of consumption
resultsIn	When the user consumes an <i>Essence</i> , different kinds of <i>ResonanceEvents</i> may be generated.
consumesEssence	A relation to the <i>Essence</i> the <i>ConsumptionEvent</i> consumes at least a part of.
requiresLicence	A relation to a licence needed for accessing the content
hasRelatedConsumptionEvent	Used for modelling usage pattern, like first A was consumed, then B and C.
Etc.	Other Class relationships can be associated with a <i>ConsumptionEvent</i> . See e.g. ETSI TS 102 822 (TV-Anytime)
Class Properties	
consumptionEventId	An identifier attributed to the <i>ConsumptionEvent</i>
consumptionStartDateTime	The start date and time of the <i>ConsumptionEvent</i>
consumptionEndDateTime	The end date and time of the <i>ConsumptionEvent</i>
consumptionEventType	The type of <i>ConsumptionEvent</i>
consumptionEventResumePoint	Reflects the resume timing data for a later <i>ConsumptionEvent</i> session on the same <i>Essence</i> .
Etc.	Many other properties can be used to define a <i>ConsumptionEvent</i> . See e.g. ETSI TS 102 822 (TV-Anytime)

### 2.2.6.2 ConsumptionDevice

Definition:

Represents a technical system to access and consume a media service. Its characteristics (seen from a service point of view) are identified into a *ConsumptionDeviceProfile*.

Example:

Examples of *ConsumptionDevices* would be e.g. a mobile phone (including all hardware and software needed for access and consumption), an OTT box together with its TV screen, a TV set with integrated cable tuner, a DAB+ radio.

Class relations	
compliesWith	A list of <i>ConsumptionDeviceProfiles</i> the <i>ConsumptionDevice</i> complies with.
Etc.	Other Class relationships can be associated with a <i>ConsumptionDevice</i> .
Class Properties	
consumptionDeviceId	An identifier associated with the <i>ConsumptionDevice</i>
consumptionDeviceType	The type of <i>ConsumptionDevice</i> in use
consumptionDeviceName	The name the <i>ConsumptionDevice</i> is known under
consumptionDeviceBrand	The brand name of the <i>ConsumptionDevice</i>

consumptionDeviceManufacturer	The name of the manufacturer of the <i>ConsumptionDevice</i>
consumptionDeviceModel	The model of the <i>ConsumptionDevice</i>
consumptionDeviceBrowser	The kind of browser used on the <i>ConsumptionDevice</i>
consumptionDeviceOs	Type of the operating system running on the <i>ConsumptionDevice</i>
Etc.	Many other properties can be used to define a <i>ConsumptionDevice</i> .

### 2.2.6.3 ConsumptionLicence

Definition:

Represents the proof held by a *Consumer* on having the right to experience a *ConsumptionEvent* and consume the published *Essence*.

The *ConsumptionLicence* is verified by a mechanism that is usually located in the *ConsumptionDevice* and referred to as DRM.

Example:

- a document stating the payment of a TV licence fee (this cannot be checked by a DRM mechanism)
- a smart card from a pay TV service containing the necessary information to decode their coded signal

Class relations	
coversConsumptionDevice	The <i>ConsumptionLicence</i> will unlock content for this <i>ConsumptionDevice</i>
Etc.	Other Class relationships can be associated to a <i>ConsumptionLicence</i>
Class Properties	
consumptionLicenceId	An identifier associated with the <i>ConsumptionLicence</i>
consumptionLicenceText	A <i>ConsumptionLicence</i> string that can be verified by the <i>ConsumptionDevice</i> , i.e. DRM
consumptionLicenceName	A name attributed to a <i>ConsumptionLicence</i>
consumptionLicenceDescription	A description of the <i>ConsumptionLicence</i>
consumptionLicenceType	The type of <i>ConsumptionLicence</i>
consumptionLicenceLink	An URL where the <i>ConsumptionLicence</i> is stored
Etc.	Many other properties can be used to define a <i>ConsumptionLicence</i>

### 2.2.6.4 Consumer

Definition:

Represents the individual who consumes the *Service* by using a *ConsumptionDevice*.

The *Consumer* is a member of the *Audience*. He consumes the *ConsumptionEvent* and initiates *ResonanceEvents*. He holds an *Account* and a *ConsumptionLicence*.

Example:

- Every member of a family watching a TV programme, possibly over only one *Account* of the service provider

Class relations	
belongsToAudience	Relation to a list of <i>Audiences</i> the <i>Consumer</i> belongs to.
hasAssociatedConsumptionEvent	A list of <i>ConsumptionEvents</i> that the user has consumed.
isRegisteredAs	Relation to the <i>Account</i> the user is registered as.
usesConsumptionDevice	Relation to the <i>ConsumptionDevice</i> that is used.
Etc.	Other Class relationships can be associated to a <i>Consumer</i> . See e.g. ETSI TS 102 822 (TV-Anytime)
Class Properties	
consumerId	An identifier attributed to a <i>Consumer</i> .
Etc.	Many other properties can be used to define a <i>Consumer</i> .

**2.2.6.5 Account**

Definition:

Represents *Account* information such as login, billing address, banking account, e-mail address ...

Example:

- a social web account of the news department of a public service media
- a person's TV licence fee related account and address
- a simple Id representing an anonymous usage pattern.

Implementers note:

The attribute set can vary and must be added for each of the applications.

Class relations	
holdsLicence	List of <i>ConsumptionLicences</i> the <i>Account</i> holds for their users
hasRelatedAccount	A reference to a related <i>Account</i> , e.g. a family <i>Account</i>
hasConsumptionContract	A relation to the <i>Contract</i> specifying the terms for consumption
Etc.	Other class relationships can be associated to an <i>Account</i> .
Class Properties	
accountId	An identifier attributed to an <i>Account</i> .
Etc.	Many other properties can be used to define an <i>Account</i> .

**2.2.6.6 ResonanceEvent**

Definition:

Represents all individual events that are countable or noticeable reactions by consumers on the *ConsumptionEvent*. E.g. clicks, likes, comments, votes, tweets, preferences, downloads...

All *ResonanceEvents* are linked via the *ConsumptionEvent* to format-related information of an *Essence* and to content-related information of an *EditorialObject*.



### 2.2.7.1 Campaign

Definition:

Represents objects that describe the framework of the *PublicationPlan*. A *Campaign* is an initial plan to release content and the result of the analysis of the *Resonance* data (e.g. likes, downloads, etc). A *Campaign* has a target *Audience* and will usually be associated to a *PublicationPlan*.

Examples could be the desired quantity of *PublicationEvents* (repetition, duration) for a specific *target Audience* and of a specific genre (e.g. sport, news, documentation, commercials) and/or of a specific type, etc. The *PublicationPlan* is supposed to meet this demand and can be checked against it.

*Campaign* is used for advertising and promotional campaigns as well as e.g. overall publication strategies in a public broadcaster.

Class relations	
hasPublicationPlan	A list of <i>PublicationPlans</i> that will help expressing the purpose of the <i>Campaign</i> .
hasInputResonance	A list of <i>Resonance</i> objects that are used as a base for the <i>Campaign</i> .
hasCampaignAudience	The <i>Audience</i> the <i>Campaign</i> targets.
Etc.	Other Class relationships can be associated with a <i>Campaign</i> .
Class Properties	
campaignId	An identifier attributed to a <i>Campaign</i> .
campaignTitle	The title of the <i>Campaign</i> .
campaignDescription	A short description of the <i>Campaign</i> .
campaignType	The type of <i>Campaign</i> .
Etc.	Many other properties can be used to define a <i>Campaign</i> .

### 2.2.7.2 PublicationPlan

Definition:

The *PublicationPlan* class describes a schedule of *PublicationEvents* (and their respective *Audiences*) with references to resulting *ProductionOrders*, and *Assets* (and their *EditorialObjects*). *PublicationPlans* can be related to each other hierarchically, strictly, i.e. membership can only be with one group.

Example:

- A *Campaign* of commercials for a product, is realised with a *PublicationPlan* defining a set of planned *PublicationEvents* using the associated *Assets*.
- A fiction film is promoted with several publications of trailers to a targeted *Audience* and before the publication of the film.

Class relations	
isMemberOfPublicationPlan	A list of <i>PublicationPlans</i> the <i>PublicationPlan</i> is a part of
hasPublicationPlanMember	A list of <i>PublicationPlans</i> that the <i>PublicationPlan</i> contains, which can be used to divide the plan into smaller units
hasAssociatedProductionOrder	A list of <i>ProductionOrders</i> that orders the production of content aimed to be published by the <i>PublicationEvents</i> related to the <i>PublicationPlan</i>
hasBusinessContract	A list of <i>Contracts</i> that are related to <i>PublicationPlan</i>
hasStakeholder	A list of stakeholders that are important to the <i>PublicationPlan</i>
hasPublicationEvent	A list of <i>PublicationEvents</i> that is a part of the <i>PublicationPlan</i>
hasAsset	The assets the <i>PublicationPlan</i> covers
Etc.	Other class relationships can be associated with a <i>PublicationPlan</i>
Class Properties	
publicationPlanId	An identifier associated with the <i>PublicationPlan</i>
publicationPlanName	A name attributed to the <i>PublicationPlan</i>
publicationPlanDescription	A description of the <i>PublicationPlan</i>
PublicationPlanStartDateTime	the start and time date of the <i>PublicationPlan</i>
PublicationPlanEndDateTime	The end and time date of the <i>PublicationPlan</i>
Etc.	Many other properties can be used to define a <i>PublicationPlan</i>

### 2.2.7.3 ProductionOrder

Definition:

The class *ProductionOrder* represents an order for production.

Describes the instance of placing an order with attributes like date, client, contractor, reference to the contract, etc.

Class relations	
hasProductionContract	Relation to a <i>Contract</i> concerning the <i>ProductionOrder</i> .
Etc.	Other class relationships can be associated with a <i>ProductionOrder</i> .
Class Properties	
productionOrderId	An identifier associated with the <i>ProductionOrder</i>
productionOrderName	The name of the <i>ProductionOrder</i>
productionOrderDescription	A description of the <i>ProductionOrder</i> .
productionOrderType	The type of <i>ProductionOrder</i>
Etc.	Many other properties can be used to define a <i>ProductionOrder</i>

### 2.2.7.4 Audience

Definition:

Represents a group of consuming customers/users by number, age, type, interests, etc.

*Audiences* can be related to each other hierarchically. Hierarchy is not strict, i.e. membership can exist with an arbitrary number of groups.

With the hasAudienceMember relation, different *Audience* groups can be linked together to model a more complex *Audience* group. The audienceIncluding, audienceExcluding indicates that the subgroup should be added or excluded from the group that is modelled.

Class relations	
hasAudienceMember	A list of specific <i>Audiences</i> that are used to model a complex <i>Audience</i>
isMemberOfAudience	A list of <i>Audiences</i> this particular <i>Audience</i> is a part of
Etc.	Other class relationships can be associated with an <i>Audience</i>
Class Properties	
audienceld	An identifier attributed to an <i>Audience</i>
audienceDescription	A description of the <i>Audience</i> group covered
audienceCount	The real counted size of the <i>Audience</i>
audienceType	Type of <i>Audience</i>
audienceAgeLow	The lowest age of a member of the <i>Audience</i>
audienceAgeHigh	The highest age of a member of the <i>Audience</i>
audienceInterest	A particular interest common to an <i>Audience</i> group
audienceIncluding	This <i>Audience</i> group part should be included in a composed group
audienceExcluding	This <i>Audience</i> group part should be excluded in a composed group
Etc.	Many other properties can be used to define an <i>Audience</i>

### 2.2.7.5 Resonance

Definition:

Represents the aggregated form (i.e. a non-individual expression) of all countable or noticeable reactions by *Consumers* on the *ConsumptionEvent*.

Examples:

Click rates, number of likes, percentage of votes, number of downloads...

Class relations	
isMeasuredBy	The <i>Agent</i> responsible for compiling and analysing the data into the <i>Resonance</i>
compilesResonanceEvents	One of the <i>ResonanceEvents</i> used as a basis for defining the <i>Resonance</i> .
Etc.	Other Class relationships can be associated to a <i>Resonance</i> .
Class Properties	
resonanceld	An identifier attributed to a <i>Resonance</i>
resonanceName	The name of a <i>Resonance</i>

resonanceDescription	A description of a <i>Resonance</i>
resonanceType	A type of <i>Resonance</i>
resonanceLocator	A locator to the document describing the <i>Resonance</i>
Etc.	Many other properties can be used to define a <i>Resonance</i>

### 2.2.8 Financial Domain

The Financial Domain is the domain, where cost and value of productions are modelled in a very simple fashion. The two classes in the domain can also be used for connecting the CCDM model to a model used for more accurately modelling financial structures, by connecting those two classes to similar classes in the external model.

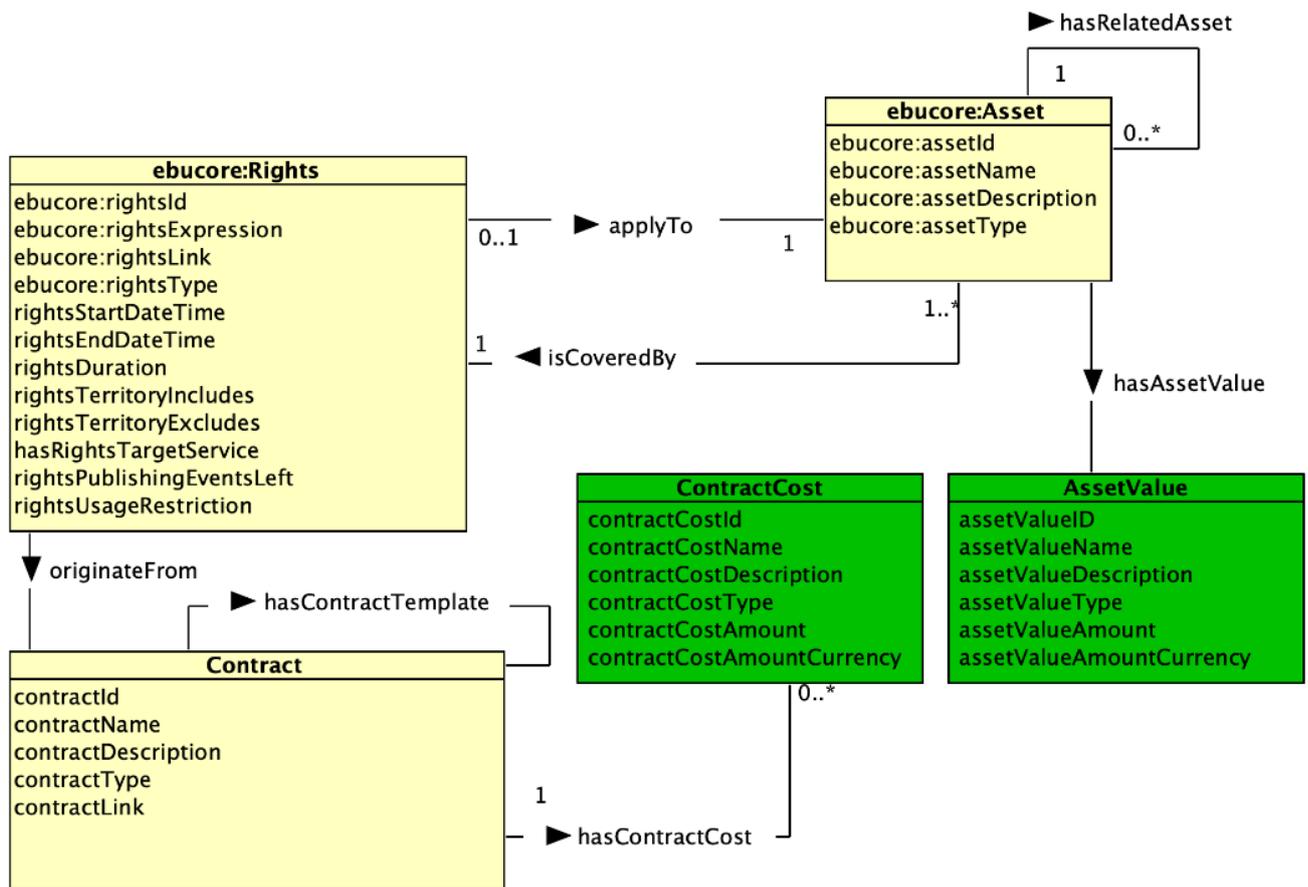


Figure 10: Financial Domain

### 2.2.8.1 AssetValue

Definition:

Represents the value of an Asset. The value can be figurative or abstract.

Class relations	
Etc.	Other Class relationships can be associated to a <i>an AssetValue</i> .
Class Properties	
assetValueId	An identifier attributed to a <i>AssetValue</i> .
assetValueName	The name of a <i>AssetValue</i> .
assetValueDescription	A description of a <i>AssetValue</i> .
assetValueType	A type of <i>AssetValue</i> .
assetValue	The estimated or actual <i>value</i> of an Asset.
assetValueCurrency	The currency in which the value is expressed.
Etc.	Many other properties can be used to define an <i>AssetValue</i> .

### 2.2.8.2 ContractCost

Definition:

Represents the cost of a contractual commitment of any kind.

Class relations	
Etc.	Other Class relationships can be associated to a <i>ContractCost</i> .
Class Properties	
contractCostId	An identifier attributed to a <i>ContractCost</i> .
contractCostName	The name of a <i>ContractCost</i> .
contractCostDescription	A description of a <i>ContractCost</i> .
contractCostType	A type of <i>ContractCost</i> .
contractCostAmount	The actual cost value.
contractCostValueCurrency	The currency in which the cost is expressed.
Etc.	Many other properties can be used to define a <i>ContractCost</i> .

### 2.2.9 Audit and Assessment Domain

This is the Domain where content can be audited. The audit can concern various technical and/or editorial quality aspects of content.

When required, an *AuditJob* is performed to assess the conformance of content against e.g. contractual *Rules* expressed as *Measures*. Results are made available in an *AuditReport*.

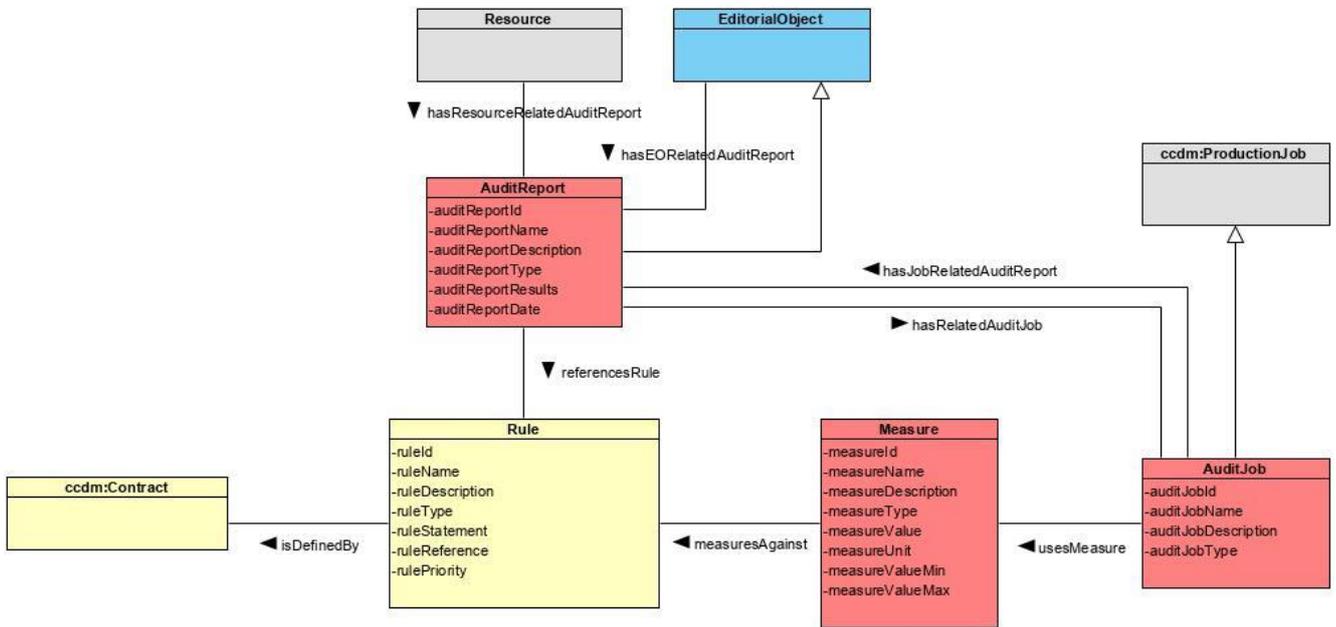


Figure 11: Audit and Assessment Domain

### 2.2.9.1 AuditJob

Definition:

A class to define an *AuditJob* performed to measure and assess (e.g. technical or Editorial) compliance against legal, commercial, regulatory *Rules*.

Class relations	
usesMeasure	A link to a <i>Measure</i> used during an <i>AuditJob</i>
hasJobRelatedAuditReport	A link to an <i>AuditReport</i> associated with the <i>AuditJob</i> and its results
Etc.	Other Class relationships can be associated to a an <i>AuditJob</i>
Class hierarchy	
superclass	<i>ProductionJob</i> is the superclass for <i>AuditJob</i>
Class Properties	
auditJobId	An identifier attributed to an <i>AuditJob</i>
auditJobName	The name of an <i>AuditJob</i>
auditJobDescription	A description of an <i>AuditJob</i>
auditJobType	A type of <i>AuditJob</i>
Etc.	Many other properties can be used to define an <i>AuditJob</i>

### 2.2.9.2 Measure

Definition:

A class to define a *Measure*, which is the measurable transcription of a *Rule*.

Class relations	
measuresAgainstRule	A link to a <i>Rule</i> for which the <i>Measure</i> was defined
Etc.	Other Class relationships can be associated to a <i>Measure</i>
Class Properties	
measureId	An identifier attributed to a <i>Measure</i>
measureName	The name of a <i>Measure</i>
measureDescription	A description of a <i>Measure</i>
measureType	A type of <i>Measure</i>
measureValue	The value of a <i>Measure</i>
measureUnit	The unit associate with the value of a <i>Measure</i>
measureValueMin	The minimum permitted value of a <i>Measure</i>
measureValueMax	The maximu permitted value of a <i>Measure</i>
Etc.	Many other properties can be used to define a <i>Measure</i>

### 2.2.9.3 AuditReport

Definition:

A class to define the *AuditReport* used to publish the results of an *AuditJob*.

Class relations	
referencesRule	A link to a <i>Rule</i> referenced in an <i>AuditReport</i>
isApprovedBy	A link to an <i>Agent</i> that has reviewed and approved the <i>AuditReport</i>
hasRelatedAuditJob	A link to <i>AuditJobs</i> which results have been used in the <i>AuditReport</i>
Etc.	Other Class relationships can be associated to a an <i>AuditReport</i>
Class hierarchy	
superclass	EditorialObject is the superclass for AuditReport
Class Properties	
auditReportId	An identifier attributed to an <i>AuditReport</i>
auditReportName	The name of an <i>AuditReport</i>
auditReportDescription	A description of an <i>AuditReport</i>
auditReportType	A type of AuditReport
auditReportResults	The combined results of one or more <i>AuditJobs</i>
auditReportDate	The date at which the <i>AuditReport</i> has been issued
Etc.	Many other properties can be used to define an <i>AuditReport</i> .

### 3. Implementation Guidelines / Questions & Answers

#### 3.1 General remarks

This section provides examples from current implementers of the EBU CCDM and is intended to provide advice and clarification for users to help them in implementing the EBU CCDM in future versions of the specification.

#### 3.2 Examples provided by SRG SSR, Swiss Confederation

##### 3.2.1 Modelling Different Viewpoints with CCDM

An example of a programme, called “ideal programme”, is shown below:



Figure 12

This example will now be represented using CCDM. The representation depends on the viewpoint, which maps nicely to the domains described in this document. Also, the following examples assume different Publication scenarios, such as “Live” or “Repetition”.

Some examples contain objects that are not directly represented in the graph of the “ideal programme”, for example, the *ProductionDevices* Cam1 and Mic1.

All these assumptions were made only to show the possibilities of modelling with CCDM.

The object graphs represent a hierarchical structure, such as that found in an XML document. To emphasise the hierarchy, it is necessary to introduce “references” (represented as dashed arrows) besides the pure object relation (represented as full arrows) in the hierarchy.

The following diagrams illustrate how to model the “ideal programme” with EBU CCDM.

View from the Editorial Domain

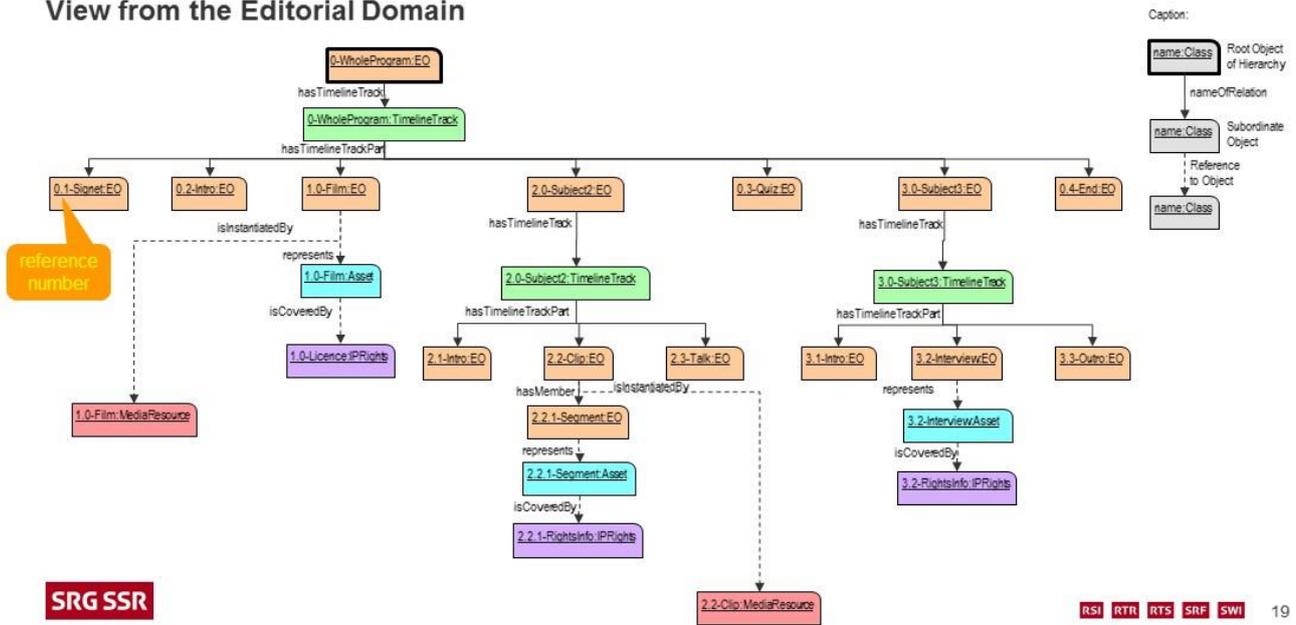


Figure 13

View from the Distribution Domain (Live)

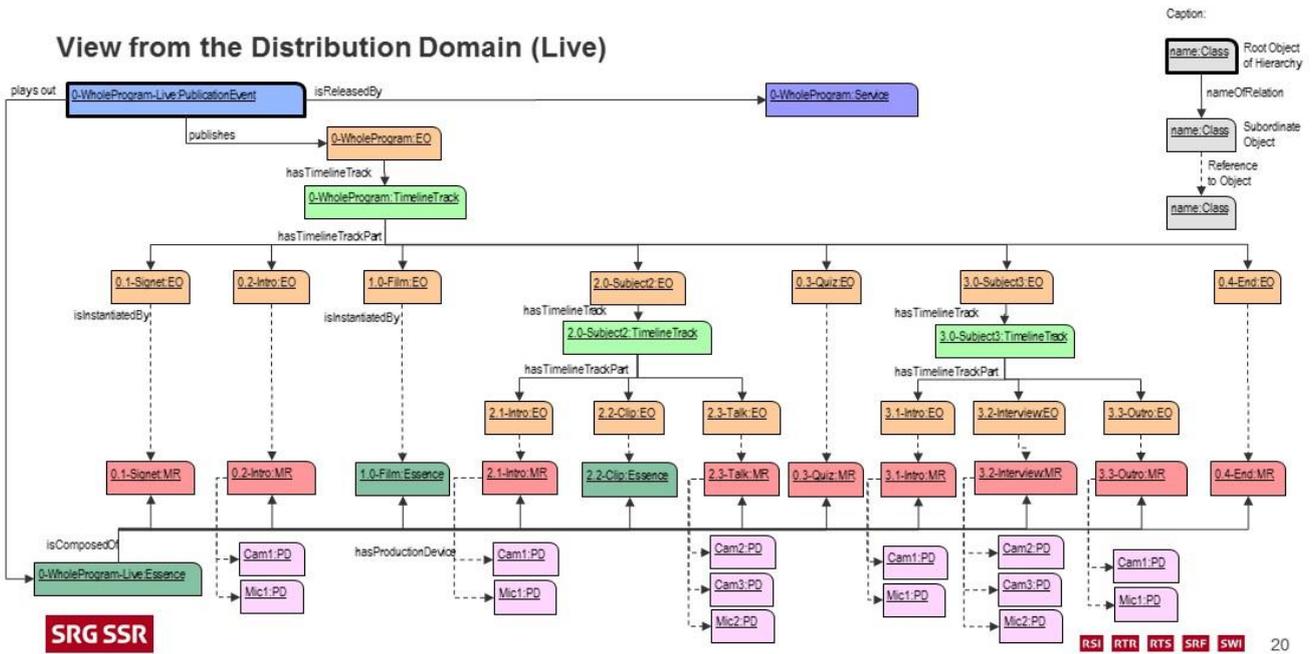


Figure 14

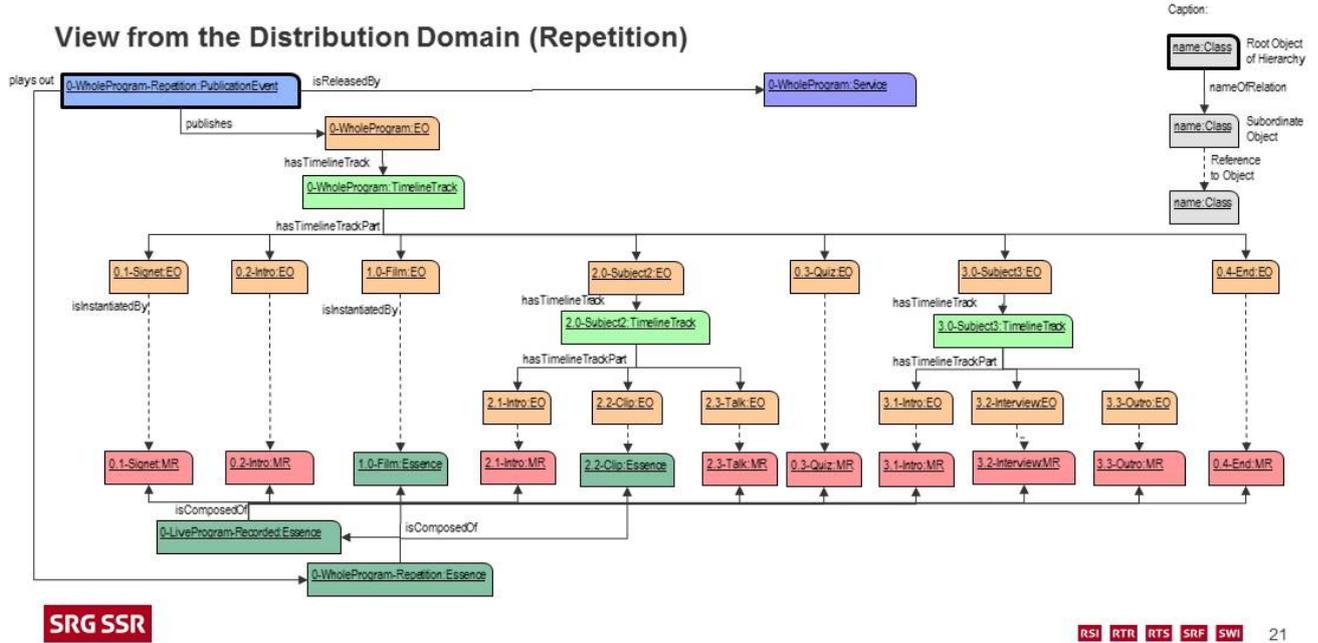


Figure 15

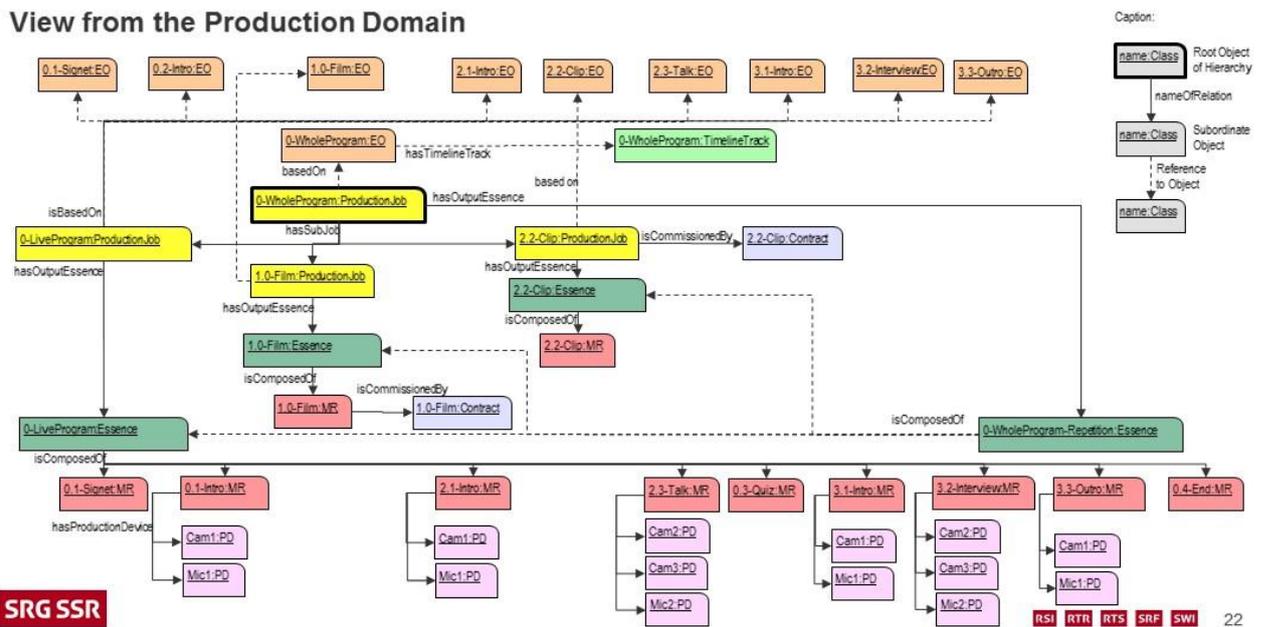


Figure 16

View from the Legal, Commercial and Regulatory Domain

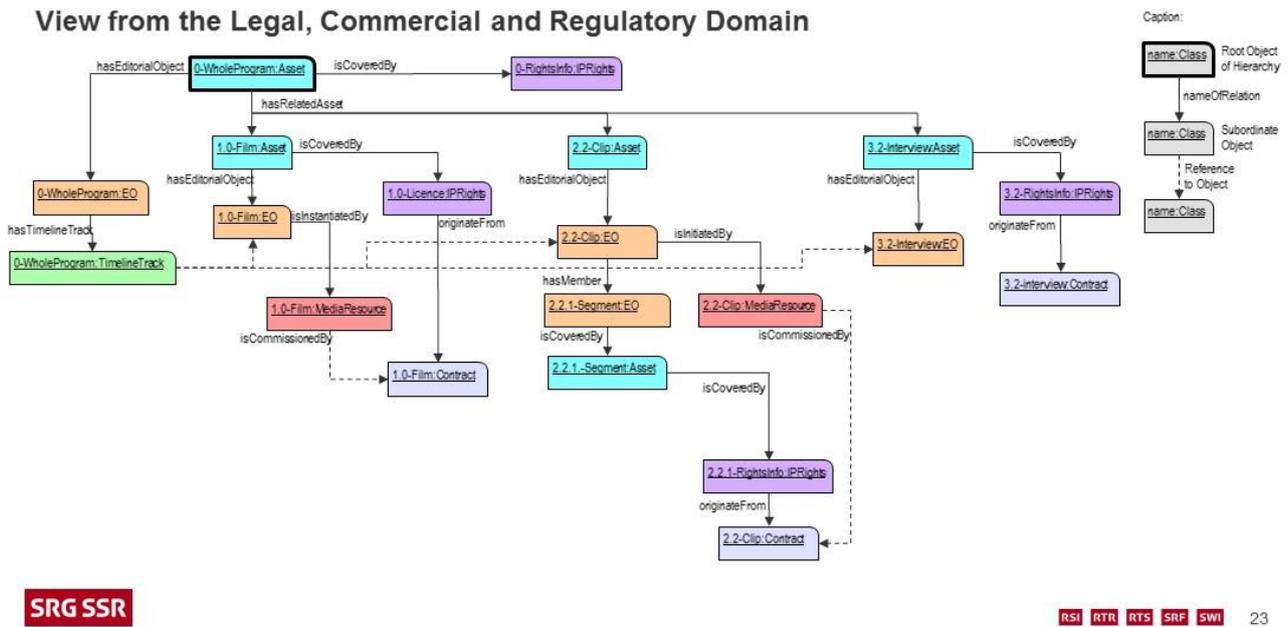


Figure 17

3.2.2 CCDM as a Comprehensive Representation of Business Objects

Business objects (BO), e.g. a business order or its products, carry business value. Managing this value is crucial to the success of an enterprise. Management relies on data, which must comprehensively represent or describe the business objects.

Figure 18 shows how a business object is represented by such data.

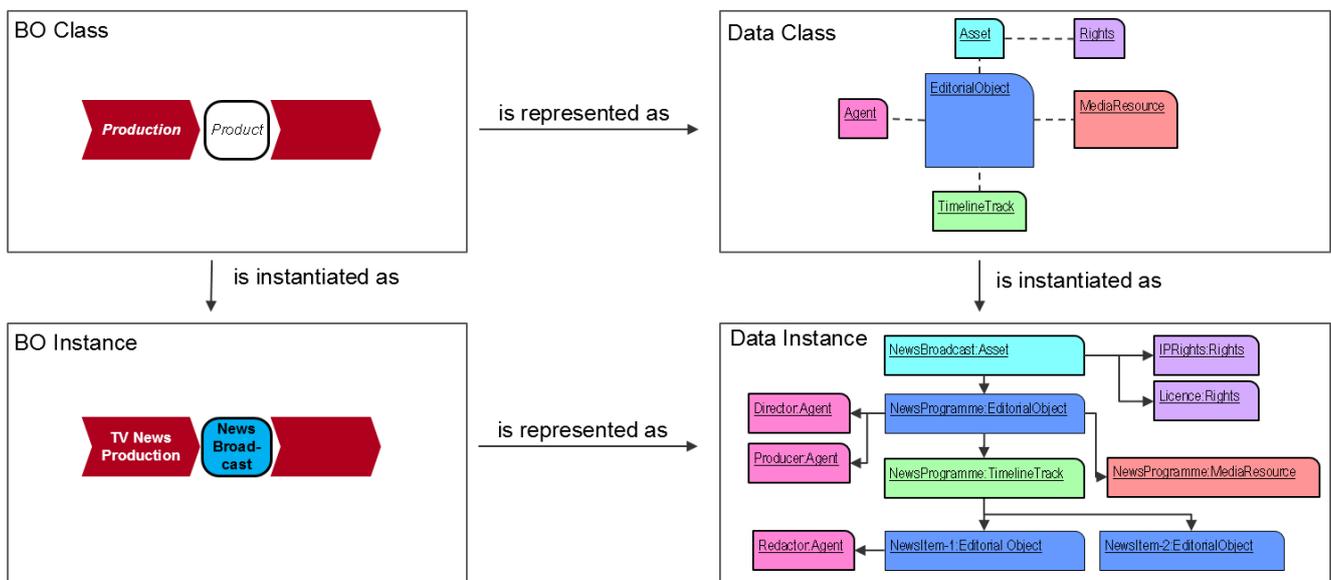


Figure 18: Business objects and associated data

The business object class “Product” is the result of the “Production” process. In real instantiations, this class can take the form of a “News Broadcast” object. A new diagram can be derived from the data. This network of objects is an instance of a generic data class model. The generic class model itself must be designed to represent the business object classes in all required ways.

Consequently, the data model can be evaluated against its ability to represent the largest possible variety of business objects. The EBU has investigated this question and conceived a generic business object and process model for media. The model is a value chain model as shown in Figure 19. It consists of business objects carrying the value, and processes that create value by transforming input objects to (more valuable) output objects.

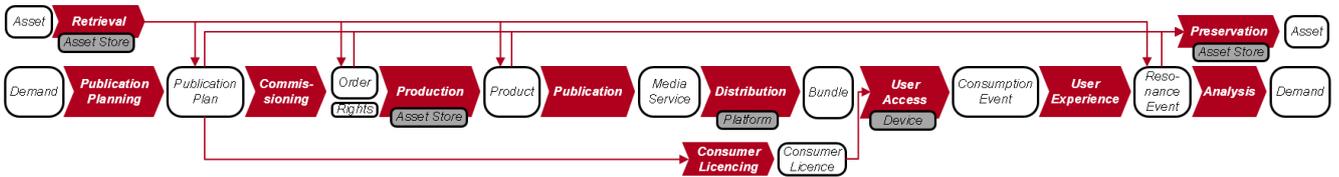


Figure 19: Generic value chain

Every business object in the value chain shown in Figure 10 must be represented by a set of data.

Figure 20 illustrates a simplified example. Check the BO “Rights” and the black line. The Rights can be represented by attributes from different data classes. In this case, from Asset (e.g. ID of the product), Rights (e.g. the permissions, obligations and prohibitions) and Editorial Object (e.g. Title, Duration).

Another example is the BO “Product” and the blue line. A “Product” may be represented by *all* attributes from the classes *within* the blue line and by *some* attributes from classes *touched* by the blue line. The same idea applies for the red line and the BO “Media Service”.

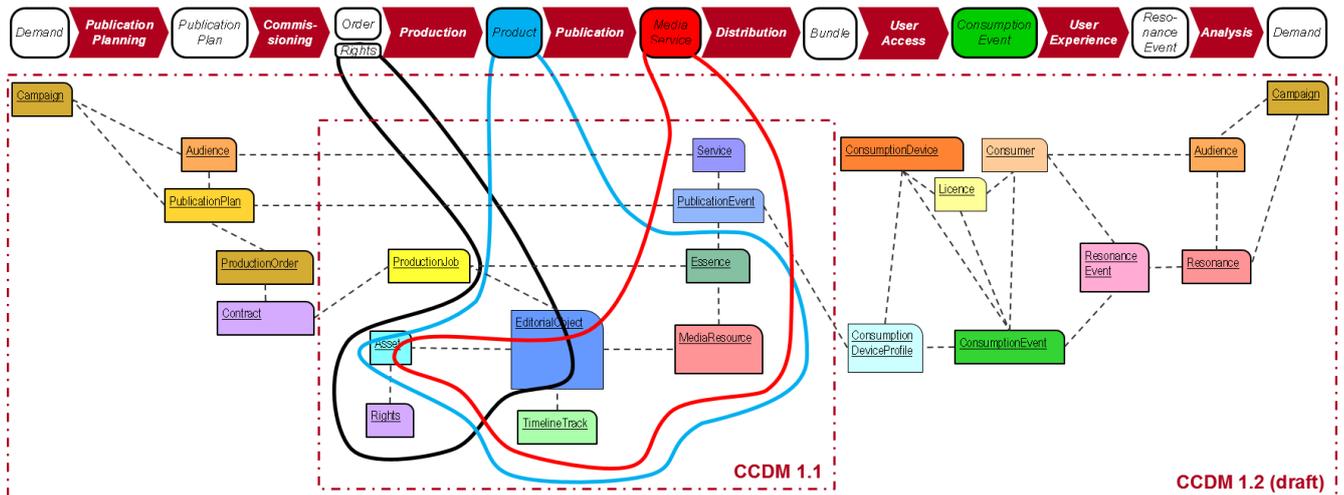


Figure 20: Example of a value chain, business objects and data

This shows that business objects can be represented by a common data model provided by CCDM.

More information on the Modelling Core Business Objects and Processes in Digital Media Enterprises can be found in EBU Tech Report 041 (<https://tech.ebu.ch/publications/tr041>).

### 3.3 Example provided by TV2, Norway

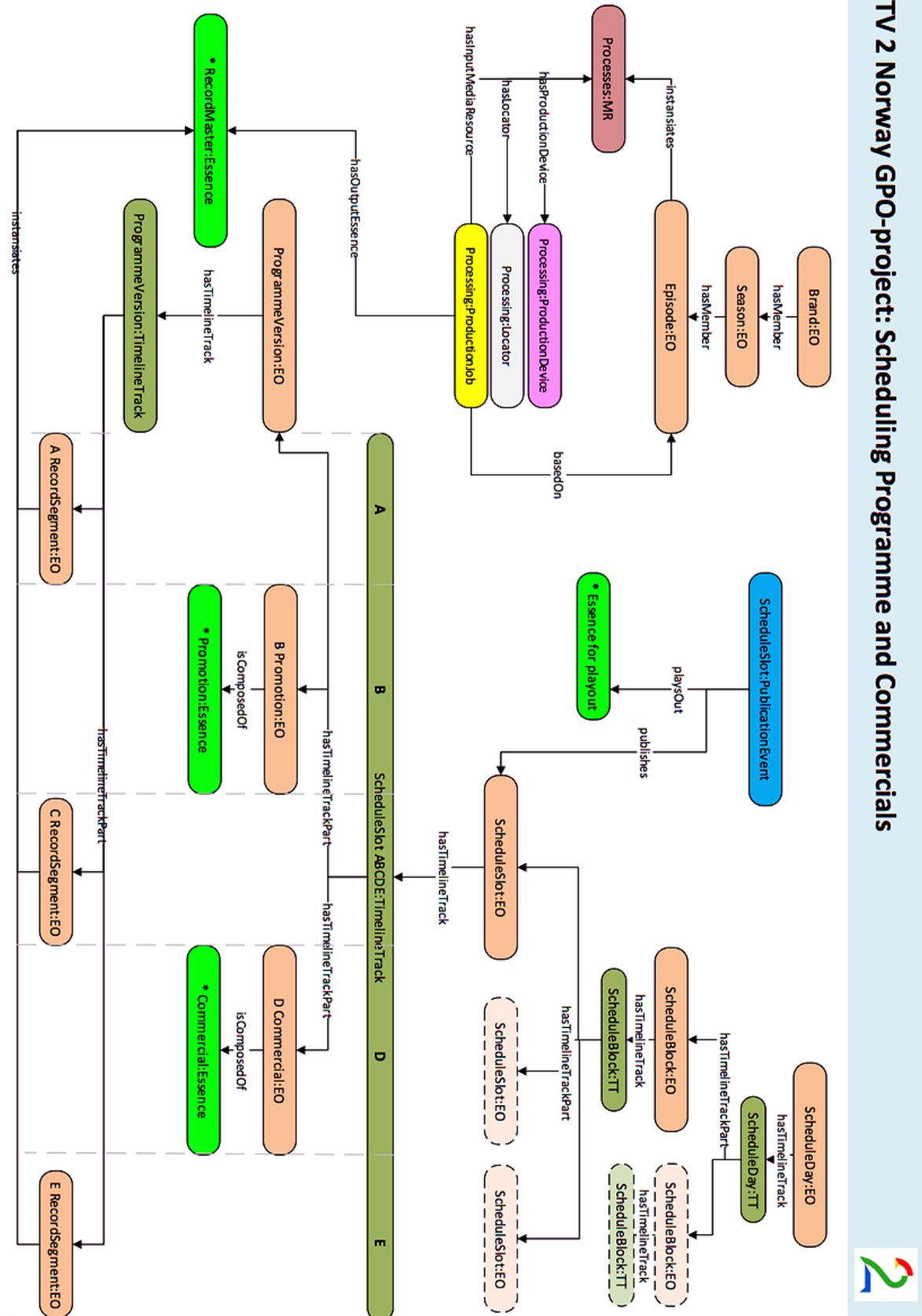


Figure 21: Class diagram from TV2 Norway

### 3.4 The total class diagram

The overall class diagram can be downloaded from [HERE](#).

### 3.5 The RDF ontology

The current specification does purposefully not use specific namespaces or datatypes.

Namespaces and datatypes are defined in `ccdm.rdf`, which definitions prevail over the current specification text.

EBU CCDM RDF ontology is an extension of EBUCore RDF. This hierarchy can be seen in the CCDM RDF file where the EBUCore imports have been made under the `<ebucore>` namespace for both classes and properties. CCDM extensions are under the `<ebuccdm>` namespace.

### 3.6 Further questions?

If you have questions on how to use or implement the EBU CCDM, please forward your queries to [metadata@ebu.ch](mailto:metadata@ebu.ch). You will receive personalised advice, and answers will enrich this section of a future version the specification, with your permission.

## 4. CCDM Compliance

The CCDM is an open framework allowing each user to adapt it to his own needs. As such, the EBU CCDM is flexible and adaptable in nature.

The CCDM ontology is provided as reference software implementation in RDF/OWL. It is available from the "Download Zone". This file contains the minimum set of classes, hierarchies of classes, *objectProperties* and *dataProperties* that compliant implementations should contain, extend, but not replace. More information of the CCDM ontology is provided in **Annex A**.

## 5. Download Zone

Filename and location	Description
<a href="https://www.ebu.ch/metadata/ontologies/ebuccdm/">https://www.ebu.ch/metadata/ontologies/ebuccdm/</a>	RDF documentation
<a href="https://www.ebu.ch/metadata/ontologies/ebuccdm/ebuccdm.rdf">https://www.ebu.ch/metadata/ontologies/ebuccdm/ebuccdm.rdf</a>	RDF / XML file

## 6. Licensing regime

The EBU CCDM is governed by Creative Commons' Attribution-NonCommercial-ShareAlike3.0 Unported (CC BY-NC-SA 3.0)

You are free: to *Share*—to copy, distribute and transmit the work, to *Remix* – to adapt the work, including under your own namespace under the following conditions:

	Attribution - You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).
	Non-commercial - You may not use this work for commercial purposes. Note: this may be used in commercial products but cannot be sold as a specific feature.
	Share Alike - If you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar license to this one.

## 7. Maintenance

The EBU CCDM specification is maintained by the EBU and suggestions for corrections or additions can be made by mailing to ([metadata@ebu.ch](mailto:metadata@ebu.ch) ).

## 8. Useful links

EBU Metadata (<http://tech.ebu.ch/metadata/> )

EBUCore (<http://tech.ebu.ch/publications/tech3293> )

Modelling Core Business Objects and Processes in Digital Media Enterprises  
(<https://tech.ebu.ch/publications/tr041>)

BBC Programmes Ontology (<http://www.bbc.co.uk/ontologies/programmes/2009-09-07.shtml> )

TV-Anytime (<http://www.etsi.org> , Standard download in the TS 102 822 series)

W3C - SKOS (<http://www.w3.org/2004/02/skos/> )

W3C- Resource Description Framework (<http://www.w3.org/TR/rdf-primer/> )

W3C - Web Ontology Language (<http://www.w3.org/TR/owl2-primer/> )

## Annex A: EBU CCDM ontology

The reference software implementation of the CCDM is provided in RDF/OWL.

A link for download is provided in § 5, "Download Zone", of this specification.

There is a variety of options for parsing and editing RDF/OWL documents and ontologies:

Files with an 'rdf' extension can be opened with text processors such as Wordpad;

- Microsoft Notepad can be used;
- More specialised software can be used:
- Protégé (<http://protege.stanford.edu/download/download.html> ) (recommended for beginners) - Note: the .rdf extension may need to be changed into .owl
- TopBraid Composer, free edition ([http://www.topquadrant.com/products/TB\\_Composer.html](http://www.topquadrant.com/products/TB_Composer.html) )