ONTOLOGY FOR MEDIA CREATION
PART 1: OVERVIEW

VERSION 2.6

Motion Picture Laboratories December 16, 2024

Contents

1 Need for an Ontology

As defined in *The Evolution of Production Workflows*, Software-Defined Workflows are a key component of the MovieLabs 2030 vision. The principal goal of a Software-Defined Workflow is to remove friction and inefficiency from the production process by better supporting human collaboration and by automating tasks that are currently manual, ad hoc, or custom per production.

The tasks in a software-defined workflow must be able to communicate securely and unambiguously. Securing this communication is covered in the MovieLabs Common Security Architecture for Production documentation. At the level of the software components themselves, removing ambiguity requires common data models and data structures. However, these models and structures are only useful if they represent what goes on in the production process. Production is a hugely complex endeavour, different parts of it have their own sets of ideas and words for those ideas, and often the same word means different things in different circumstances. This ambiguity is sometimes OK in human communication (especially within a particular segment of the process) but is actively detrimental when building software – it creates friction and uncertainty.

This challenge is not new: if there were not ways to deal with it, people could not make movies. At the human level, it is managed by hoping for a shared (and usually implicit) understanding of words and processes. But sometimes this communication only iteratively converges on a mutual understanding. And sometimes outright miscommunications occur. One goal of our ontology work is to try to improve the efficiency of this human communication, whether in verbal or visual communication about workflows.

At the software level, variations in terms and their meanings are more problematic. People attempt to deal with it by writing an endless series of adapters and shims that match one application or department's worldview to way that other applications or departments model the problem. This is a perpetual and increasing burden – there are always more applications and new processes or technologies to integrate – and it is an opportunity cost as well since software developers spend time knitting applications together rather than developing new products and features.

Why is this worth fixing? The increased efficiency enabled by a software-defined workflow allows more time for the actual creative processes. For automation to succeed, there must be common data models that are understood across the many components of the production workflow. Shared data models reduce errors caused by re-entering information and multiple translation steps, support a transition from manual communication to automated communication, and greatly simplify integrating new applications and processes.

It is clear that there are two parts to this problem. First, there has to be broad agreement on the production processes commonly in use, which of those are important, and how people perform them: what they do, what they use, and how they communicate about it. This is difficult because no two productions are the same, but it is not impossible.[1] The second part is significantly easier: once we know what matters, we can develop shared vocabulary and shared data models to represent it.

Both of these can be addressed by **an Ontology**. An Ontology defines a set of "things," their properties, and the relationships between them, all within a particular area, scope, or domain. In practice, an ontology *defines* all *essential* objects and how they *relate* to each other. All three of those italicized words are important. Without shared definitions, a system and the people who use it fall into the Humpty Dumpty trap of arbitrariness.[2] If something essential is missing, there is no way to communicate about it or use it.[3] Finally, every part of the production process is connected to something – the script, the previous and subsequent steps in the workflow, the participants involved, etc. – and each component acquires additional relationships over the course of the production.

The ontology becomes the language of building repeatable workflows; once it can be articulated, it can be constructed. An ontology is necessary but is not, of course, sufficient. A visual language requires iconography and labels. And a software-defined workflow needs many other things, such as reliable ways to communicate the standardized data and a common security architecture. However, the real value to the production comes not from how something is communicated but what is communicated. The ontology and its data models allow more people and systems to understand that information.

An important concept in the 2030 Vision is "appropriate granularity," meaning most things can be composed into broader or higher-level things or decomposed into more detailed things.[4] Some examples are constructing a full 3D model of a character out of geometry, textures, and rigging; and dividing an editing task into a set of subcomponents. This principle is built into definitions and data structures in the ontology, where these units are generally referred to as Groups.

The principle of appropriate granularity also helps with the general problem of appropriate scope, which often afflicts ontologists. If the subject is large or complicated, there are two risks. If we define too much, the ontology will be too constraining, meaning that the ontology may not work well with real-world aspects of the original problem space.[5] If we define too little, there is the potential for ambiguity in the definition of a "thing" or its relationships, and therefore insufficient precision for automation.

The production process can be vast, complicated, and highly variable. Producing a hyper-detailed ontology may exclude some useful and reasonable ways of working on a production (or delay its release

[1] Each production is a snowflake, but snowflakes are made of frozen water; the Ontology is the water.

[2] "When I use a word," Humpty Dumpty said, in rather a scornful tone, "it means just what I choose it to mean—neither more nor less."
"The question is," said Alice, "whether you can make words mean so many different things."
"The question is," said Humpty Dumpty, "which is to be master—that's all."
- Lewis Carroll, Through the Looking Glass

[3] "Wovon man nicht sprechen kann, darüber muss man schweigen." (Whereof one cannot speak, thereof one must be silent.) – L Wittgenstein, *Tractatus Logico-Philosophicus*

[4] This principle is innately part of two industry standards already – EIDR and MDDF.

[5] It also runs the risk of never being finished.

so far that the industry proceeds without it, continuing a path of fragmentation); the result of an overly abstract ontology is that some of the standardization needed for software-defined workflows is missing.

We address both issues – appropriate granularity and appropriate scope – by using a technique called *connected ontologies.* Common and foundational concepts, structures, and relationships are defined precisely in a base ontology, with obvious places to connect to more granular ontologies for specific components of the production process. This compartmentalization brings many technical benefits and allows connected but separate parts of the workflow to develop at their own pace.

Individual components of the ontology may have their own initial scope restrictions, much as the base ontology does. Over time, the ontology will add more connected ontologies as it is used in more specialized areas, and some connected ontologies will require adjustments in the base ontology.

For example, Visual Effects (VFX) work at the highest level uses a relatively well understood (but non-standardized) way of doing things. These details (VFX Sequences, VFX Plates, groupings of required assets, terms and concepts for the tasks that communicate with the broader workflow, etc.) are part of the basic ontology, allowing any VFX process to connect to the workflow in a standard way. There is certainly some commonality across the multiple implementations of individual VFX workflows, and these details belong in a connected ontology, which specific implementations can extend as necessary to meet their needs. Too much detail in this will prevent some VFX processes from using it. Further, suppose an entirely new way of managing a VFX pipeline arises. In that case, that way of working can be a separate connected ontology, the use of which will not disrupt any of the higher-level mechanisms, components outside the VFX workflow, or existing VFX workflows. Defining “too much” VFX-specific material in the base ontology overly constrains future developments and may impede integration with the current universe of VFX processes.[6]

As a different kind of example, the Slate is used in multiple places in the production process. A software-defined workflow will use the same representation of a Slate across all its components, rather than using different ones for different pieces as is the case now – or even, in some cases, requiring someone to view the footage that has the image of the on-set slate. If the Slate is not adequately defined, instances of it will not be useful for automation across the whole production process.

Having explained what an ontology is, what it helps with, and how we are building it, the question remains: “What can I do with it?” As we continue to flesh out the data models and definitions, we will be working with the industry to enable new system implementations that adopt it over time. The more immediate use is as a common communication mechanism among applications and services that know similar things (scenes, actors, shots, sequences, etc.) but don’t know how to communicate those things to each other and therefore can’t work together. We expect that using the Ontology as a translation

[6] Although when developing such an ontology, it is usually initially necessary to go into what turns out to be extraneous detail. “The road of excess leads to the palace of wisdom... You never know what is enough until you know what is more than enough.” – Wm. Blake, *The Marriage of Heaven and Hell*

mechanism (using plug-ins, shims, and other technical means) will be the focus for most adopters for some time.[7]

[7] Not quite a Babel fish (see https://en.wikipedia.org/wiki/Babel_fish), but close.

2 What this Ontology Covers

The Ontology is structured and documented by category; however, some things will fall into one or more categories; a Script, for example, is its own “thing” in the ontological sense, *and* it is also a document or an asset that can be treated as an object in a data storage system or a filing cabinet. We have attempted to organize by conceptual similarity or relatedness rather than by implementation similarity.[8]

The current categories are as follows. The formal definitions of what they include are taken from the respective Concept documents.

- *Creative Work*: A uniquely identified production.
  This is currently covered in Context but will be separated with links to a revised MovieLabs Creative Works Ontology.[9]
- *Context*: Informs scope within the construction process of a Creative Work.
- *Asset*: A physical or digital object or collection of objects specific to the creation of the Creative Work.
  The top-level discusses the concepts of Asset, Structural Asset, and Functional Asset. Individual groups of Assets are covered in separately published appendices. Current (2.5) ones cover:
  - Camera Metadata
  - Asset Versions
  - Audio
  - CG Assets
  - Images

  Functional Assets vary in their level of granularity, and this documentation provides a set that is either generally useful or extremely common in specific workflows.
  Wherever possible, the ontology references external technical specifications standards, including ones from SMPTE, USD, the Academy Software Foundation, as well as other industry practices.
- *Participant*: The people, organizations, or services that are responsible for the production of the Creative Work.
- *Task*: A piece of work to be done and completed as a step in the production process
- *Infrastructure*: The underlying systems and framework required for Creative Work production are generally not specific to a particular Creative Work.
- *Relationships*: Describes and defines the connections between Assets, Tasks, Participants, Contexts, etc. It covers:
  - Connections between top-level concepts

---

[8] However, “…notoriamente no hay clasificación del universo que no sea arbitraria y conjetural.” (notoriously, there is no classification of the universe that is not arbitrary and conjectural) – J L Borges *El Idioma Analítico de John Wilkins.* In the same essay, see the categorization of animals from “a certain Chinese encyclopedia entitled *Celestial Emporium of Benevolent Knowledge.”*

[9] The current version is https://movielabs.com/creative-works-ontology/.

    - Composition and decomposition of top-level concepts, e.g., between a composite Asset and its components

  More detailed relationships (for example, the connection between an EDL and an editing Task) are generated by individual sub-projects and covered in the concept or schema documents that define them. The Relationship dictionary includes all Relationships, no matter where they are defined.
- *Utilities*: covers common concepts and basic technical requirements
    - Miscellaneous: Common concepts used when constructing a wide variety of other concepts, e.g., Identifier, Description, and Composition..
    - Units of Measure: These are a special type of Utility class and are usually defined by reference to other formal standards, e.g., SMPTE and ISO.

2.1 How to Use this Ontology

This Ontology is not a schema. To summarize, it is a data model and a set of defined terms for concepts in the film and television production process. Software that uses the model needs a schema, and we provide schemas in two languages: RDF and JSON. The two schemas have some differences, for instance the use of composition in JSON and inheritance in RDF, but both implement the common conceptual model defined in these documents.

As mentioned above, the Ontology can never cover everything that any application might want or need. We strongly encourage the use of the schemas as a starting point for implementations. Both the RDF and JSON schemas can be expanded and extended as needed without damaging interoperability of the basic concepts, attributes, terms, and relationships. We add new features to the ontology and the schemas based on industry feedback. In the past, some of this feedback has been requests for new attributes for existing concepts, and some has resulted in new work as a mix of new concepts and the addition of connected ontologies.

Some examples of using and extending the ontology and the schema include[10]:

- Using OMC JSON to exchange information between components of productin workflows.
- Using the RDF schema to extend an existing linked data-based asset manager. The ontology's model is preserved and extended, and the triplestore-based application can import ontology data in both RDF and JSON, since they both implement the same conceptual model.
- Using the JSON schema as a basis for a GraphQL schema that results in ontology-compliant query results from a noSQL database, the internal schema of which implements the ontology's model with the JSON schema as a starting point. Interoperability is preserved, even though the database schema is different.

[10] All of these have been implemented.

- Implementing the Ontology in a graph database where the nodes and edges implement the concepts, attributes, and relationships defined by the Ontology. A front end allows import of data expressed using the JSON schema. In this case, the implementation is independent of either of the current schemas, and interoperability is supported via import and export in JSON.
- Using the Ontology's defined terms as the basis for a visual language for software-defined workflows. The visual language includes details for things that are not included in the Ontology, but for everything that is in the Ontology it uses the Ontology's terms and definitions. Some of these details have been proposed as additions to the Ontology.

In all of these cases, it is the Ontology and its definitions that underpin interoperability, with the schemas supporting individual implementations.

3 Technical Notes

The things defined in each section of the Ontology have many common features with each other. There are also common models and idioms across the various sections.

Identifiers

Most items have one or more Identifiers. An identifier is assumed to be unique within its particular scope of use. An identifier refers to an instance of the item, not to its type, definition, or class. Identifiers are often resolvable – using the identifier, an external system, such as a database can provide information about the item, including metadata and any underlying data (such as a video file). If an item is used or stored outside the context in which it was created, it should have an identifier so other applications or systems can consistently track, manage, and use it.

Custom Data

The Ontology is a framework for developing software-defined workflows, and as mentioned above, does not and cannot provide structure and definitions for everything someone might possibly want to know about an item. Besides the standard data, which is used for automated communication, most classes have a Custom Data field for information that cannot be expressed in the Ontology or needs to be available in a particular format.

For example, if an application communicates Camera Metadata in the Ontology-defined form, it may still want to send the original, less standard data; in that case, it can put the original information in the Custom Data field of the standard Camera Metadata. An application that uses these extra data fields has to know or assume that another application that reads those fields knows how to interpret them.

Since OMC uses linked identifiers, it is also possible to just include a reference to the Custom Data without including all of it in the entity to which it pertains. As an example, if a CG workflow has application specific data, it can include a named relationship to the Identifier of the external data. As long as the receiving application can resolve that Identifier, that's all the originator needs to communicate.

Structural and Functional

Many of the concepts used in production have two aspects. The first is intrinsic to an instance[11] of the concept, which the Ontology calls its Structural Class: a digital image is always a digital image, and a person is always a person.[12] However, those fixed things can be used in many ways: a digital image might be a piece of reference art in pre-production, and a texture for a computer graphics system later in the production, and a person can be an actor and a director for the same Creative Work. The Ontology calls this the item's Functional Class. This also allows a single instance of a Functional Class, e.g., a Script, to

---

[11] An instance is what happens when a concept is made into an actual thing. For example, Chico Marx is an instance of the concept of Person, a photograph of Harpo Marx is an instance of an Image, Rufus T. Firefly is an instance of the concept of Character, and *Duck Soup* (1933) is an instance of a Creative Work.

[12] "Peoples is peoples. No is buildings." – Pete, in *The Muppets Take Manhattan*.

be used with different structural classes, e.g., a paper document and a PDF, each of which is still in some sense “the script.” Note that this use of the term “structural” should not be confused with its use to refer to data structures and the organization of metadata.

Files

Note: OMC uses the word “file” for data outside the Ontology itself, whether that is essence data or blobs of non-OMC metadata. Historically, this has been stored in a file system as noted above. For simplicity OMC uses the word “file” to encompass other storage as well, such as cloud storage and networked storage.

4 Documentation and Other Artifacts

There are six kinds of documentation for the Ontology. All of these have version numbers. A change in the major version number indicates that some incompatible changes have been made; a change in the minor version number indicates extensions or corrections but no incompatible changes.

1. **Concept-oriented documents:** These define sets of terms and concepts and include links to the appropriate Ontology and Schema documentation. They cover the conceptual (as opposed to technical) model of the Ontology and are a good starting point for people who want to understand how the Ontology is structured. They give examples and explanations and include:
    - Formal definitions of terms and concepts (which are also included in the Dictionary)
    - Some sample attributes of the items are defined for those terms when they clarify the definitions or clarify how the items are used. The lists of attributes are not complete and contain only things that are essential to being able to use the items or that themselves require formal terms and definitions.
2. **Formal ontology documents (RDF):** These provide an RDF schema for the terms, concepts, and attributes given in the concept-oriented documents. The schema also includes more formally defined relationships than the overview documents, which are mirrored in the other schemas. These are intended for people who want to understand the detailed technical data model or want to implement it themselves either as RDF or in a different schema or modeling language. The RDF is also useful for generating diagrams of the data model.
3. **Implementation-oriented Schema documents (JSON):** These define a JSON schema starting with the terms and concepts, with basic attributes that add enough detail to allow a practical, useful implementation. The details include, for example, ways of dealing with inclusion vs. reference for complex attributes, full sets of required or common attributes, and an extension mechanism. They are direct implementations of the formal Ontology.

    The JSON schema comes with a full set of documentation and software that validates OMC JSON documents against the schema. The validator covers syntactic correctness and some aspects of recommended practices.
4. **A Dictionary** of terms, concepts, and vocabulary, and common synonyms. This is available in three forms: text (e.g., a web page), spreadsheet, and SKOS. There is a separate dictionary for relationships. This is primarily intended as a quick reference for people and as a convenient packaging of terms and definitions for applications that may need them, including various kinds of user interfaces.
5. **Mapping Tables**: Some areas of the Ontology are difficult primarily because many defined terms must be mapped to other sets of terms for the same things. Where the number of mappings is small (e.g., for the various regional variants in the meaning of “series”), these mappings are included in the Dictionary. Where the domain is large, e.g., for Camera Metadata, the mappings are provided in a separate spreadsheet for ease of use.
6. **File naming documents:** File naming, e.g., for sound files or VFX plates, is currently ad hoc, and these provide naming conventions and practices for some current file-based workflows. Where applicable, these refer to concepts and terms from the concept-oriented documents.

Separately, we have defined a visual language for representing software-defined workflows using the Ontology. The Ontology supports the visual language by providing precision to the elements that can be depicted using it. Everything in the visual language maps onto to one or more elements in the Ontology.

4.1 Notational Conventions

In documents generally:

- The definition of a term included in the Dictionary is in bold, followed by the definition, e.g., **Creative Work:** A uniquely identified production.
- When a defined term is used in the text of a document, it is capitalized, for example in "The Production Scene is usually derived from a numbered scene in the Script," Production Scene and Script are defined in the Ontology. (Note, a word that is part of defined term may sometimes be capitalized by itself as a shorthand, e.g., "Scene" may be used to indicate "Narrative or Production Scene.")
- References to other Ontology Documents are in ***bold italic***, e.g., ***Part 3: Assets*** or ***Part 3A: Camera Metadata***

For Sample Attributes in the concept documents:

- If a data field or attribute is formally defined in this ontology or a connected ontology, it is italicized, e.g., *Setup* as an attribute refers to a defined concept.
- Attribute […] indicates an attribute can appear more than once, e.g., *Identifier* […]
- →Thing means that an attribute is expressed as a relationship to a Thing, e.g., the →*Script* attribute of Creative Work means there is a relationship Creative Work→*Script*
- A combination of the two indicates that the concept can have relationships to a set of things, e.g., →Components […]
- Many elements of the Ontology have a Context element. (See **Part 2: Context**.) Relationships declared in the Context are implied to have the item to which the Context is attached as their starting point, for example, Narrative Location→Context→Narrative Scene.

  Contextual relationships that are especially important to the concept being defined are given in the sample attributes tables as C→Thing or C→Thing […] as appropriate. These relationships can just as well be on the object that has the Context. For example, if Narrative Location has "C→Narrative Scene" as an attribute, it is ok to have the relationship directly on the Narrative Location or in its Context, e.g. Narrative Location→Narrative Scene or Narrative Location→Context→Narrative Scene.

  Some implementations (e.g. RDF) place these relationships directly on the class as well as allowing them in Context, and others (e.g. JSON) place all relationship in a Context.

4.2 Globalization Considerations

Practices in the production process vary from territory to territory and often within a territory for different types of production, e.g., features vs. episodic content. The examples and conventions in these documents are currently biased towards the US film industry and will be expanded in future versions.

The terms themselves are given in English. The Dictionary will provide them in other languages. It will also have synonyms for terms concepts that are referred to by different names (e.g., a Creative Work can be called a Production or a Title) or terms that are used for different concepts (e.g., the different meaning of Series in the US and UK broadcast industries or the multiple definitions of Short.)

5 Future Expansion

Future versions of the Ontology will expand its breadth and depth, driven by industry needs.

Greater breadth – increasing the domains covered – might include VFX tasks and assets, on-set data capture for VFX, or breakdown processes. There are other areas where we would expect to broaden it by connecting to pre-existing ontologies, e.g., to cover legal and contractual matters related to development and production.

Greater depth – adding more detail to things that already exist in the Ontology – can also come in two ways. The Ontology itself can add to existing models, for example by expanding the notion of "character" more fully, adding detail to how one Creative Work is related to others or expanding the set of dictionary terms in a domain. It is also possible to add depth by connecting to existing or emerging ontologies that are specific to the details of a particular domain, such as a more detailed model for video encoding.

Both kinds of expansion are much simpler with connected ontologies than they would be with an ever-expanding monolithic one. In addition to expanding the ontology, we also will be applying it in the creation of data model schemas, e.g., JSON, as part of our other projects in security and software-defined workflows.

Appendix A Ontologies, Dictionaries, and the Like

This Overview has stressed the importance of using well-defined terms and concepts. In that spirit, here are some definitions of the types of some of the documents, and some ancillary terms.

Controlled Vocabulary: This is just a list of allowed terms for use in a particular set of circumstances., e.g., as a value in a data field. This is also sometimes called an Allowed Value Set.

Dictionary: This is a list of terms – often used as controlled vocabulary - with definitions. Dictionaries often include synonyms.[13]

SKOS: SKOS is a technology used for building dictionaries, with the added feature that the items in the dictionary can be related to each other conceptually, indicating, for example, that a term is broader or narrower than another term, or explicitly is or is not the same as another one. The result is a dictionary, but one that partakes a bit more of the lexicographer's art[14] than a simple list does.

Taxonomy: A Taxonomy is a hierarchical organization of things. Class inheritance in data structures and the standard naming systems for biological organisms are both taxonomies. If a Taxonomy just contains words, generally words further down the hierarchy are narrower than the ones above them.

Ontology: Ontologies define concepts, the term used to name them, properties, and connections between those concepts. The connections are called Relationships.

Relationships: In an ontology, relationships can be fully specified: they are named, have directionality, and can themselves have attributes. They can also have allowed domains (the things the relationship is allowed to come from) and ranges (the things the relationship is allowed to connect to).

[13] "Glossary" is a synonym for "dictionary."

[14] "Lexicographer: a writer of dictionaries; a harmless drudge, that busies himself in tracing the original, and detailing the signification of words." - Samuel Johnson, *A Dictionary of the English Language.* Apparently, self-deprecating humor is a job requirement as well.