ONTOLOGY FOR MEDIA CREATION

PART 5: TASKS

VERSION 2.8

Motion Picture Laboratories December 5, 2024

Contents

1 Introduction

What is a task? It was originally an approximate synonym for "tax", derived from Latin *taxare*, "to assess the value of."[1] That usage died out, except in legal records and references, by the early seventeenth century, and the current meaning, a piece of work to be done or undertaken[2], emerged in the fifteenth or sixteenth century,

In the production process, a Task is any piece of work that must be done and completed as a step towards the finished Creative Work. Many of the other pieces of the Ontology converge around Tasks: Tasks are carried out by Participants; Tasks can take Assets as input and produce them as output; most Tasks require Context to be carried out properly and efficiently; some Tasks require particular pieces of Infrastructure. All of these dependencies and requirements can be expressed as relationships inside the Task's Context.

Film and television have always had tasks, even if they were not explicitly defined. For a silent one-reeler, a simple list might do to manage a production: finish the script; hire director and actors; acquire camera and lights; film each scene; edit the selected takes together with intertitles; get final approval; duplicate film for distribution. Some of these steps have to happen sequentially (no filming until actors, director, script, and equipment are available; no approval until the final edit is done), and some can be done in parallel (you can finish the script and find a director in parallel; edit one scene while another is being filmed). Many of these steps are iterative: a scene will be filmed multiple times, until the director is happy; there can be multiple editing passes until final approval is granted. All of these steps can be broken down into smaller pieces, as needed by the Participants actually doing the work: get camera, lights, actors, and director on set, then film the scene; get raw footage and director's notes, edit, then turn the result over for approval.

The simple example given above is a simple, non-automated workflow. Software-defined workflows required well-defined Tasks. With that, the SDW can manage and track not only what needs to be done, but what each component depends on, allowing management and orchestration of Assets, coordination of Participants, and implementation of dynamic security policies. Adding automation and improving workflow efficiency with AI and machine learning are faster and more effective when the Tasks themselves are understood, which only comes as a result of definitions and relation.

Currently, Tasks are assigned to Participants via a fragmented network of project management systems. In the film and television space, the lack of a shared language and data model for Tasks means that basic information about Tasks, such as assignment, status, and dependencies, information, assignment, and status, cannot cross the boundaries between these systems: it must be shared verbally, via email, or through generic scheduling tools that do not understand – or sometime even retain – this information. There are some production tools that allow for customization of workflows with industry specific language, but they are walled gardens attempting to have one system for all Tasks. The ability to have a

[1] The change of the sound of "x" between "ks" and "sk" is common and ancient. "Axe" instead of "ask," for example, is found in *The Canterbury Tales* (late 14th century): "Yow loveres axe I now this question…"

[2] The Cambridge dictionary adds "especially one done regularly, unwillingly, or with difficulty." The first qualifier is relevant to the production process in many cases, though the others are outside the scope of this document.

shared view of a given Task across all the systems used in a production is key to the collaboration tenets of the 2030 vision.

The following sections cover some of the general concepts that must be addressed when defining a general notion of Task for the production ontology.

1.1 Scope

There are often hundreds of Tasks to be completed in the production of a creative work. This Ontology focuses on the Tasks that start with the development of a creative work and go through all the way to the delivery of the final master.[3] All Tasks before and after these tasks will be out of scope.

Some Tasks are generically recognized as part of the production process, while others are very specific to a particular production or kind of production. It is very difficult, or perhaps impossible, to produce a complete list of all kinds of Tasks and therefore we do not try to do so. Rather, we provide a framework within which all those specific Tasks can be described. As the industry converges on some of the more common ones, we will add them as formally defined subclasses of Task, much as the Assets Ontology adds particular Asset Functional Classes.

1.2 Unions and Guilds

There exists several unions and guilds (i.e., organizations) such as IATSE, SAG, ASC, etc., that have attempted to create some standards around Tasks in the film and television industry through the collective bargaining process. This is directly related to the work done by given Participants. There are also non-binding organizations like VES, EBU, SMPTE that have made efforts to create best practices for Task definitions in non-unionized roles. Mapping tables will be provided from the preferred names and descriptions to the various union, guild, and standards representations where applicable.

1.3 Structural and Functional

It is tempting to divide Tasks into Functional and Structural components, as the Ontology has done with Assets and Participants. However, there is not yet any broad – or even partial – consensus on what a Task Structural Class should take as its underlying principles.[4] However, on the chance that such a thing can be found, Tasks have a Task Structural Class for future use. Implementations are free to leave it empty or conduct their own experiments with it; there is some feeling that what makes a structural Task will be different for different parts of the production process.

1.4 Appropriate Granularity

As with many other elements of the Ontology, the principle of appropriate granularity applies to Tasks. This means that a Task can be composed of other Tasks, or broken down into components, each of which is a Task.

---

[3] Or masters. Many productions generate different versions (3D, region-specific background art, etc.) more or less simultaneously with the 'main' production.

[4] Unlike Assets and Participants, where the structural/functional boundary is relatively clear.

For example, write script can be composed of multiple sub-tasks, each of which can be managed as a separate Task if required to support a particular workflow or production. A given production might see "write script" as a repeating loop of sub-tasks (create ideas, review script, and refine script) but to the studio the Task "write script" is the only thing being tracked. Similarly, "shoot scene" can involve performing scenes and stunts as well as capturing picture and sound.

1.5 Relationships

Also, like other elements of the Ontology, individual Tasks can be related to other things. Tasks can have formal relationships to Assets, e.g., as inputs and outputs; to Contexts used to inform the work being performed in the Task; and to Participants, e.g., the entity performing the Task.[5] As with other elements of the Ontology, these tend to be bundled up into one or more Contexts.

Tasks can be connected to other Tasks in multiple ways. A Task can be composed of other Tasks. For example, "script breakdown" is composed of individual departmental breakdown Tasks, and "generate dailies" has multiple steps, including sync sound, create proxies, create package, etc. Tasks can also depend on other Tasks, for example when an animation is composed of individual tasks for lighting, geometry, rigging, and so on.

1.6 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*

[5] See Role in ***Part 4: Participants*** for the reified definition of this Relationship.

- 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.

2 Concepts and Terms

1.7 Task

Task: A piece of work to be done and completed as a step in the production process.

"Write script," "perform scene," and "create visual effects" are all examples of tasks.

A Task can be composed of other Tasks – see *Task Group* below.

Several Relationships that are central to managing and connecting Tasks are enumerated here; they are important enough to be mentioned explicitly.

Sample Attributes

Attribute	**Description**
Identifier […]	One or more identifiers for the Task. At least one of these should be resolvable within the production environment.
Name	The name of this instance of the Task, e.g., "Shoot Scene 1"
Description	A description of the Task
→ Task Structural Characteristics	Currently undefined; see above.
→ Task Functional Characteristics	An instance of *Task Functional Class* (see below).
→ *Task Group*	See below.
→ *Work Unit*	The Work Unit used to perform this task. **See Part 4: Participants**.
State	An indication of the current state of the Task, e.g., "assigned," "complete," "in process" or "waiting"
State Details	Structured or unstructured data with extra application- or Task=specific details about the State, e.g., what the Task is waiting on.
Custom Data	Anything that is application or workflow dependent that can't be otherwise expressed in the Ontology or needs to be present in a particular format.
→ *Context* […]	Any Context needed by this Task. In general, Context is required to be able to run a Task
C→ Scheduling	Information about how a task is scheduled. See below
C→ hasInputAssets […]	Assets that are input for this Task
C→ hasOutputAssets […]	Assets that are output by this Task
C→ Informs […]	Tasks that receive information from this Task. See notes.
C→ isInformedBy […]	Tasks that provide information to this Task. See notes.

Notes:

This Task class is aimed primarily at workflow management applications. The Custom Data field can be used for extra information about an instance of a Task, to be used either while it is running, such as more detailed information about its location (If not covered in Context information), state or resource consumption (if it is being performed by a Service) or afterwards, such as data to feed into AI or machine learning systems.

Tasks and Participants are connected through a Work Unit, defined in **Part 4: Participants**. Production management systems and workflow managers interact with instances of Tasks, often through the Work Unit rather than directly through the Task.

Tasks often deal with Assets, but many Tasks do not. Consider a pair of Tasks for "scout locations" and "choose location." The output of the first is information about possible Production Locations, which serves as input to the second. The output of the second one is a decision about a Production Location. It is possible to treat all of these pieces of information as small Assets, but that may not fit well with existing production processes (send an email with a list of three locations) and databases (choosing a particular location may be as simple as marking a row in a spreadsheet which is not tracked by the production management system.) An even simpler example is asking for and getting approval for something with a phone call. The Informs/isInformedBy relationships cover these non-Asset Tasks. This kind of communication is often called a "trigger," and can be implemented with a variety of techniques including webhooks and messages.

The state of an Asset is different from the state of a Task. A task can change the state of an Asset (e.g., from "draft" to "finished" or "approved"), but that information is stored in the Asset, not the Task – a Task's state relates only to the Task itself.[6]

hasInputAssets and hasOutputAssets can be subclassed for particular kinds of task. For instance, hasVFXSequenceInput might be a subclass of hasInput for a VFX Task.

Context can be very general (just a Scene Descriptor) or very specific (a note that says "add bandage"). Context can also refer to supporting activities defined in the Narrative (Styling, Narrative Audio, Effects, etc. – see **Part 2: Context** for these.)

State is a string, and benefits from having some controlled vocabulary available. StateDetails contains any extra information that a Participant examining the State might find useful. Some implementations may combine state and StateDetails into a single structure, but it is important to have access to simple indication of a Task's state.

[6] Asset state will be covered in a future version of this Ontology.

1.7.1 Scheduling

Scheduling Tasks in a production is a complex process. Some Tasks depend on a single preceding Task being finished; some depend on multiple other Tasks being completed before they can run; some Tasks can decide that a previous task must run again. Basic decisions about running a Task can depend on the state of preceding Tasks. A workflow manager can use the State, StateDetails, and extra data fields of the instances of Tasks it coordinates.

However, keeping a production on track and on budget requires knowing when things should be done, how long it takes them to be done, and when they were actually done. This often depends on when previous Tasks in the workflow were completed. This temporal information is significantly more complex than dealing with the State of a task, and there are many differing sets of practices and terminology for it.

We expect that scheduling information will eventually be built in connected ontologies, but we provide this bare-bones set of information as an example of the kinds of things a scheduling system has to manage.

Sample Attributes

Term	**Definition**
Scheduled Start	The date and/or time a task is scheduled to start.
Actual Start	The date and/or time a task actually starts
Scheduled End	The date and/or time a task is scheduled to end
Actual End	The date and/or time a task actually ends

Notes:

There are many ways to specify dates and times. More formal connected ontologies will specify them more closely.

Scheduled Start may be relative to the Actual End of preceding Tasks

Scheduled End may be relative to the Start Time of this Tasks, e.g., “This task is scheduled to take five days from start to end.”

Maintaining both scheduled and actual start and end allows analysis and investigation of unexpected delays or efficiencies.

Managing dependencies using the State of an instance of a Task requires agreed-on terms and definitions for that State.

1.7.2 Task Functional Class

Task Functional Class: The use or purpose of a Task within the production process.

The Task Functional Class has information about the kind of Task being performed, e.g., Write Script or Create Visual Effects. Details about a particular use of this functional class are kept in the Task itself. For example, for the Task Functional Class Create Visual Effects an instance of a Task could have Context added to cover the task of "Create visual effects explosions for Shot 231 in Sequence 42." The Task Functional Class provides a coarser level of granularity than an instance of a Task, which simplifies the work of scheduling systems, workflow managers, and personnel assignment.

The attributes of a Functional Class are a mix of descriptive metadata and relationships to other things in the production, both of which depend on the particular functional class. See "Functional Classes" below for things defined in other parts of the Ontology that are Task functional classes.

Sample Attributes

Attribute	**Description**
Identifier […]	One or more identifiers for the Tasks. At least one of these should be resolvable within the production environment.
Task Functional Class Name	The name of this Functional Class, e.g., Perform Scene; the Instance name is in Task, rather than in Task Functional Class.
Description	A description of this Task
Custom Data	Anything that is application or workflow dependent that can't be otherwise expressed in the Ontology or needs to be present in a particular format.
→ *Context* […]	Context needed by this Task Functional Class

Notes:

As with Participant Functional Class, in some cases it is sufficient for there to be a single instance of each Task Functional Class used in a production – one instance of Shoot and one instance of Perform Scene, for example. This can simplify some kinds of searches, e.g., "find all Shoots." All of the details of the Task's requirements and dependencies are kept in the Task itself.

1.8 Task Group

Tasks can be composed of other Tasks, which in turn can be further composed of other Tasks. Task Groups cover both composition and decomposition.

For decomposition, consider the breakdown of a given creative work during pre-production. The script must be broken down into catalogues, shots, and scenes. The department heads must provide all of the information required to budget and plan a production. That information has been brought together into one breakdown that can be used to get a green light for a production.

For composition it is often useful to know everything that a single Participant has to do. All of the Tasks associated with that Participant can be put into a Task Group, and then the Task group can be used for finding scheduling conflicts, or if the Participant for all of them changes (e.g., a new Editor is hired.)

Task Group: A Task made of other Tasks, where the assemblage is treated as a single unit.

Sample Attributes

Term	Definition
Identifier […]	One or more identifiers for the Task Group. At least one of these should be resolvable within the production environment.
Name	The name of the Task Group.
Description	A description of the Task Group.
→ Task Components […]	The Tasks that make up the Task Group.

Notes:

A Task Group is usually attached to a Task. However, if the list of Tasks has an independent external existence, e.g., it is managed by a scheduling system that does not use the Ontology, it can have an Identifier and a name to allow at least some level of cross-communication.

1.9 Functional Classes

This section provides a list of some common Tasks in film and television productions. Future versions of the Ontology will provide a larger set either directly or through domain-specific connected ontologies. The number of kinds of Tasks will also continue to grow, as the industry adopts new ways of working (whether socially or technologically driven). The ones included here are in general use, and the names and definitions should be viewed as an example set of best practices.

Task Functional Class Name	Definition	Notes
Write Script	Develop and revise a script.	This is an iterative process.
Develop Creative Style	Prepare the look and tone of the production.	More detailed Tasks related to this include Capture Creative Reference Material
Previsualize	Creative visuals for the creative elements of the work.	
Create Set	Make a Set for use in a Production Scene.	This applies to animation as well as to real-world productions.
Capture Technical Reference Material	Capture technical reference material about the state of the shooting environment during production.	
Perform Scene	Perform actions and dialog.	

Task Functional Class Name	Definition	Notes
Shoot	Capture a shot.	This includes all forms of capture.
Maintain Continuity	Ensure consistency of the narrative in relation to the production across takes and time	
Create Editorial Dailies	Create reviewable video and or audio of a given day's work for the editorial process.	There are other kinds of dailies.
Edit	Assemble shots into a sequence conforming to the narrative.	
Create Visual Effects	Integrate and manipulate live action and/or CG footage to create imagery for the finishing process.	This includes CG-based effects as well as traditional manipulation of the footage.
Create Titles	Create imagery for the contributions to the creative work.	This is a complex process and can cover both opening and closing titles and credits.
Conform Finish	Assemble the final imagery consisting of captured high-resolution material, VFX shots, titling, and opticals based on final editorial decisions.	
Master	Create the various packages for archive and distribution.	
Distribute	Make the creative work available to a public audience..	

Appendix A External Definitions

These are terms defined elsewhere in the Production Ontology, included here for ease of reference.

Media Creation Context: Informs scope within the construction process of a Creative Work.

See **Part 2: Context**

Asset: A physical or digital object or collection of objects specific to the creation of the Creative Work.

See **Part 3: Assets**

Camera Metadata: Capture-specific details and information about the Camera itself.

See **Part 3A: Camera Metadata**

Participant: The entities (people, organizations, or services) that are responsible for the production of the Creative Work.

See **Part 4: Participants**

Task: A piece of work to be done and completed as a step in the production process.

See **Part 5: Tasks**

Creative Work: A uniquely identified production.

See **Part 6: Creative Works**

Relationship: Describes and defines the connections between elements of the Ontology, such as Assets, Tasks, Participants, and Contexts.

See **Part 7: Relationships**

Infrastructure: The underlying systems and framework required for the production of the Creative Work; it is generally not specific to a particular Creative Work.

See **Part 8: Infrastructure**

Utilities: Common data models and data structures used in multiple places and in multiple ways in a larger system.

See **Part 9: Utilities**

Identifier: An identifier uniquely identifies an entity within a particular scope.

See **Part 9: Utilities**