THE EVOLUTION OF PRODUCTION WORKFLOWS

Empowering Creative Processes with Software-Defined Workflows

movie labs

Motion Picture Laboratories, Inc. (MovieLabs) is a nonprofit technology research lab jointly run by Paramount Pictures, Sony Pictures Entertainment, Universal Studios, Walt Disney Pictures and Television and Warner Bros. Entertainment.

MovieLabs enables member studios to work together to evaluate new technologies and helps the industry develop next-generation content experiences for consumers, reduce costs, enhance security and improve workflows through advanced technologies.

CONTENTS

NOTE: No effort is being made by Motion Picture Laboratories or its member studios to in any way obligate any market participant to make use of the information provided in this document. Whether to use the document in whole or in part is left entirely to the individual discretion of individual market participants, using their own independent business judgment. Moreover, Motion Picture Laboratories and its member studios each disclaim any warranty or representation as to the suitability of this document for any purpose and any liability for any damages or other harm you may incur as a result of its use.

EXECUTIVE SUMMARY

The MovieLabs *Evolution of Media Creation* white paper lays out a vision of production workflows that can be more easily assembled, managed, and secured than current workflows. This will result in faster turnaround times, improved iteration, and better collaboration. As stated in the vision, our intention is to enable distributed or remote creative teams to achieve more with their available resources by focusing their talents on creative processes and not on the mundane, everyday tasks that slow them down, such as finding the correct files, securing them, reorganizing them, correcting miscommunications, or understanding the dependencies and status of a task or asset.

This paper builds on the MovieLabs 2030 Vision by providing additional information on software-defined workflows. We describe the essential formalisms and specific mechanisms that can enable flexible workflows with increased automation and interoperability.

Studios and production companies assemble highly individualized workflows and processes tailored to their preferences and to the nature of the production. Nevertheless, many of the crucial elements are shared by different workflows. To a significant and surprisingly useful degree, a workflow can be described generically as *Assets* related as inputs or outputs to *Tasks*, which are performed by *Participants*. For example, *Editing* can be described as a *Task* carried out by an *Editor* that typically has inputs that include *Proxies* and *Camera Logs* and produces outputs that include *EDLs* and *proxy video cut files*. An alternative view is of *Assets* being transformed by *Tasks* into other *Assets*. In that view, *Proxies* and *Camera Logs* are transformed by *Editing* into *EDLs* and *proxy video cut files.* With the proper *Context* (e.g., “Edit the Bridge Scene”), this abstract workflow can be instantiated for a specific purpose. While software-defined workflows are mostly focused on tasks and assets that involve some form of computing, those that exist entirely in the manual/physical domain are also included in our scope.

The production process is a combination of manual and automated processes. Our goal is to support the human creative tasks by connecting them to the greatest extent possible through software-mediated collaboration and automation. This can revolutionize the production process, making it nimble and adaptive. Building on these concepts, we define a framework that allows software to understand and communicate information about workflows. This enables the development of interoperable tools that will support automation in a myriad of areas, such as collaboration, compute and rendering orchestration, asset movement, cost calculation, personnel scheduling, and project dashboards. This paper introduces our approach, which we believe can be immediately applied to workflow and tool development.

SECTION 1
INTRODUCTION

MovieLabs articulated our 10-year vision for the future of media creation in *The Evolution of Media Creation* and *The Evolution of Production Security*, which address migration to the cloud, a new approach to security, and advanced flexible workflows designed to better support the evolving creative process. This paper further expands on the workflow aspects of the 2030 vision.

Of the 10 general principles articulated in *The Evolution of Media Creation*, three focus on advancing workflows (Principles 8–10):

8. Individual media elements are referenced, accessed, tracked, and interrelated using a universal linking system.
9. Media workflows are non-destructive and dynamically created using common interfaces, underlying data formats, and metadata.
10. Workflows are designed around real-time iteration and feedback.

To achieve these principles, it is necessary to develop data and structural foundations on which flexible workflows can be built and the work can be performed, managed, and supported.

Organizations typically have customized workflows consisting of ad hoc collections of tools and processes. These often require custom integration for each project or are disrupted by evolving requirements. Wouldn't it be nice if workflows could be assembled like interconnecting children's blocks, where integration is as simple as connecting the pieces in the desired configuration? While it will never be *that* easy, we can enable more flexible workflows by defining a minimal set of standards and practices for workflow interactions, thereby promoting their interoperability and minimizing the work needed to rapidly create a bespoke workflow. The creatives decide what must be done, and the workflow components are interconnected for them.

Current Workflows

Today's workflows can get the job done, as evidenced by all the great films and TV produced. But there are huge inefficiencies and hidden costs ranging from direct (e.g., cost and time to integrate) to indirect (e.g., fewer iterations possible due to inefficiency) to tragic (e.g., people wasting time on the wrong version of an asset). A workflow that is created for a given project or team addresses the envisioned needs, but once something changes, the workflow must be modified lest it interferes with the creative flow.

"A workflow that is created for a given project or team addresses the envisioned needs, but once something changes, the workflow must be modified lest it interferes with the creative flow.

Adaptability must be designed and built in. Workflows necessarily vary from production to production, and production demands can require changes on the fly. Unfortunately, most of today's workflows are not designed for adaptability. How could they be? The basic pieces on which they are built are not themselves designed for interoperability or interchangeability. The creative process brings together teams who have their own methods and tools, invariably requiring process and engineering work to facilitate collaboration. Even something small, like a lack of file-naming conventions, can delay work. Workflow integration can be difficult, expensive, and time consuming. At the beginning of a project, this can be an annoyance. If a mid-course correction is required (e.g., vendor change), this can be a schedule/budget killer.

Also, a framework for integration provides an opportunity for increased use of microservices. These developments are examples of essential requirements for future workflows.

Future Production Workflows

The future of production will rely on highly configurable workflows that can be continually adapted to support new creative needs of the production, implement new business requirements, or interact with new partnerships. Production teams will design and directly manipulate workflows, and software will manage the processes of collaboration and orchestration.

Future workflows must evolve quickly and correctly, whether the motivation for change is creative, technical, financial, or something else. Anyone designing a workflow will have the ability to choose which tasks are used to perform specific functions, what assets and associated information those tasks communicate, which participants are involved, and what the rules are to move or gate the process. Examples of rules that can be built into workflow automation include “Raw image captured invokes proxy encoding service” and “director’s approval required at this point.”

“Future workflows must evolve quickly and correctly, whether the motivation for change is creative, technical, financial, or something else.

We use the term software-defined workflows (SDW) to broadly describe workflows that fit this model. For discussion, we will use the following definition:

> *A software-defined workflow uses a highly configurable set of tools and processes to support creative tasks by connecting them through software-mediated collaboration and automation.*

Software-defined workflows make it practical to develop reusable components and to automate aspects of the workflow that are currently manual.

SECTION 2
ENABLING SOFTWARE-DEFINED WORKFLOWS

A production is a unique combination of technology and talent that comes together, fleetingly, to create content. Even though each production workflow has unique characteristics, there is still substantial commonality. These commonalities are our opportunity to identify the foundational "building blocks" of workflows and how we can describe them in a machine-readable and machine-actionable manner.

From Formalism to the Final Cut

We are working towards workflows that are flexible, extensible, comprised of reusable components, and always meet the specific needs of the creatives. The production process is complicated and does not lend itself to ad-hoc architectures. A rigorous approach will help ensure that workflow elements will indeed be interoperable and reusable.

The first step in connecting pieces is defining what those pieces are. In this case, they are *assets*, *tasks*, and *participants* (i.e., people and organizations) involved in the media creation process. It is also necessary to define the relationships between them. Workflows ultimately decompose into some combination of these elements, and workflows will be constructed by assembling them in accordance with the project's requirements. Assets, tasks, participants, and their relationships and decompositions are described in the *Workflow Formalisms* section.

The next step is defining the mechanisms that enable tools and automated processes to communicate about work so that people can perform and manage work using the assembled workflow. They need, for example, data definitions, interface definitions, discovery mechanisms, orchestration/coordination mechanisms, work parameters, and other rules. Just as children's blocks fit together, these definitions allow workflow building blocks to work together. For example, if a task produces a frame image, for that image to be consumable by another task, it must be identified, formatted, described, and stored or transmitted in a manner that the consuming task understands. The task and the frame must also have the context of the frame's creation, often relative to the script or another type of breakdown . This is still possible without the common mechanisms, but custom code or plug-ins must be developed for each pair of tools or systems being connected, and that is what we are trying to avoid. As articulated in our security white paper, *The Evolution of Production Security*, security access controls also need to be integrated into and driven by the workflow. The mechanisms that make interoperability practical are described in the *Common Mechanisms* section.

Interoperable tools can be developed using assets / tasks / participants definitions and relationships, as well as mechanism definitions. Interoperability can be achieved by consistently following the definitions. Sometimes this is as easy as following conventions such as file naming. Sometimes it requires refactoring tools and services to comply with standards. The benefits are tremendous: rapidly assembled workflows that support creatives.

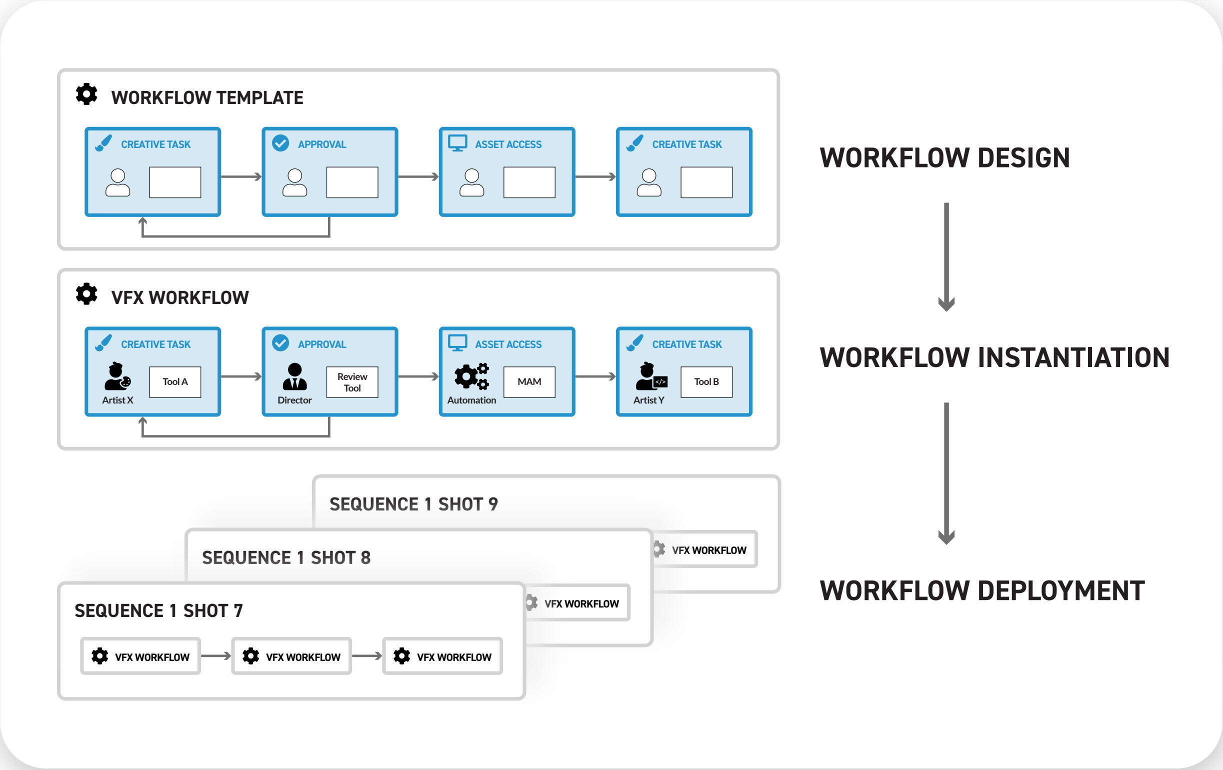
"Note that pipelines employ specific tools and services but only identify types of assets and roles (i.e., not individuals). They are like reconfigurable factories, waiting for materials and staff.

As we move towards implementation, we introduce pipelines, the machinery that enables workflows. Pipelines are sets of tasks, assets, participants, relationships, and mechanisms that perform some defined functions. These range from simple repetitive tasks—good candidates for automation—to complex creative work that requires lots of human interaction, which also benefits from formalism.

To build a pipeline, tasks are instantiated as tools, services, and the skilled people who perform those tasks. Asset types and formats are selected to support the needs of the production. Later, participants are enlisted to perform the specific roles associated with the tasks. Note that pipelines employ specific tools and services but only identify types of assets and roles (i.e., not individuals). They are like reconfigurable factories, waiting for materials and staff.

Finally, we are ready to make movies and TV. Project-specific work is achieved by assigning work to teams and providing the necessary assets. Those teams then use one or more pipelines to do what they do. This may seem like a lot of complexity, but in fact, it's the opposite. Everything along the way is well defined, and the progression from abstractions through effective teams is a manageable progression. This means less gaffer's tape and fewer last-minute surprises.

Workflows progress from abstract models to pipelines to actual work like this:



When it comes to the creative work, the film and TV-making processes have certain well-defined constructs for breaking down work for the purposes of budgeting, planning, scheduling, and execution. Common constructs include script/scene/shot and VFX sequence/shot, but there are conventions for defining almost any activity in the creative process. From the standpoint of software-defined workflows, these constructs can be associated with assets (in and out), tasks, and participants. That association provides "context." In other words, participants perform tasks with assets all within some *context*. The contextual model defines how the asset-task-participant-relationship model is applied to the creative process.

Workflow Formalisms

Formalization Is a Precursor to Software-Defined Workflows

Formalization is the process of creating unambiguous definitions and structure. While human interactions can absorb a certain amount of ambiguity, computer interfaces are, by comparison, not flexible. Without formalisms, it is hard to build systems from components drawn from disparate sources, developed at different times for different purposes.

With formalization, it becomes possible for independently created systems to interact and communicate with each other. The inputs and outputs of applications in a non-formalized environment will need regular remapping to the inputs and outputs of new and changed components. This remapping requires considerable time and effort for every pair of systems, compounding delays, cost, and technical risk. From a system-wide perspective, simply mapping pairs does not scale. A formal model enables common intermediate formats and reduces the number of mappings from one per pair to one per application, and often one per system.

The goal of our formalization is to create an extensible data model that provides common structure but does not constrain what features applications provide or how workflows are designed and assembled when using it. Rigor promotes consistency, stability, scalability, and efficiency. With a well-designed extensible data model, software-defined workflows become feasible.

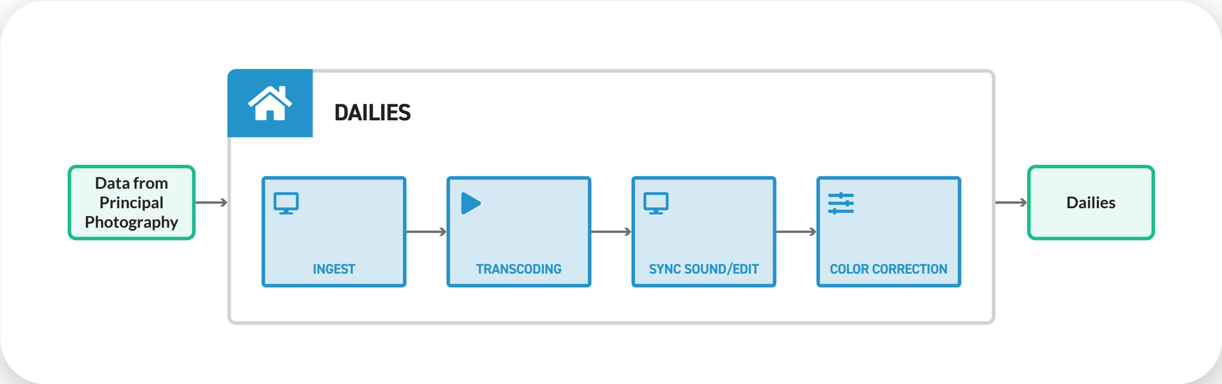
"Rigor promotes consistency, stability, scalability, and efficiency. With a well-designed extensible data model, software-defined workflows become feasible.

Taxonomies, Ontologies, and Controlled Vocabularies

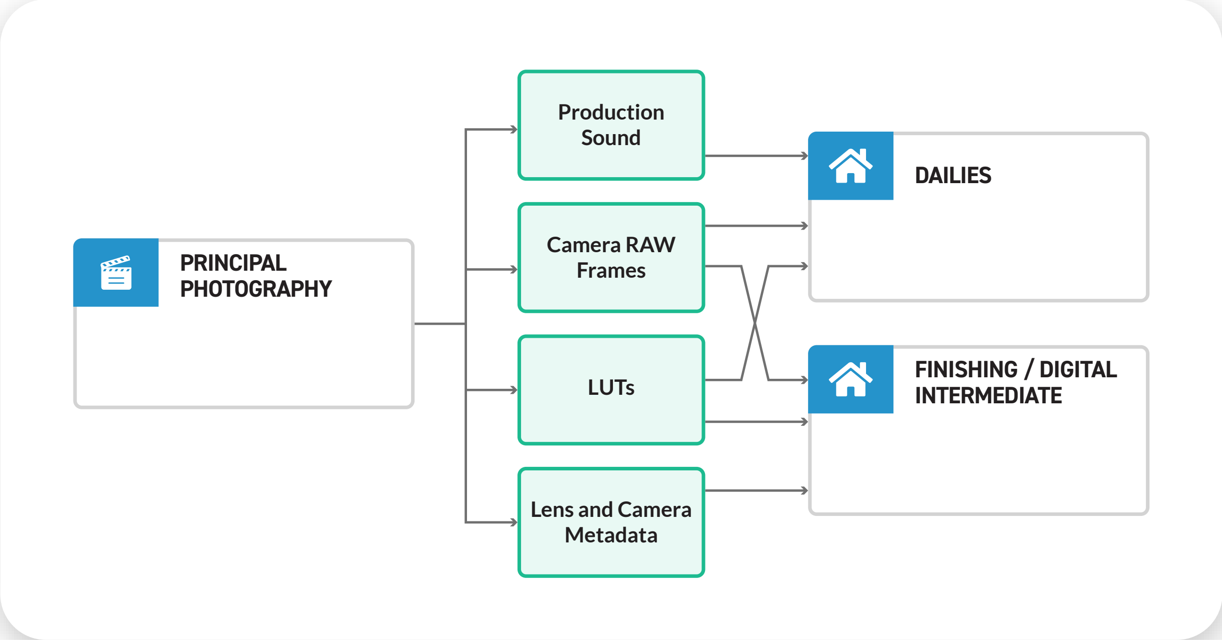
There are two commonly used concepts for formal models. A *taxonomy* is a strictly hierarchical arrangement of items by some notion of type or class (as opposed to a collection of things, such as frames in shot), in concert with an established set of well-defined terms for naming them and a way to determine where in the arrangement a particular item fits. For example, *frames* could decompose into *raw frames*, *plates*, etc.; furthermore, *plates* could decompose into *main plates*, *element plates*, *background plates*, etc.

An *ontology* defines a set of things, the properties of those things, and the relationships between them. The term itself refers to 'the nature of being.' In practice, an ontology defines all essential objects and how they relate to each other. This becomes the language of building workflows, and once you can articulate it, you can build it. In a production workflow, each task, asset, and participant (i.e., people or organizations) can be identified and described at an appropriate level of detail. These objects are connected to each other in the context of production (e.g., a dailies house colorist [participant] creates dailies [task] from camera files [assets], producing dailies videos [assets]).

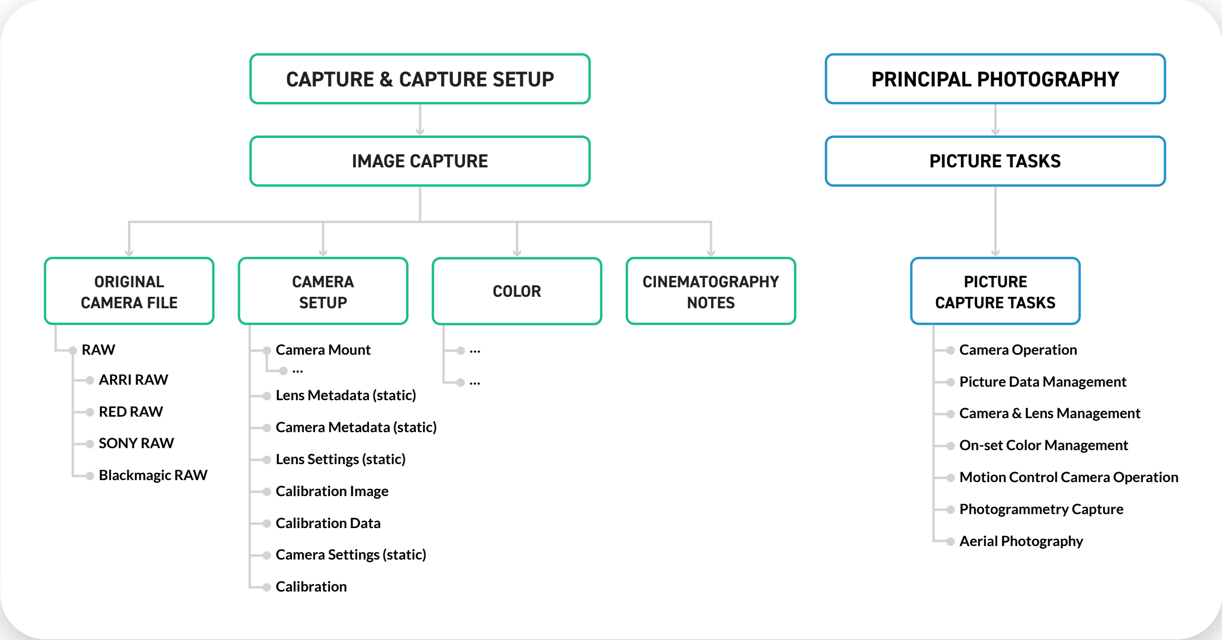
The following shows a *task-centric* view of dailies (assets in green and tasks in blue):



And here is an *asset-centric* view that illustrates assets that go between Principal Photography, dailies, and DI:



You probably noticed these diagrams are not complete. Detail is important, but not to everyone. Sometimes we want to talk about Raw Frames, but in other usage contexts, we need to know if they are ARRI Raw Frames or RED Raw Frames. Sometimes we talk about making proxies, and sometimes we need to dive into debayering. The model must support views at both the high level and lower levels. The following diagram partially illustrates a possible decomposition of assets (on the left) and tasks (on the right) for the same process.



Another important aspect of an ontology is *controlled vocabulary*. Neither people nor computers can communicate without agreed-upon language. However, when people need clarification, they ask. Since computers can't yet ask, we create controlled vocabularies so that they don't have to. Controlled vocabularies are more than a list; they capture additional information about how terms relate. For example, a "Blackmagic RAW Frame" is more specific than a "Raw

> "Another important aspect of an ontology is controlled vocabulary. Neither people nor computers can communicate without agreed-upon language. However, when people need clarification, they ask. Since computers can't yet ask, we create controlled vocabularies so that they don't have to.

Frame." Controlled Vocabularies also capture synonyms (e.g., "model" vs. "mesh") and homonyms/homographs (e.g., "model" for a VFX construct vs. "model" for a person displaying clothes), including the context in which each term applies.

For more explanation and examples of an ontology in the film industry, check out the MovieLabs Creative Works Ontology at https://movielabs.com/creative-works-ontology/. It includes an example of a formal representation of an ontology and various tools to view and understand the ontology.

Building Blocks of the Ontology

When discussing ontologies for workflows involving assets and/or asset management, we talk about *defining* objects and *connecting* them. The basic objects of this MovieLabs Production Ontology are Participants, Tasks, Assets, and Context.

People are the lifeblood of the production process. People interacting within a system are *participants* in the system. From a modeling standpoint, the concept of participants includes individuals, teams, organizations, and, in some cases, automated processes; any of these can control or interact with tasks.

A *task* is an action or group of actions within the production process. A task is performed by people, software, hardware, or a combination of these, and it acts on assets. Tasks can be decomposed into other tasks either as sequential elements in a pipeline or as tasks that can be done in parallel. *Audio Mixing* is a high-level task; *Balance Dialogue* is a more detail-level view of that task. Composition is the reverse, combining fine tasks into coarser ones.

Assets are physical things or digital things, such as props, files (e.g., videos, audios, images, text, and metadata), or records in a database, that can be referenced (e.g., an inventory number for a costume) or have digital representations, such as a 3D model.

Assets have structural characteristics (e.g., "it is an image") and functional characteristics (e.g., "it is a VFX Plate"). These are represented as metadata. Also included in metadata is information about the creation of that asset, its relationship to the project (e.g., scene/shot), and its disposition (e.g., approvals). Many assets are just data (e.g., customer-facing metadata).

Relationships connect things, in this case, assets, tasks, and participants. They describe the nature and direction(s) of the relationship (e.g., "model is created by artist," "model is used in scene," and "scene is shot in location"). The relationships model forms the glue by which workflow elements connect. In many cases, the relationships become the basis for APIs.

Context addresses purpose and intent, often corresponding with story and character development, script breakdown, VFX work breakdown, and various other activities within the overall process, from concept through distribution and ultimately archive. Wherever there are tasks, assets, and participants, there will also be context.

Note that the benefits of the formalisms go beyond just software-defined workflows. Controlled vocabularies, task/asset/participant definitions, relationships, and other information provide a basis for consistent communication (both human and computer), improved production management, and greater consistency in technical definitions.

Common Mechanisms

Software-defined workflows abandon the notion that interoperability is limited to applications that are designed specifically to work together. Instead, we adopt the model that applications can interoperate with any other as long as they follow a set of interoperability rules and security policies, either natively or with adapters. When we talk about mechanisms for supporting software-defined workflows, we are referring to those rules. If the data and messages are mutually intelligible and the mechanisms for interaction are well defined, applications can work with each other, discover each other, and share data as required.

> "Software-defined workflows abandon the notion that interoperability is limited to applications that are designed specifically to work together.

One approach is to create standardized APIs and schemas, and those are certainly part of this solution. However, we are addressing interaction at a more foundational level so that other API and schema developers can have common concepts and language to build upon. Building on this foundation will allow these developers to spend less time building foundations and more time focusing on the unique aspects of their services.

Software Design Workflow Structural Components

Software-defined workflows are based on standardization of certain interfaces and functionality. For example, given high-resolution camera images, one can create lower-resolution video (proxies) using standard components. However, for this to work, the inputs, outputs, and conversions must be understood and predictable (i.e., sufficiently standardized).

Architecturally, there is a rich collection of methods that must be addressed. The following are some areas being considered by MovieLabs for common definition:

- Identification – Assets, tasks, participants, services, and other components of the system must be identified (possibly by a variety of methods).
- Object retrieval – Given the identity of an asset, task, or service, there must be a means to locate and then retrieve it.
- Non-destruction – One of the sub-principles of software-defined workflows is that assets, once created, are not modified (i.e., workflows are non-destructive). In some cases, modifications or transformations come in the form of metadata (e.g., a separate graphic contains a director's note on a drawing, or color adjustments are captured in a LUT). New assets are real-time transformations of the original asset using metadata. In other cases, an asset is created and becomes part of the asset inventory. Our model will support both destructive and non-destructive mechanisms.
- Persistent metadata – Metadata must describe an asset's persistent, possibly evolving properties. This falls into multiple categories, some of which are:
    - Descriptive and technical metadata – What the asset is and how it is encoded.
    - Object derivation, history, and provenance – Whether the asset is original (e.g., a camera frame) or was derived from another asset (e.g., a proxy from a camera frame). If it was derived, what was done (e.g., it was debayered, or a specific LUT was applied). Provenance describes those involved in an object's history. This information can be used to trace any object to its origin (e.g., original raw camera frame).
    - Creation and usage context – Information about the asset's creation or use, such as work, scene, shot, camera, take, and filming location, and its original intended use (e.g., plate for VFX shot 123). This also includes versioning.

- Active metadata (e.g., approvals, comments, etc.) – Data associated with the current state or use of an asset. This includes status (e.g., director's approval), as well as data associated with assets (e.g., director's *drawn* notes on an image).

- Workflow management and orchestration

  - Resource management – Resources to monitor and manage both human and technical (storage, computational) resources.

  - Notification – Communication of events between entities.

  - Security management – Tools to control access and monitor integrity.

  - Disposition/approval management – Managing the status of workflow elements based on policy and approvals.

Although not a specifically required tool for software-defined workflows, the microservices model is highly relevant. Microservices, a term whose definition is beyond the scope of this paper, encapsulate functionality that is easy to access and easy to configure into flexible workflows. One can envision certain workflows built largely on microservices.

"The term pipeline is commonly used to describe sets of software, people, and processes that perform a function. We will use that term, but with a stricter definition: Pipelines are subsets of software-defined workflows that can be used across projects.

Building Pipelines

While the previous sections addressed building blocks, we now talk about combining those blocks into something useful. The term pipeline is commonly used to describe sets of software, people, and processes that perform a function. We will use that term, but with a stricter definition: Pipelines are subsets of software-defined workflows that can be used across projects.

Combining Tasks, Assets, and Participants

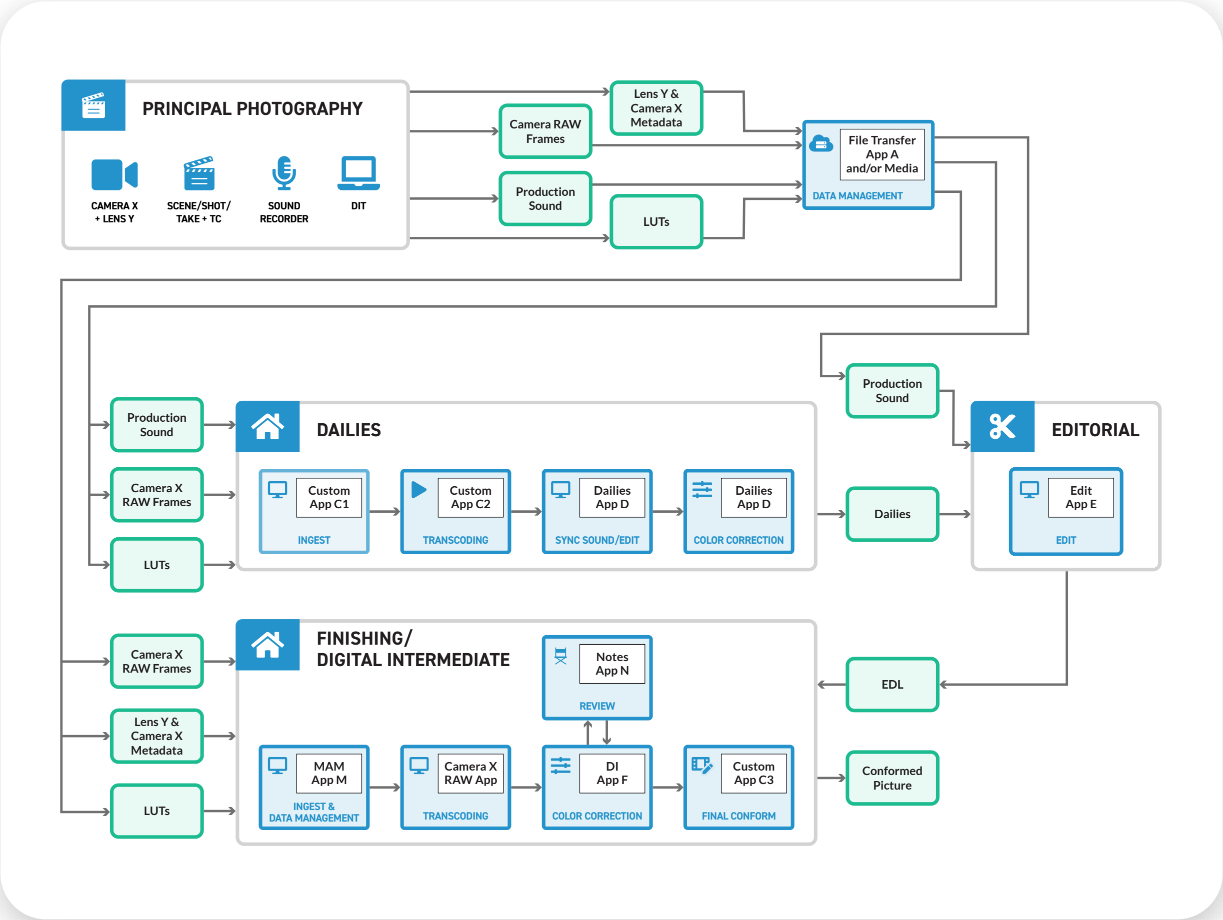
Keeping in mind that pipelines are an intermediate step, pipelines define some things and only partially define others. Tools and services are defined as actual software. People are defined by job function, but actual people will be assigned later. Assets can be constrained to particular encodings (e.g., OpenEXR file), but actual assets appear later (e.g., VFX Shot 231, Plate frame 1721).

Formalisms define the generic building blocks and their interconnections. Building on these formalisms, pipelines instantiate tasks, asset types, and the connections between tasks as tools, files, and APIs. More formally, pipeline definitions constrain the abstraction defined in the ontology. Interfaces follow relationships. Tools and services align with sets of tasks. Asset variants are constrained within their functional and structural types (e.g., image/plate). Participant roles are identified and constrained within their definitions and interact along relationship lines.

Notional Pipeline

This following is intended to illustrate how a pipeline might look. In this example, building blocks are software tools and services with standard interfaces. These do much of the automated heavy lifting in the pipeline. Tools might use microservices to perform certain functions.

A pipeline design connects tools and services, sequenced in the manner desired by the creators. The designer accounts for tasks, assets, and participants—the people or organizations who actually drive the process. The workflow designer also captures workflow rules (e.g., dailies require the approval of person X before they can be distributed). The software and data elements of a workflow might look something like the following illustration.



Every line in the picture corresponds with an interaction between workflow elements. When these are well defined, swapping out components becomes easier and more predictable, stable, and secure. When formats and interfaces are bespoke, custom engineering is required for each change.

Although not illustrated above, software-mediated collaboration is achieved through an 'orchestration system' that coordinates activities in the workflow. When resources are available for a task, the task or participant is notified that it can begin.

> "When workflow elements are well defined, swapping them out becomes easier and more predictable, stable and secure. When formats and interfaces are bespoke, custom engineering is required for each change.

The orchestration system also keeps track of the schedule and raises flags when work is not as planned. Workflow design and orchestration can be supported by suitable tools (e.g., a drag-and-drop workflow designer).

Ready to Make Movies and TV

Location 3, Day 1, Scene 24. Everyone is on set. Everything is on set. Cameras rolling. Sound recording. Slate! Action! The pipeline illustrated in the previous section is built, ready, and staffed, so we're ready to start processing the data that comes off the camera. Work is proceeding as defined by the assistant director and production accountant. Elsewhere, the marketing team is busy developing concepts for collateral. A previous animator begins animating the main character. One actor has shown up unshaven, so later, a VFX artist will touch up his facial hair.

> "Location 3, Day 1, Scene 24. Everyone is on Set. Everything is on set. Cameras rolling. Slate! Action!

Along the way, there are numerous production, mastering, and distribution activities. Ultimately, an archivist will catalog data and prepare it for long-term storage.

Each of these examples represents the context for work performed. Assets are consumed and/or produced by each task within this context. Tasks are performed by participants using tools assembled into pipelines. These processes can be performed efficiently and predictably because they are built as software-defined workflows.

SECTION 3

CONCLUSION

In this paper, we expand on the concept of software-defined workflows originally defined in our 2030 Vision, which are built around a set of common components. We also outline an approach to give developers the ability to provide creatives with the workflows they want in a timely and efficient manner. The key components are formalisms and specifications for mechanisms that use those formalisms and are developed through industry collaboration. Although not discussed in this paper, we expect these will be supported by open-source software libraries to ease implementation and to promote interoperability and adoption. Ontologies, taxonomies, and controlled vocabularies will be made available for use in APIs and other mechanisms. At some point, APIs and the industry as a whole will benefit from a more structured or standardized environment in which they are developed and integrated.

Our approach is incremental, producing useful near-term results and benefits while building towards the vision. The 'ontology' will be built as a family of connected ontologies, with the initial ontologies addressing overall structure as well as portions of the media creation process where near-term solutions are needed. Mechanism definition will be prioritized to support active developments where common approaches will be most beneficial.

Collaboration is essential to the successful development and adoption of software-defined workflow technologies. As stated in our 2030 Vision, we fervently believe that success will be best achieved if all potential beneficiaries work together appropriately to advance cross-industry goals.

CONTRIBUTORS

This paper would not have been possible without the invaluable support and contributions of the following executives:

Shadi Almassizadeh, *Disney*

Bill Baggelaar, *Sony Pictures*

Richard Berger, *MovieLabs*

Annie Chang, *Universal Studios*

Daniel De La Rosa, *Sony Pictures*

Daphne Dentz, *Warner Bros*

Eddie Drake, *Disney*

Raymond Drewry, *MovieLabs*

Barbara Ford-Grant, *MovieLabs*

Anthony Guarino, *Paramount Pictures*

Jim Helman, *MovieLabs*

Renard Jenkins, *Warner Bros*

Josh Limor, *Paramount Pictures*

Daniel Lucas, *MovieLabs*

Horst Sarubin, *Universal Studios*

Craig Seidel, *MovieLabs*

Steven Shapiro, *Disney*

Doug Sherman, *Universal Studios*

Adam Slohn, *Warner Bros*

Spencer Stevens, *MovieLabs*

Mark Turner, *MovieLabs*

Matt Walters, *Disney*

Chris Vienneau, *MovieLabs*

MovieLabs is a jointly run industry lab that enables member studios to work together to understand new technologies and drive the right ones to adoption. We help set the bar for future technology and then define specifications, standards, and workflows that deliver the industry's vision. Our goal is always to empower storytellers with new technologies that help deliver the best of future media.

Email: info@movielabs.com
Twitter: @ MovieLabsNews
Website: movielabs.com

HEADQUARTERS
222 Kearny St, Suite 300, San Francisco, CA 94108